A COMPUTER PROGRAM FOR THE BEARING CAPACITY ANALYSIS OF A SINGLE PILE BASED ON SPT WITH APPLICATIONS

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of
Dokuz Eylül University
In Partial Fulfillment of the Requirements for
the Degree of Master of Science in Civil Engineering, Geotechnics Program

by
Okan ÖNAL

July, 2001
IZMIR

M.Sc THESIS EXAMINATION RESULT FORM

We certify that we have read this thesis and **"A COMPUTER PROGRAM FOR THE BEARING CAPACITY ANALYSIS OF A SINGLE PILE BASED ON SPT WITH APPLICATIONS"** completed by **Okan ÖNAL** under supervision of **Prof. Dr. Arif Şengün KAYALAR** and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Arif Şengün KAYALAR

Supervisor

Prof. Dr. Necdet TÜRK

(Committee Member)

Yrd. Doç. Dr. Gürkan ÖZDEN

(Committee Member)

Approved by the

Graduate School of Natural and Applied Sciences

Prof.Dr. Cahit HELVACI
Director

ACKNOWLEDGMENTS

Pile foundation is very often applied in Izmir region, and Standard Penetration Test is the most frequently used soil investigation method, by which some of the soil parameters can be correlatively determined.

In this thesis, a computer program based on Standard Penetration Test has been prepared for the estimation of the ultimate axial bearing capacity of a single pile.

The author is thankful to the advisor of this thesis, Prof. Dr. Arif Şengün KAYALAR for his valuable guidence throughout the completation of this thesis.

He is also thankful to Özgür BAŞARAN, for his proposals in applying Visual Basic programming language, and to his father İsmet ÖNAL for printing the thesis.

The author hopes that the computer program prepared in this study will be useful in solving some of the geotechnical engineering problems.

ABSTRACT

The ultimate bearing capacity of a foundation pile can be estimated by several different approaches.

Standard Penetration Test being the most frequently used soil investigation technique in Izmir region and all over the World, provides a resistance profile for the estimation of single pile bearing capacity.

In this thesis, a computer program for the analysis of the ultimate axial load bearing capacity of a single pile based on the Standard Penetration Resistance (SPT-N) profile has been prepared in Visual Basic 5.0 programming language. In the program, the soil layers has been divided into sub layers in order to obtain detailed results. The program has been written so that the results and resulting graphs can be sent to a printer.

In order to test the program, runs with the data of thirty-two boring logs from eight different locations surrounding the Izmir Bay has been performed, and a tentative regional modeling of ultimate bearing capacity has been obtained.

Computer Program provides the calculation of the ultimate axial bearing capacity of a single pile in a short time. Thus, a regional modeling can be formed very quickly, if there is enough penetration resistance data

ÖZET

Bir temel kazığının nihai taşıma kapasitesi bir çok yaklaşımla bulunabilmektedir.

Standart Penetrasyon Deneyi, İzmir çevresinde ve dünyada yaygın olarak kullanılan bir zemin araştırma yöntemidir. Bu deney ile, kazığın taşıma gücü hesabında yararlanılan, zeminin direnç profili elde edilmektedir.

Bu tez çalışmasında, tekil bir kazığın eksenel nihai taşıma gücü hesabını Standard Penetrasyon Direnci profiline dayanarak yapma imkanı sağlayan bir bilgisayar programı hazırlanmıştır. Programlama dili olarak Visual Basic 5.0 kullanılmıştır. Programda, zemin katmanları bir çok alt tabakaya ayrılarak detaylı analiz yapılabilmektedir. Program, sonuçları ve grafikleri bir yazıcıdan verecek şekilde oluşturulmuştur.

Programın sağlıklı çalıştığını gösterebilmek için İzmir Körfezi çevresinde yer alan 8 ayrı lokasyona ait 32 adet sondaj kuyusu verisi programa girilerek, sonuçları alınmış, yöresel, geçici bir kazık taşıma kapasitesi modeli yan ürün olarak elde edilmiştir.

Program tekil bir temel kazığının nihai taşıma kapasitesinin ayrıntılı ve hızlı bir şekilde hesaplanmasını sağlamaktadır. Yeterli penetrasyon direnci verisi olması halinde, bir bölge için kazık taşıma kapasitesi modellemesi hızlı bir şekilde oluşturulabilmektedir.

CONTENTS

Chapter Four

COMPUTER PROGRAMMING

Chapter Five

APPLICATIONS

Chapter Six

CONCLUSIONS

REFERENCES

APPENDIXES

LIST OF TABLES

Page

LIST OF FIGURES

CHAPTER ONE
INTRODUCTION

The ultimate bearing capacity of a foundation pile can be estimated by several different approaches.

The Standard Penetration Test (SPT), which is applied widespread during soil investigations in Izmir region and all over the World provides a resistance profile for the calculation of single pile bearing capacity.

In this thesis, a computer program in Visual Basic 5.0 for the analysis of the ultimate axial load bearing capacity of a single pile based on the Standard Penetration Resistance (SPT-N) profile has been prepared. The program supports stepwise calculation of the bearing capacity, tabular arrangement of the results and their graphic presentation starting from the ground surface down to the last SPT depth.

Data inputting forms have been created. Save and load procedures have been formed for saving a project on the hard drive or for loading a previously saved project.

For detailed calculation mode, the program divides the main (geological) layers of the soil profile existing on the boring log into sub layers by taking into consideration the representative SPT depths. A code for the calculation of the effective overburden pressures and for the correction of SPT-N values has been prepared.

The computer program, accepts the main soil layers on the boring log only in two categories as sand or clay, which is decided by the user according to their common behavior.

For a sand layer, two base and two skin friction methods have been considered in the program. In the first method, which is a correlative method by Meyerhof, the base and skin friction resistances are calculated with the corrected Standard Penetration numbers. The second method is the Static Formula Method, in which the internal friction angle of sand layers have been correlatively calculated with the aid of corrected SPT-N values.

In the case of a clay layer, Hansen's Method (which is a total stress analysis approach) is adopted for the prediction of the base resistance. The undrained shear strength parameter c_u is estimated with known correlation equation of the corrected SPT-N values. Either total stress analysis approach (α–method) or effective stress analysis approach (β–method). has been offered to the user for skin friction calculations.

It has also been arranged that any value of the shear strength parameters can be entered in the program instead of the values obtained by correlation equations.

Correction of the base resistances has been supplied with two different methods. Namely, Meyerhof's Method and the Averaging Method.

The program is capable of plotting graphs of the soil profile, effective overburden pressure, raw and corrected SPT-N values, and the ultimate bearing capacity versus depth.

The computer program has been run with the data of thirty-two boring logs from eight different locations selected to be located at points surrounding the Izmir Bay. The results have been examined for testing the performance of the computer program. As a by-product of these applications, a small tentative regional modeling of ultimate bearing capacity has been presented.

In the thesis, brief theoretical explanations are given in Chapter Two and Chapter Three. In Chapter Two, an outline of the ultimate bearing capacity of a single pile

and the formulations used in the programming has been presented. The Standard Penetration Test, its corrections and correlations have been mentioned in Chapter Three. The details of the computer program have been given in Chapter Four. Chapter Five includes the details of the applications and their results.

The flow chart of the computer program has been presented in Appendix A. Appendix B contains the code of the program. A user manual is given in Appendix C, and a Sample Application has been presented in Appendix D. Appendix E contains resulting graphs of the application section (Chapter Five). The trend lines and curves, which have been prepared in Microsoft Excel, have been presented in Appendix F. The program CD has also been given as attachment of the thesis.

CHAPTER TWO
BEARING CAPACITY OF PILES

2.1 Pile Types According to Method of Installation

Piles are structural members used for transmitting loads to lower levels in the soil mass. Pile material may be timber, concrete, and/or steel. Piles can be installed with two different methods.

In the first method, the installation causes displacement and disturbance of the soil around the pile. These piles are driven piles of steel or precast concrete. Driving tubes or shells, which have a driving shoe, can also form them. These tubes and shells can be either filled with concrete after driving and leaved underground or can be withdrawn during the concrete filling process for further use.

In the second method, the installation causes no soil displacement. These are bored piles. To form a shaft, soil is removed by boring and then concrete being cast in the shaft to form the pile. The shaft may be cased or uncased depending on the type of soil.

2.2 Bearing Capacity

The ultimate axial load-carrying capacity of a pile can be given by a simple equation as the sum of the load carried at the pile point plus the total frictional resistance derived from the soil-pile interface. The base resistance is the product of the base area A_b and the ultimate bearing capacity q_f at base level. The shaft resistance is the product of the perimeter area of the shaft A_s and the average value of ultimate shearing resistance per unit area f_s. fs is generally referred to as the 'skin friction', between the pile and the soil.

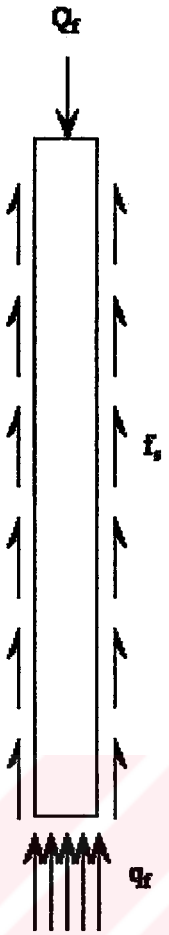


Figure 2. 1 Load carrying mechanism of a pile

The weight of soil displaced or removed is generally assumed to be equal to the weight of the pile. Thus, the ultimate load Q_f, which can be applied to the top of the pile, is given by the equation:

$$Q_f = A_b q_f + A_s f_s \tag{2.1}$$

To determine the allowable load on the pile, an appropriate load factor is applied to Q_f (Craig, 1992).

In case of several layers along the pile, the total skin friction resistance f_s is equal to the sum of the skin friction resistance of each layer as given below:

$$f_s = \sum_{i=1}^{n} f_{s,i} \times A_{s,i} \tag{2.2}$$

Where, $f_{s,i}$ and $A_{s,i}$ are the skin friction and surface area for the i^{th} layer, respectively, n is the number of layers.

Ultimate bearing capacity of piles in sand and in clay is evaluated generally with different approaches. These are discussed in the following two subsections.

2.3 Piles in Sands

The ultimate bearing capacity of a pile in sand depends on the relative density. In case of driven piles, relative density adjoining the pile changes. It is increased in loose sands due to soil displacement, and decreased in dense sands, which may be loosened.

The ultimate bearing capacity of a pile in sand at base level is

$$q_f = \sigma_o' N_q \tag{2.3}$$

Where, σ_o' is the effective overburden pressure at base level, N_q is the bearing capacity factor (Craig, 1992).

The bearing capacity factor N_q, can be expressed by the following formula (Bowles, 1996):

$$N_q = e^{(\pi \tan \phi')} \tan^2 (45^\circ + \phi'/2) \tag{2.4}$$

Skin friction can be expressed as

$$f_s = K_s \sigma_o' \tan \delta \tag{2.5}$$

Where, K_s is the coefficient of lateral earth pressure, σ_o' is effective overburden pressure, and δ is angle of skin friction between the pile and the sand (Craig, 1992).

In Equation (2.5) effective over burden pressure is taken as an average value for a layer.

Relationship between $K_s \tan \delta$ and ϕ'; for driven and bored piles is shown in Figure 2.2.

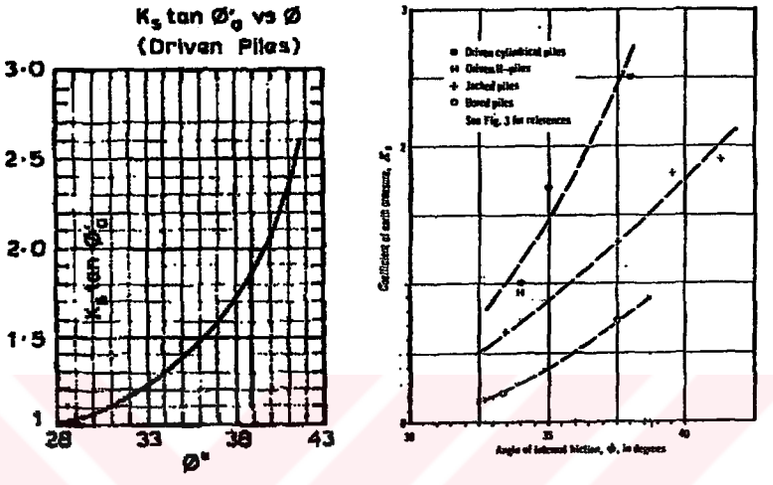


Figure 2. 2 Relationships between K_s and ϕ'. (a) $K_s \tan \delta$ and ϕ' relation for driven piles ($\phi = \frac{3}{4} \phi'_1 + 10$) (After Poulos & Davis, 1980), (b) K_s and ϕ' relation for bored piles (After Meyerhof, 1976).

In homogenous sands both point resistance and average skin friction of a pile would increase with greater depth of penetration. However, large-scale experiments and field observations have shown that the theoretical relationship hold only when the pile point is above a certain critical depth z_c. Below this depth, the point resistance and average skin friction remain practically constant in a homogenous soil due to effects of soil compressibility, crushing, arching, and other factors (Meyerhof, 1976, p198).

Relationship between critical depth z_c over pile diameter D and ϕ' is shown in Figure 2.3.

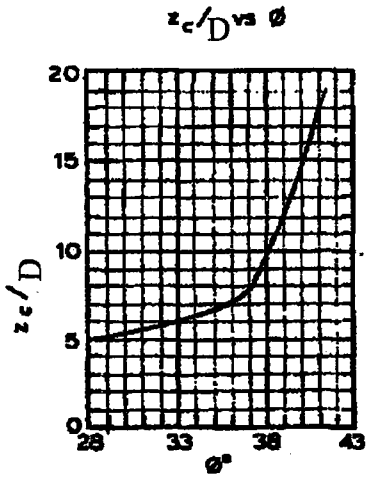


Figure 2. 3 Relationship between z_c/D and ϕ'. (For driven piles $\phi = ¾\ \phi'_1 + 10$, for bored piles $\phi = \phi'_1 - 3$) (After Poulos & Davis, 1980).

Due to the critical depth limitation and to the problem of obtaining values of the required parameters, Meyerhof has proposed the following correlations to estimate values of q_f and f_s for piles driven into a sand stratum:

$$q_f = 40\ N\ D_b/B \leq 400\ N \tag{2.6}$$

Where, N is the statistical average of the SPT-N numbers in a zone of about 8B above to 3B below the pile point, and D_b is the length of the pile embedded in the sand. Also the skin friction is,

$$f_s = 2N \tag{2.7}$$

In the above equations q_f and f_s are in kN/m^2 (Bowles, 1996).

2.4 Piles in Clays

In the case of driven piles, the clay adjacent to the pile is displaced both laterally and vertically. The clay in the disturbed zone around the pile is completely remoulded during driving. The excess pore water pressure set up by the driving stresses dissipates within a few months, as the disturbed zone is relatively narrow

(of the order of D). Dissipation is accompanied by an increase in the shear strength of the remoulded clay and a corresponding increase in skin friction (Craig, 1992, p338).

Ultimate bearing capacity of piles in clay can be evaluated either with total stress analysis or with effective stress analysis. Total and effective stress analysis methods are briefly explained below.

2.4.1 Total Stress Analysis

In the case of bored piles, a thin layer of clay (of the order of 25 mm) immediately adjoining the shaft will be remoulded during boring. In addition, a gradual softening of the clay will take place adjacent to the shaft due to stress release, pore water seeping from the surrounding clay towards the shaft. Water can also be absorbed from the wet concrete when it comes into contact with the clay. Softening is accompanied by a reduction in shear strength and a reduction in skin friction. Limited reconsolidation of the remoulded and softened clay takes place after installation of the pile.

The relevant shear strength for the determination of the base resistance of a pile in clay is the undrained strength at base level. The ultimate bearing capacity is expressed as

$$q_f = c_u \times N_c \qquad (2.8)$$

Where, c_u is the undrained shear strength and N_c is the bearing capacity factor.

Based on theoretical and experimental evidence, a value of N_c of 9 is appropriate (i.e. Skempton's value for D/B > 4).

The skin friction can be correlated empirically with the average undrained strength c_u of the undisturbed clay over the depth occupied by the pile, i.e. (Craig, 1992, pp338, 339).

$$f_s = \alpha \, c_u \tag{2.9}$$

Where, α is adhesion factor. The values of α as a function of undrained shear strength are given in Figure 2.4.

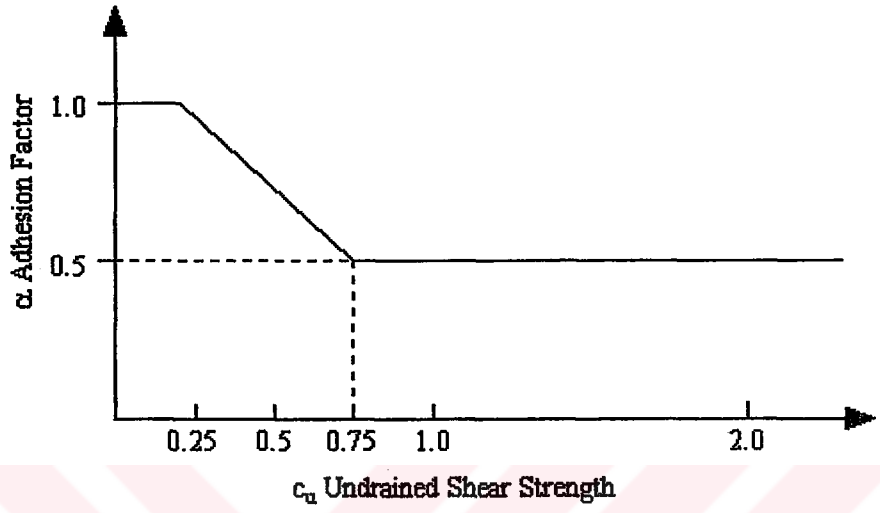


Figure 2. 4 Values of α versus undrained shear strength (After ASCE,1993)

Due to α, this method is called as α-method.

2.4.2 Effective Stress Analysis

An alternative approach is to express skin friction in terms of effective stress. The zone of soil disturbance around the pile is relatively thin; therefore dissipation of the positive or negative excess pore water pressure set up during installation should be virtually complete by the time the structural load is applied. In principle, therefore, an effective stress approach has more justification than one based on total stress. In terms of effective stress the skin friction can be expressed as

$$f_s = K_s \, \sigma_o' \tan \phi' \tag{2.10}$$

Where, K_s is the average coefficient of earth pressure and σ_o' is the average effective overburden pressure adjacent to the pile shaft. Failure is assumed to take

place in the remoulded soil close to the pile shaft, and therefore the angle of friction between the pile and the soil is represented by the angle of shearing resistance in terms of effective stress (ϕ') for the remoulded clay: the cohesion intercept for remoulded clay will be zero. The product $K_s \tan \phi'$ is written as a coefficient β, thus

$$f_s = \beta \sigma_o' \tag{2.11}$$

Approximate values of β can be deduced by making assumptions regarding the value of K_s, especially in the case of normally consolidated clays. However, the coefficient is generally obtained empirically from the results of load tests carried out a few months after installation. For normally consolidated clays the value of β is usually within the range 0,25 to 0.40 but for overconsolidated clays values are significantly higher and vary within relatively wide limits (Craig, 1992, p339).

Due to β, this method is called as β-method.

2.5 Base Resistance Corrections

The purpose of the base resistance correction is to soften the base resistance values in passing between the layers, which have different base resistances.

Two different methods for base resistance correction exist. The first method has been proposed by Meyerhof (1976). The following formulation has been applied to the base resistances:

$$q_{b,c} = q_{b,w} + ((q_{b,l} - q_{b,w})\, H / 10\, D) \leq q_{b,l} \tag{2.12}$$

Where $q_{b,c}$ = Corrected base resistance, $q_{b,w}$ = Base resistance in weak soil, $q_{b,l}$ = Limit base resistance in dense sand, H = Distance of the pile base to the weak layer, and D = Pile diameter as shown in Figure 2.5.

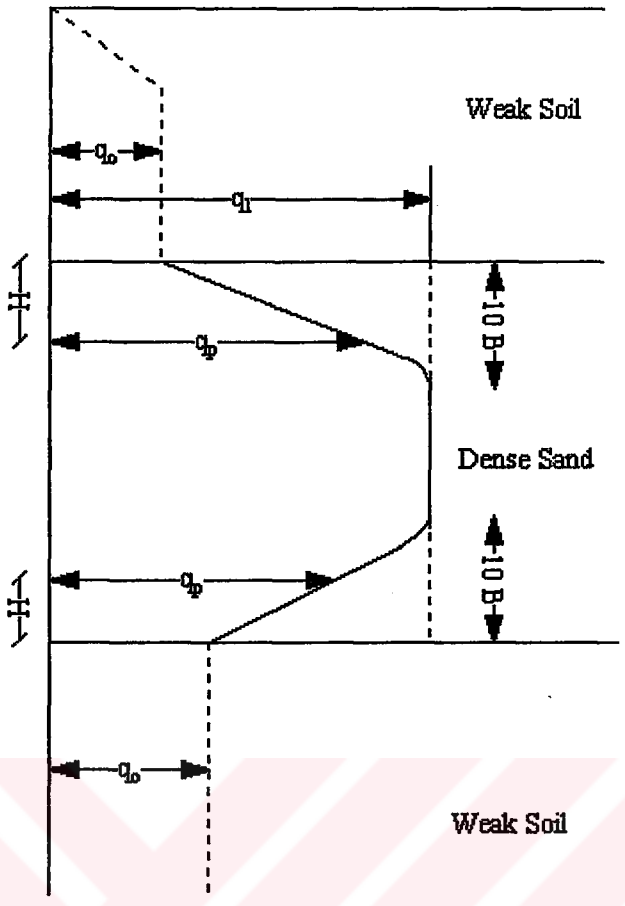


Figure 2. 5 Base resistance correction method by Meyerhof.

In the second method, average base resistance in the vicinity of the pile base is taken into account as corrected base resistance. Generally, average of the base resistances within a depth interval of three pile diameters below and five pile diameters above the pile tip is calculated.

First method is preferred when the difference in base resistances in weak soil and dense sand is high. When the differences in base resistances in the vicinity of pile tip is not very high, then the second method is used.

CHAPTER THREE
STANDARD PENETRATION TEST AND ITS CORRELATIONS

3.1 Standard Penetration Test

Standard penetration test is performed during a boring to obtain an approximate measure of the soil resistance to dynamic penetration, and to get a disturbed specimen of soil from the desired depth. It is currently the most common, economical and easy test method to obtain subsurface information. Altough the test can be performed in a wide variety of soils, its results are more consistent for sandy soils where large gravel particles are absent. The method has been standardized as ASTM D 1586.

To perform the test, the drilling crew, after reaching the desired depth, first removes the string of drill rods. The borehole must be cleaned out until the required depth. In this grade, to prevent the water flow in the borehole, water should be added, if the test is to be performed under the ground water level. If the head of water is below the ground water level, piping can occure at the bottom of the hole which can loosen the soil and invalidate the test result. After the drilling rods are removed, a standart 51 mm (2 in) outer diameter split spoon sampler, as shown in Figure 3.1, is attached to the drill rods and lowered carefully to the bottom of the hole. With the sampler resting at the bottom of the hole, a 63.6 kg (140 lb) weight is allowed to fall freely 762 mm (30 in) onto a collar that is attached to the top of the drill string until 450 mm (18 in) of penetration has been achieved. The boring log shows refusal and the test is halted if 50 blows are required for any 150 mm increment, 100 blows are obtained to drive the required 300 mm or 10 successive blows produce no advance.

For each 150 mm penetration the blow counts are recorded. The first 150 mm interval is named as seating drive. This interval is necessary to seat the device and to by-pass any disturbed sand at the bottom of the borehole. The number of blows to pass the other two 150 mm intervals is named as the standard penetration resistance or N value. After the sampler has been brought back to the surface, the samples are removed and classified before being placed into jars, labeled, and sealed with wax for transport (Kulhawy & Mayne, 1990).

Tests are normally carried out at intervals of between 0.75 m and 1.5 m to a depth below foundation level at least equal to the width (B) of the foundation.

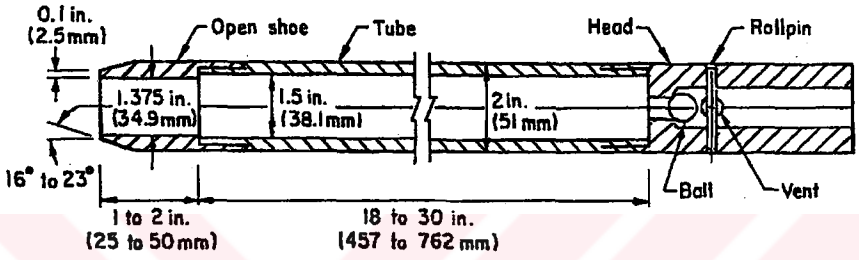


Figure 3. 1 Standard split spoon sampler (Kulhawy & Mayne, 1990).

The overall equipment and setup for SPT are shown in Figure 3.2.

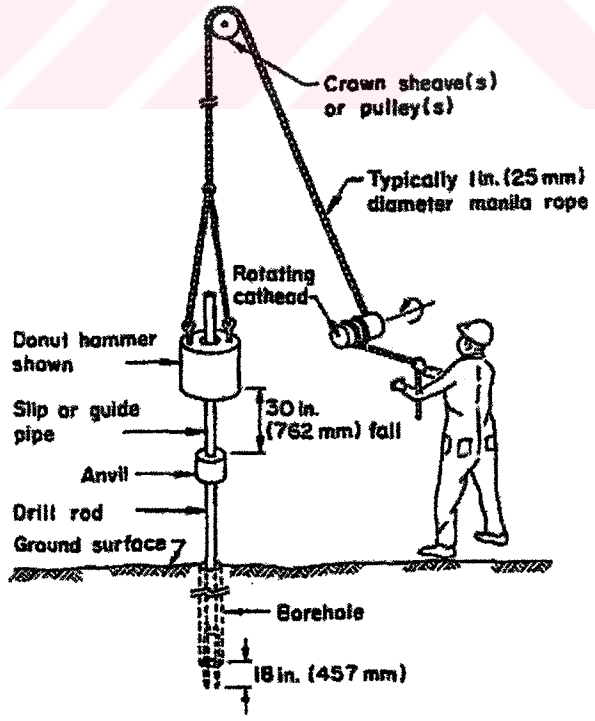


Figure 3. 2 Equipment used to perform the SPT (Kulhawy & Mayne, 1990).

3.2 Standard Penetration Test Corrections

It has been observed that N values from adjacent boreholes or from using different equipment in adjacent holes were not reproducible. The input driving energy and its dissipation around the sampler into the surrounding soils have been the principal factors for the wide range in N values. The actual input driving energy to the sampler to produce penetration has been found in the range of 30 to 100 percent.

These discrepancies appear to arise from equipment type, drive hammer configurations and actuating system, bore hole diameter, liner existence inside the split barrel sampler, overburden pressure and length of drill rod.

From the several recent studies, it has been suggested that the SPT be standardized to some energy ratio E_r which should be computed as

$$E_r = (E_a / E_{in}) \times 100 \qquad (3.1)$$

Where, E_a = Actual hammer energy to sampler and E_{in} = Input Energy.

Due to the above mentioned factors, it has been observed that there is a wide scatter in E_r and the resulting blow count N. On the other hand, it is expected that there should be a unique N for the soil at some depth. Hence, it has been suggested that the drill system-dependent E_r of Eq. (3.1) be referenced to a standard energy ratio value E_{rb}. In this way a drill rig with, say, $E_r = 45$ would, on adjustment to the standard E_{rb}, compute approximately the same N count as from a drill rig with $E_r = 70$.

Average values of the standard energy ratio E_{rb} as 55, 60 and 70 are suggested by Schmertmann, Seed et al. and Skempton, and Riggs, respectively.

The standard blow count N′ can be computed from the measured N as follows:

$$N' = C_n \times N \times \eta_1 \times \eta_2 \times \eta_3 \times \eta_4 \quad (3.2)$$

Where C_n = Adjustment factor for effective overburden pressure correction, η_1 = adjustment factor for energy correction, η_2 = adjustment factor for rod length correction, η_3 = adjustment factor for sampler correction, and η_4 = adjustment factor for bore hole diameter correction (Bowles, 1996).

Values of η_i adjustment correction factors are given in Table 3.1

Table 3. 1 Factors η_i For Eq (3.2) (After Bowles, 1996)

	Hammer for η_1				**Remarks**
	Average energy ratio E_r				
	Donut		**Safety**		
Country	**R-P**	**Trip**	**R-P**	**Trip/Auto**	R-P =Rope-pulley or cathead
United States / North America	45	-	70-80	80-100	$\eta_i = E_r/E_{rb} = E_r / 70$ For U.S. trip/auto w/E_r=80
Japan	67	78	-	-	η_i = 80/70 = 1.14
United Kingdom	-	-	50	60	
China	50	60	-	-	

Rod Length correction η_2		
Length >10 m	η_2 = 1.00	N is too high for L < 10
6-10	=0.95	
4-6	=0.85	
0-4	=0.75	

Sampler correction η_3		
Without liner	η_3 = 1.00	Base value
With liner: Dense sand, clay	=0.80	N is too high with liner
Loose sand	=0.90	

Borehole diameter correction η_4		
Hole diameter 60-120 mm	η_4=1.00	
150mm	=1.05	
200mm	=1.15	

There are a number of overburden correction equations. The following equation is the simplest of all and plots nearly the average of all the other methods (Liao & Whitman, 1986 [in Bowles, 1996]).

$$C_n = (95.76 / \sigma_o')^{1/2} \quad (3.3)$$

Where, σ_o' is in kN/m^2

Energy ratio × blow count is a constant for any soil, so

$$E_{r1} \times N_1 = E_{r2} \times N_2 \quad (3.4)$$

Or

$$N_2 = (E_{r1}/E_{r2}) \times N_1 \quad (3.5)$$

Using the relationship given by Equation (3.4) one can readily convert any energy ratio to any other base, but one do has to know the energy ratio at which the blow count was initially obtained (Bowles, 1996).

Negative excess pore water pressure sets up during driving, when the standard penetration test is carried out in very fine medium to dense sand or silty sand below the water table. Negative excess pore water pressure increases the effective stress, which in turn causes an increase in the penetration resistance. If the measured N value is greater than 15, it may be corrected using the following equation (Craig, 1992):

$$N' = 15 + \tfrac{1}{2}(N - 15) \quad (3.6)$$

Where, N' = Corrected penetration resistance, N = Measured penetration resistance.

3.3 SPT-N Correlations

The SPT-N value has been correlatively used to assess the values of some soil parameters such as unit weight γ, relative density D_r, angle of internal friction ϕ,

undrained compressive strength q_u, and the stress-strain modulus E_s. It has also been used to estimate the initial settlement and the allowable bearing capacity of foundations resting on sandy soils.

SPT-N correlations must be used cautiously. Databases used for these correlations may not be large enough, may belong to specific soils or may have been obtained from published literature in which the range of E_r may be on the order of 30 to 100 percent.

As an example chosen among numerous correlations presented in the literature are the values given in Table 3.2. In Table 2 empirical values of ϕ, D_r and γ for normally consolidated granular soils based on SPT-N under an effective overburden pressure of 100 kN/m^2 are presented.

Table 3. 2 Empirical Values for ϕ, D_r and γ of Normally Consolidated Granular Soils Based on SPT-N for an Effective Overburden Pressure of 100 kN/m^2 (After Bowles,1996)

Description	*Very loose*	*Loose*	*Medium*	*Dense*	*Very dense*
Relative density D_r	*0*	*0.15*	*0.35*	*0.65*	*0.85*
SPT N'_{70}: Fine	*1-2*	*3-6*	*7-15*	*16-30*	*?*
Medium	*2-3*	*4-7*	*8-20*	*21-40*	*>40*
Coarse	*3-6*	*5-9*	*10-25*	*26-45*	*>45*
ϕ: Fine	*26-28*	*28-30*	*30-34*	*33-38*	
Medium	*27-28*	*30-32*	*32-36*	*36-42*	*<50*
Coarse	*28-30*	*30-34*	*33-40*	*40-50*	
γ_{wet} kN/m^3	*11-16*	*14-18*	*17-20*	*17-22*	*20-23*

In this study, following SPT- N value correlations with angle of friction ϕ for sandy soils and unconfined compressive strength q_u for clayey soils have been used.

$$\phi = 0.36\, N_{70} + 27 \tag{3.7}$$

$$q_u = kN \tag{3.8}$$

The first equation is for buildings [Shioi & Fukui, 1982 (in Bowles,1996)]. The value of k in the second equation tends to be site-dependent: however, a value of k=12 has been used (Bowles, 1996).

3.4 Energy Ratio Assessment for the SPT Equipment Used in Izmir

SPT hammer used by the firms located in Izmir is Donut type and the driving mechanism is rope and pulley.

According to Table 3.1, this type of SPT equipment has an average energy ratio of $E_r = 45$ percent.

Considering a standard energy ratio of $E_{rb} = 60$ percent, the value of energy correction factor η_l becomes 45/60 = 75 percent. This value can be suggested for a general energy correction. In case of a different value of standard energy ratio, such as $E_{rb} = 70$ percent, the value of η_l can be taken as 45/70 = 64 percent.

CHAPTER FOUR
COMPUTER PROGRAMMING

4.1 Program Solution Method

In this study, a computer program has been prepared for the determination of the ultimate axial bearing capacity of a single pile based on Standard Penetration resistance.

The program has been prepared in Visual Basic 5.0 programming language. It is composed of 13 forms and one module. The program consists of 4400 rows of Visual Basic code.

The program has input and output forms. The inputs are soil profile, SPT resistance values and pile properties.

The program first processes the sub layering operation using inputs, which provides calculation of detailed bearing capacity values. After sub layering, it evaluates the parameters, which are required for bearing capacity calculation. These parameters are: effective overburden pressures, corrected SPT-N values for each sub layer, internal friction angle ϕ for sand sub layers, and undrained cohesion c_u for clay sub layers.

In the bearing capacity calculation process, user interaction is needed to choose the calculation method for each main layer.

Two calculation methods are available for sand layers. These are Meyerhof's Method and Static Formulation Method. In Meyerhof's Method, the base resistance and skin friction are correlatively calculated from corrected SPT-N values. In case of Static Formulation Method, the internal friction angle has been used for the base

resistance and skin friction calculation. The values of friction angle can either be correlatively obtained by the program from SPT-N values or can be entered interactively.

For clay layers, the base resistance is calculated using c_u with Hansen's method. Two methods for skin friction resistance calculation have been offered to user. The skin friction calculation methods for clay layers are α and β methods.

After the calculation of base and skin friction resistances, the base resistances are corrected according to two methods depending on successive layer strengths. These are Meyerhof's Method (the rapid transition method) and averaging method. Resulting values and graphs of depth versus effective overburden pressure, SPT-N, corrected SPT-N, base resistance, corrected base resistance, skin friction, and ultimate bearing capacities can be obtained.

Above mentioned topics have been presented in some detail in the following subsections.

4.1.1 Inputs

The main input document of the program is the log of boring with penetration resistances. A Geological Engineer prepares the boring log during boring and execution of SPT in the field. In the boring log, several information about the SPT, beginning and final depth of each test, SPT-N values, geological layers, color and smell of soil samples are present.

To form the soil profile for computer evaluation, the beginning and final depths of the geological layers existing in the boring log are entered in the 'Field Identification' form. In the program, the soil layers have been classified as sand or clay. By examining the soil data, soils are separated as sand or clay according to their behavior. Also the unit weights of layers and the depth of ground water table are needed for calculating the effective overburden pressures.

The SPT beginning depths and the SPT-N values are entered in the 'SPT Properties' form for performing the sub layering operation. The energy ratio of the SPT equipment should also be entered in this form.

The pile installation method has been taken into consideration as driven or bored in the program. Installation method, pile sectional property (circular or square) and diameter (or width) of the pile can be entered in the 'Pile Properties' form.

4.1.2 Layering

The geological layers in the boring log constitute the main layers in the program. In order to make detailed calculations, each main layer in the soil profile has been divided into sub layers as displayed in Figure 4.1.

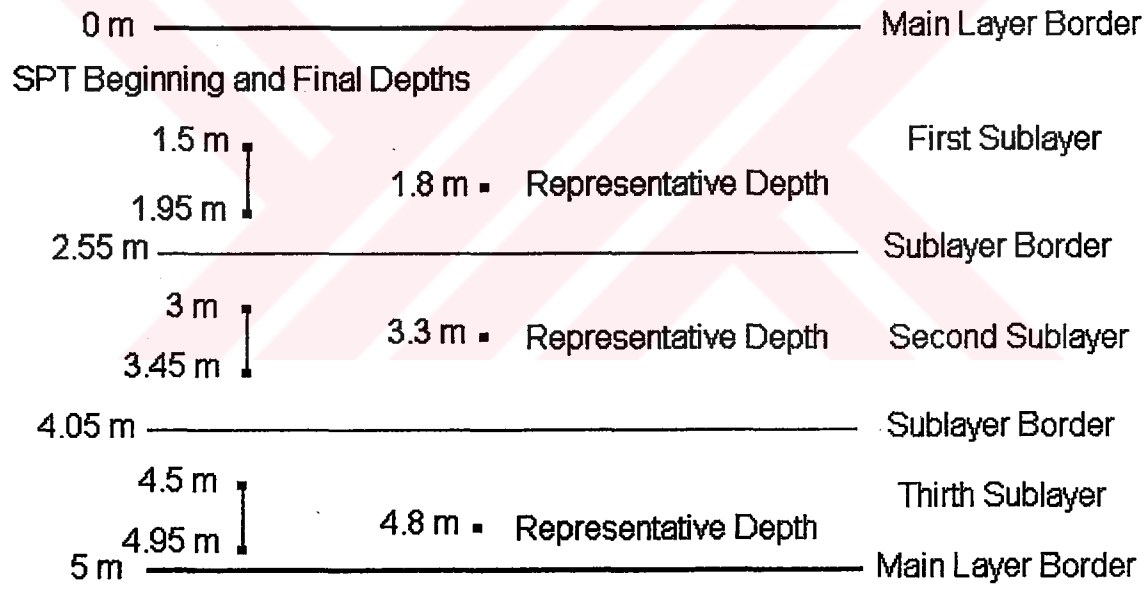


Figure 4. 1 Sub layering of the first 'Main Layer'

A representative depth has been assigned to each sub layer of a main layer. This representative depth is the depth corresponding to the middle depth of the second and third 150 mm increment of the sampler. This means that it is 150 mm above the SPT final depth. These representative depths provide the separation of a main layer into sub layers. Arithmetical averages of the two successive representative depths

constitute sub layer interfaces provided that there is no ground water table level or main layer interface in between these representative depths. Otherwise, ground water table level or main layer interface replaces the border of a sub layer.

As displayed in Figure 4.1, number of sub layers is equal to the number of SPT in a main layer. In algorithm of the program, each sub layer has been used as a segment to which the parameters are assigned, in which formulations are executed, and results are obtained. Thus, the formulations prepared for a segment are executed for all sub layers in the soil profile to obtain detailed results by making use of the calculation capacity of the computer. The 'Grid Object' in Visual Basic has been used to represent sub layers (segments) easily. Each sub layer has produced a row in the grid object. Sand layers have been prepared as green and clay layers have been prepared as orange to get a visual comfort as shown in Figure 4.2.

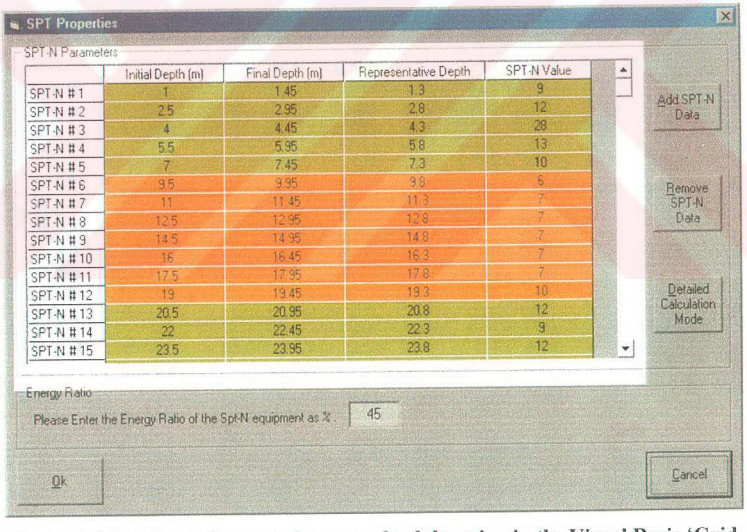


Figure 4. 2 Sand and clay main layers and sub layering in the Visual Basic 'Grid Object'

As mentioned in the preceding paragraph, the main layers have been divided into sub layers in order to make detailed calculations. If the user needs more detailed calculation, by pressing 'Detailed Calculation Mode' button on the 'SPT Properties' form, the sub layers will be divided into 25 cm thick fine sub layers. For each of these fine sub layers the base resistance and skin friction calculation is performed which makes the resulting graphs more detailed and smoother. SPT value of a sub layer is assigned to all of its fine sub layers. For this reason in the detailed calculation mode the initial and final depths of SPT are not available. This operation is shown in Figure 4.3.

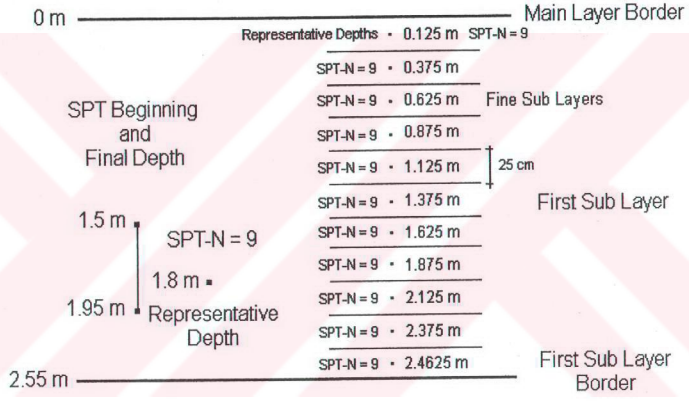


Figure 4. 3 The division of the first sub layer into fine sub layers in 'Detailed Calculation Mode'

4.1.3 Calculation Procedure

Calculations are performed at the representative depth in a sub layer. The calculated parameters of the sub layers have been assigned to index integers. So a parameter can later be easily called back with its index number. For example the overburden pressure of the 10^{th} sub layer has been called as SigZero(10).

For each sub layer, effective overburden pressure σ_o' at the representative depth is calculated. An effective overburden pressure calculation algorithm has been prepared and executed at representative depths. The results have been assigned to effective overburden pressure index integer. The overburden pressure corrections for SPT-N values are performed without user interaction. To process the overburden pressure correction C_n values are calculated by the Equation (3.3). Calculated C_n values are assigned as new parameters to the sub layers.

The overburden pressure correction on SPT-N numbers for each sub layer are performed by the following formula:

$$N' = C_n \times N \tag{4.1}$$

Where, C_n is adjustment factor for effective overburden pressure, and N is penetration resistance.

In case of very fine sand or silty sand below the water table and SPT-N > 15, negative pore water pressure correction may be applied to N' values by Equation (3.6).

At this stage all representative depths possess an effective overburden pressure and a corrected SPT-N value.

To obtain ϕ' and c_u parameters of the sub layers, correlation equations (Equation 3.7 and Equation 3.8) are used. But these equations are valid for 70 percent energy ratios. The energy ratio for which the correlation is valid is used for energy correction. On the 'SPT Properties' form, the energy ratio of the SPT equipment is entered by the user (Figure 4.2).

ϕ' and c_u parameters of sub layers are also assigned to index integers for bearing capacity calculation.

The above explained calculation procedures can be seen in Figure 4.4 in their order of appearance.

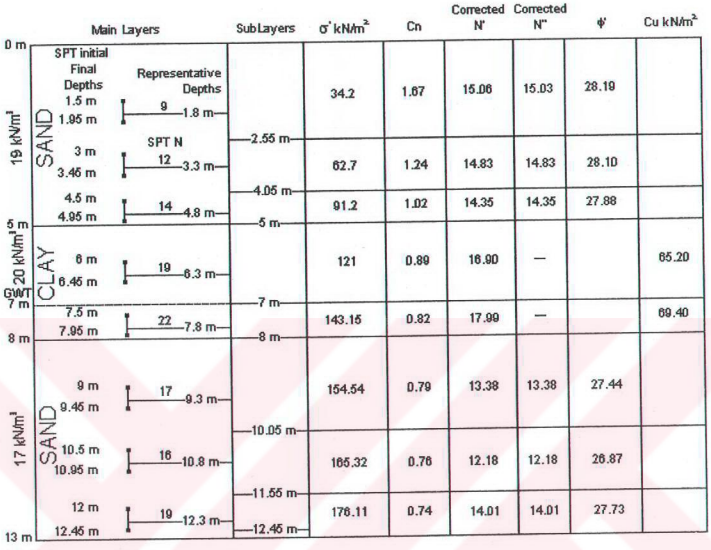


Figure 4. 4 The layering and calculation sequence of the profile

The base area of the pile and the skin area of each sub layer are calculated depending on the pile section and diameter.

The user, selecting the calculation method in the 'Bearing Capacity Calculation' form, which is shown in Figure 4.5, can activate the bearing capacity calculation for the main layers. The method will be applied to each sub layer of the main layer. For a sand layer only the buttons for sand layer calculation methods are available, and for a clay layer only the buttons for two skin friction calculation methods are available.

At this point, user can enter any laboratory values for ϕ' and c_u instead of the values derived from correlations. By clicking any cell of ϕ' and c_u, a new form will be displayed for inputting the new value.

For sand layers, two calculation methods (Meyerhof's Method and Static Formula) are available in the program. Applications of these two methods are presented below.

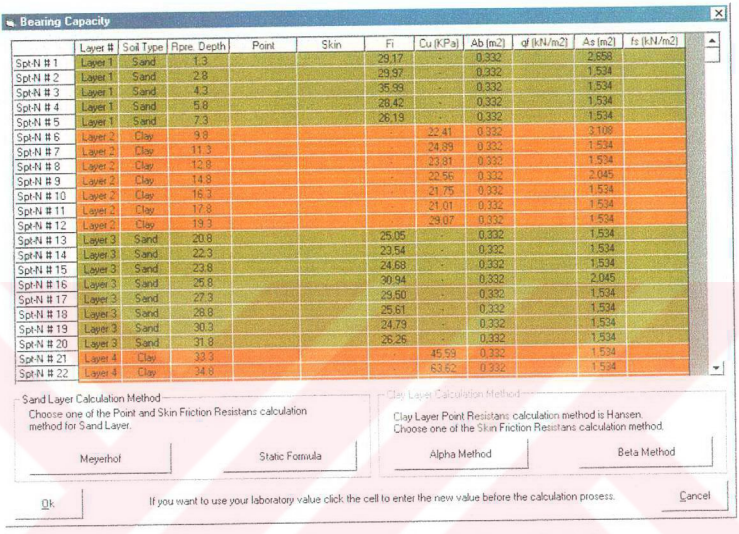


Figure 4. 5 'Bearing Capacity' form before calculation

i) Meyerhof's Method

In this method the point and skin resistances of the pile have been calculated by Meyerhof's correlation equations (Equation 2.6 And Equation 2.7) with corrected SPT-N value of each sub layer.

SPT energy ratio for Meyerhof's equations is proposed as 55 percent (Bowles, 1996). Thus for an energy ratio of 45 percent, $\eta_1 = 45 / 55 = 82$ percent.

To obtain the q_f value, the energy correction for E_{rb} = %55 is processed and Equation (2.6) is applied.

Because of the critical depth limitations in the sand layers, the q_f value has been limited by 400 N_{55}'. That means D_b / D ratio is limited by 10.

The skin friction resistance of each sub layer is calculated by the correlation Equation (2.7).

In skin friction resistance calculations for this method (and also for the other methods), the skin area is evaluated until the representative depth for the sub layer, in which the pile base exists.

ii) Static Formula

In this method the point and skin resistance of the pile is calculated with static formula by using internal friction angle.

The ultimate bearing capacity at base level is determined by Equation (2.3).

Because of the bearing capacity limitations of sand layers, the critical depth value is determined from the correlation between the ϕ' and z_c/D (Figure 2.3). z_c is the critical depth and D is the pile diameter. The graph has been reestablished in Microsoft Excel and a polynomial trend line of order 6 has been derived as shown in Figure F.1. This trend line has been used as a function in Visual Basic to get the z_c/D value for any ϕ' value. Details of this trendline and of similar trendlines that will be mentioned about in the rest of this chapter are presented in Appendix F of this thesis.

In order to calculate the critical depth value of a main layer, the ϕ' values of all sub layers has been averaged to use in the critical depth equation. In using Figure 2.3, ϕ'_1 values before pile installation are revised as $\phi' = ¾\ \phi'_1 + 10$ for driven piles and as $\phi' = \phi'_1 - 3$ for bored piles as suggested in Paulos & Davis, 1980.

Below the critical depth, σ_o' term is kept constant. To do so, effective overburden pressure calculation algorithm is processed for the critical depth, and the result is

used as the σ_o' term in the bearing capacity formulation for the depths exceeding the critical depth.

The value of skin friction for each sub layer over the length of pile embedded in the sand has been calculated with Equation (2.5).

$K_s \tan \phi'$ term has been taken from the correlation between ϕ' and K_s as displayed in Figure 2.2. Curves have been reestablished in Microsoft Excel and a polynomial trend line of order 6 has been derived as shown in Figures F.2 and F.3. These trend lines have been used as a function in Visual Basic to get the $K_s \tan \phi'$ value for driven piles and K_s value for bored piles for any ϕ' value. In using Figure 2.2(a), ϕ_1' values are revised as $\phi' = ¾\ \phi'_1 + 10$ for driven piles as suggested in Paulos and Davis, 1980.

The same σ_o' values as for q_f with critical depth limitation are used in determining f_s.

For clay layers the base resistance has been calculated with the Equation (2.8), Application of the two methods for skin friction calculation (α-Method and β-Method) is summarized below.

i) Alpha Method

The skin friction for each sub layer has been calculated with the Equation (2.9). Adhesion factor α values are determined using Figure 2.4. This graph has been converted to a mathematical line equation, which has three parts, and applied in Visual Basic.

ii) Beta Method

By selecting this method, a new form is presented on the screen. In this form, the plasticity index and over consolidation ratio of the clay layer is required. If the user enters a plasticity index I_p value, this entered value is used for the clay layer. Otherwise, the I_p term is calculated by the program with the following correlation for normally consolidated clays given by Skempton (in Craig, 1992).

$$c_u / \sigma_o' = 0.11 + 0.0037\, I_p \qquad (4.2)$$

Where; c_u is the undrained cohesion of the clay sub layer, σ_o' is the effective overburden pressure at the representative depth of the sub layer.

The relation in Figure 4.6 between I_p and ϕ' for remolded clays has been used, in order to determine ϕ' and then the skin friction with Equation (2.10)

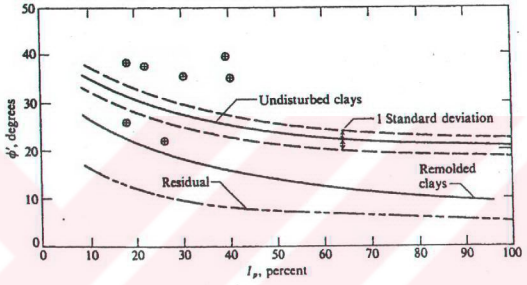


Figure 4. 6 Correlation between I_p and ϕ' for remolded clays (After Bowles, 1996)

The graph has been reestablished in Microsoft Excel and a polynomial trend line of order 6 has been derived as shown in Figure F.4. These trend lines have been used as a function in Visual Basic.

For the over consolidation frame of the window, two radio buttons have been located. The user can enter the OCR value by selecting the 'OCR radio' button. The value will be used for all of the sub layers in the clay layer.

If the user doesn't know the OCR value and has a doubt that the clay layer is over consolidated, the second radio button should be clicked. This section requires entering an I_p term. By using the c_u value of the sub layer, the overburden pressure is calculated at the representative depth with Equation (4.2), and compared with the actual effective overburden pressure, which has been calculated at the representative

depth. If the value of the ratio of the effective overburden pressure derived from the correlation equation to the actual effective overburden pressure exceeds 1.1, the sub layer is assumed to be over consolidated and the calculated over consolidation ratio is used in Equation (4.6).

K_s, the coefficient of lateral earth pressure for normally consolidated and over consolidated clays has been derived from the following equations, respectively.

$$K_s = (1 - \sin \phi') \tag{4.3}$$

$$K_s = (1 - \sin \phi') \times \sqrt{OCR} \tag{4.4}$$

By inserting the above expressions for K_s in the skin friction formula the following formulations are obtained.

For normally clays

$$f_s = (1 - \sin \phi') \times \sigma_o' \times \tan \phi' \tag{4.5}$$

For over consolidated clays

$$f_s = (1 - \sin \phi') \times \sigma_o' \times \tan \phi' \times \sqrt{OCR} \tag{4.6}$$

4.1.4. Base Resistance Corrections

The base resistance correction has been applied with two methods. The purpose of this correction is to soften the base resistance values by passing between the layers, which have different base resistances.

In the first method, (Meyerhof's Method) Equation (2.12) has been applied to the base resistances. This equation is valid for a 10 D distance inside a dense layer from a weak layer, either at the top or at the bottom of the dense sand layer.

In the second method, the calculated sub layer base resistances in the vicinity of the pile base have been averaged. Averaging has been done for a depth of three pile diameters below the pile base and five pile diameters above the pile base. But, user may change these values before base resistance correction.

4.1.5 Plotting of Graphs

Calculation results are presented with several graphs. To plot the graphs, a new form is displayed. Soil profile is shown fixed for all the plots at the left side of the screen. Effective overburden pressure, raw and corrected SPT-N Values, and bearing capacity versus depth graphs can be plotted by selecting the desired command from the menu.

In the bearing capacity versus depth graph, the base resistance, corrected base resistances according to the two base resistance correction methods, skin friction, and ultimate bearing capacity graphs can be plotted by selecting the corresponding boxes in the same graph.

CHAPTER Five
APPLICATIONS

5.1 Problem Introduction

In order to test the algorithms of the computer program, thirty-two different soil and penetration resistance profiles from eight different soil investigation locations surrounding Izmir Bay have been analyzed. Soil data have been obtained from Faculty archives. Locations of the soil investigation areas are shown in Figure 5.1.

Location Number	Region	Project Name	Drilling Company
1	Karşıyaka	Vilayetler Hizmet Birliği İzmir Karşıyaka Eğitim Tesisleri İnşaatı Zemin Etüdü ve Temel Mühendisliği Değerlendirme Raporu	Zetaş A.Ş.
2	Karşıyaka	KSK Plaza ve Spor Tesisleri İnşaatı Sondaj Raporu	Ege Temel Sondajcılık
3	Turan	Turan Askeriyesi Zemin Etüd Raporu	Ege Temel Sondajcılık
4	Bayraklı	İzmir Piyale Trafo Merkezi İnşaatı Sondaj Raporu	Ege Temel Sondajcılık
5	Mersinli	İzmir İl Özel İdare Müdürlüğü Hizmet Binası İnşaatı Zemin Etüd Raporu	Alkım Ltd. Şti.
6	Alsancak	Yaşar Eğitim ve Kültür Vakfı İzmir Müzesi Restorasyon İnşaatı Sondaj Raporu	Ege Temel Sondajcılık
7	Alsancak	Alsancak TMO ve TCDD Binaları İnşaatı Sondaj raporu	Ege Temel Sondajcılık
8	Çankaya	Tarım Kredi Kooperatifleri Birliği İzmir Bölge Müdürlüğü Hizmet Binası İnşaatı Sondaj Raporu	İzmir Temel Sondajcılık

Figure 5. 1 Locations of the soil investigation areas

Computer program has been run with the SPT data at each boring location. Malfunctions, which have been met in certain conditions, have been determined and corrected during these applications. In order to determine the malfunctions, each operation performed by the program has been observed step by step. The presentation of the output graphs has also been improved at this stage.

Another outcome of application study has been the evaluation of ultimate bearing capacity of single bored piles at subject matter locations. The number of locations is obviously not enough for a detailed regional modeling of ultimate bearing capacity in İzmir. However, the distribution of the soil investigation locations has been selected so that a representative range of ultimate bearing capacity could be evaluated.

The author had the chance of taking part the application of the standard penetration tests in location (5). GEDİK A.Ş., a drilling company, has performed the borings and the standard penetration tests and the resulting report has been approved by Dokuz Eylül University. At this soil investigation area, SPT tests have been carried out at eight different locations, and the entire test results have been evaluated in this study.

5.2 Solutions

Computer program has been run for a bored pile having 65 cm diameter.

In the ultimate bearing capacity calculation process, static formulation for sand layers, and alpha method for clay layers have been used. The base resistance correction with average method has been performed for a depth interval of 3 pile diameters under the base, and 5 pile diameters above the base.

The unit weights of sand layers have been correlatively determined from Standard Penetration Resistance (SPT-N) values. The unit weights of clay layers have been either obtained from laboratory test results or calculated from water contents by assuming that clay is saturated.

The resulting graphs have been presented in Appendix D. In this Appendix section, boring profile, SPT-N and corrected SPT-N, minimum corrected base resistance, skin friction, and minimum ultimate bearing capacity versus depth graphs have been presented for entire borings.

5.3 Evaluation of the Results

For Location (1) (Bostanlı), the ultimate bearing capacity has been found to increase stepwise at 13 m depth in the upper sand layer. The below lying clay layer met in boring 1 and boring 2 has been evaluated as inappropriate for a pile base due to probable excess settlement. The sand layer located approximately at a depth of 20 m has been found to provide a good bearing capacity, which increases with depth.

Bearing capacity of a single pile for Location (2) (Karşıyaka/Donanmacı) has been found to vary from boring to boring. But in general, it can be observed that the probable sand layer at a depth of 30 m constitutes a good medium for base resistance.

At Location (3) (Turan), the same pattern of layering has been observed for the two borings. Ultimate bearing capacity increases stepwise starting with the sand layer below 13 m depth, and increases linearly in the clay layer located below the sand layer.

For Location (4) (Bayraklı), the ultimate bearing capacity starts to increase at 9 m depth, and has a stepwise increase at approximately 15 m depth. But after that depth, relatively low bearing capacity has been observed. Bearing capacity starts to increase at 21 m depth.

Locations (5), (6), (7) and (8) lie on Melez Creek sediments. The distributions of the layers in the soil profile have been found to be more or less similar. A generalization has been made. The sand layer located approximately between 13 m and 20 m depths for Locations (6) and (7) has been observed to provide a good medium for pile base. This layer also exists in Location (8), but between 19 m and 22 m depths.

No errors and anomalies have been met during the applications, and the results have been proved that computer program had worked correctly.

During the application phase, it has been realized that, new bearing capacity calculation formulas for sand or clay layers could be added easily to the computer program.

The program also constitutes a base for lateral ultimate bearing capacity calculation, analysis of liquefaction, and negative skin friction calculation.

This computer program has the capability of calculating the ultimate bearing capacity for a single pile in a short time. Thus, a detailed regional modeling of ultimate bearing capacity can be made quickly, if there exists enough, reliable SPT and soil profile data.

CHAPTER SIX
CONCLUSIONS

The computer program which has been prepared for the estimation of the ultimate axial bearing capacity of a single pile principally based on SPT data has been written in Visual Basic Programming language, having thirteen forms and one module and consisting of 4400 rows of Visual Basic code.

This program needs boring log data with SPT-N resistances. Shear strength parameters can also be interactively inputted, if required.

Basically, Meyerhof's Method and Static Formula have been used with the required relations present in Paulos & Davis, 1980. Base resistance corrections have been included for the case of rapid transition in successive layer strengths. Stepwise ultimate bearing capacity calculation with fine analysis and graphic presentation of the results have been provided.

Validity of the results obtained with the program has been verified with hand calculation.

Thirty-two boring data from eight different locations surrounding the Izmir Bay have been executed with the computer program. The activity of the program has been observed during the applications. No errors and anomalies have been met during these applications, and the results have been proved that the program had worked correctly.

As a by product of the applications, it can be concluded that the piles located inside the sediments surrounding the Izmir Bay with moderate lengths provide their carrying capacity both form friction and end bearing. Especially, the dense sand layer

present in Melez Creek sediments on the South East of Izmir Bay with about 4 to 8 m thickness, starting from a depth of approximately 13 m to 18 m below the ground surface depending on its location has been found meaningful to mention about which seems to supply a good bearing for the piles, provided that the settlement of the below lying clay layers are acceptable for a special application.

To have a professional usage of the computer program, the project should be developed. Bearing capacity calculation details suggested by other investigators should be included in the program. Group effect, negative skin friction, settlement, lateral load carrying capacity, and liquefaction analysis could be added to the computer program. The program has been prepared in such a way that additions can quite easily be done.

REFERENCES

Craig, R. F.(1992). Soil Mechanics. London: ELBS with Chapman & Hall.

Bowles, E. J. (1996). Foundation Analysis and Design. New York: McGraw & Hill Companies Inc.

Meyerhof, G.G. (1976). Bearing Capacity and Settlement of Pile Foundations, Proceedings ASCE, 102, No. GT3, 195-228.

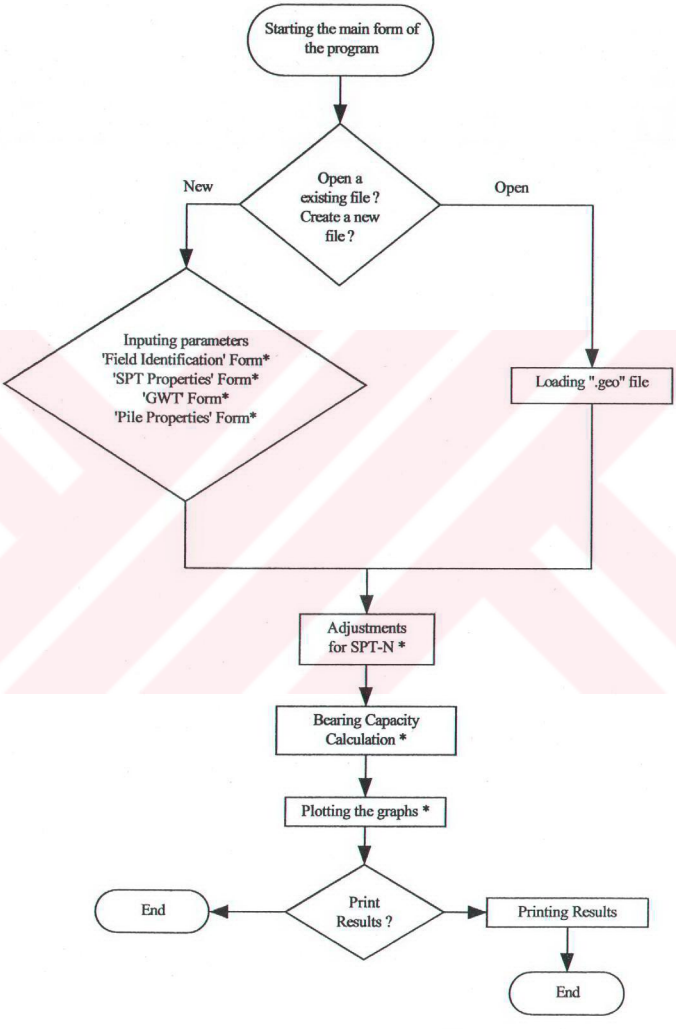
Poulos, H. G., & Davis, E. H. (1980). Pile Foundation Analysis and Design. New York: John Wiley and Sons.

ASCE (1993), Design of Pile Foundations, Technical Engineering and Design Guides as adapted from the U.S. Army Corps of Engineers, No:1.

Kulhawy, S.H., & Mayne, P.W. (1990). Manuel on Estimating Soil Properties for Foundation Design. New York: Electric Power Research Institute, Inc.

APPENDIXES

APPENDIX A Flow Charts of the Computer Program



* Flowcharts of these forms are given in the following pages.

Figure A. 1 The Flow Chart of the 'Main Form'

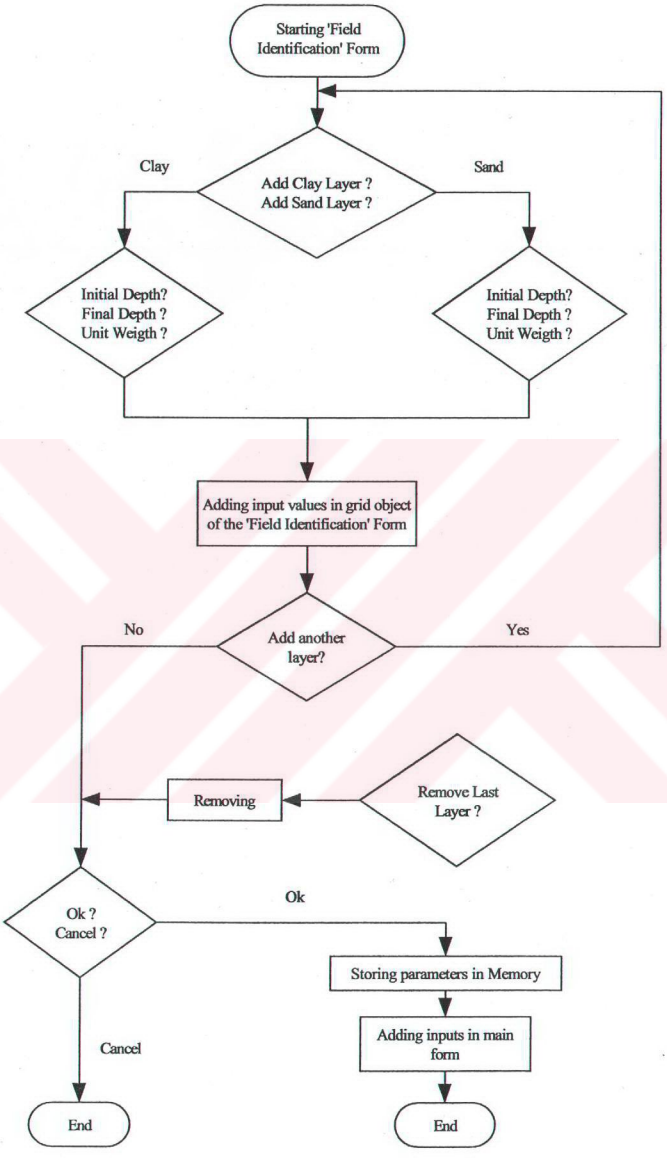


Figure A. 2 Flow Chart of the 'Field Identification' Form

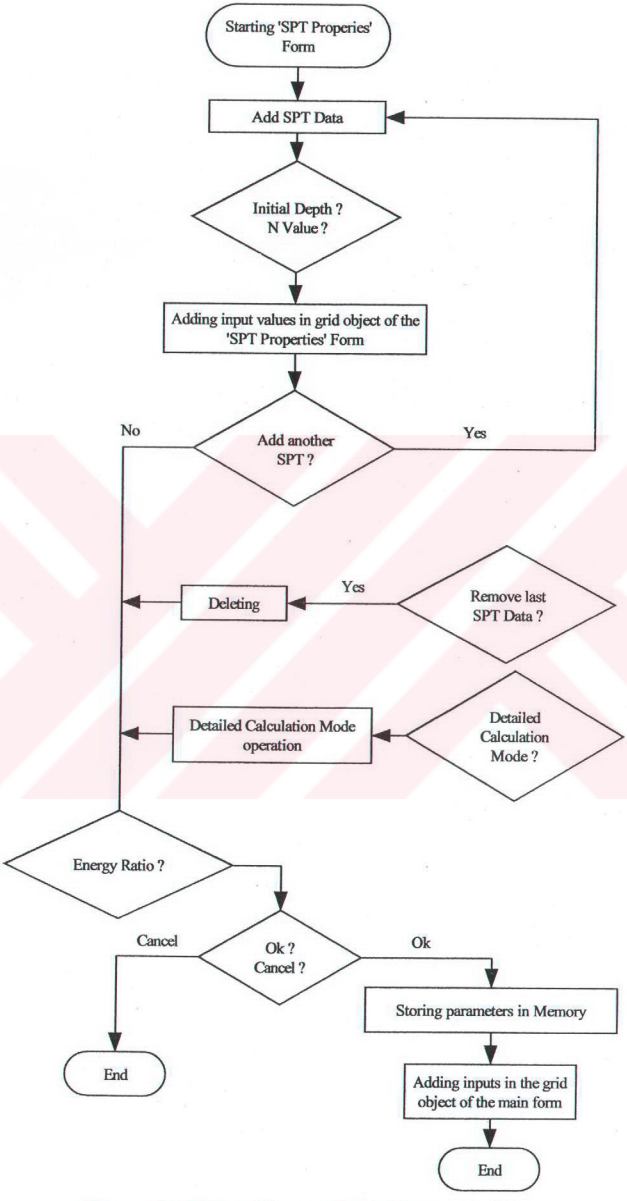


Figure A. 3 Flow Chart of 'SPT Properties' Form

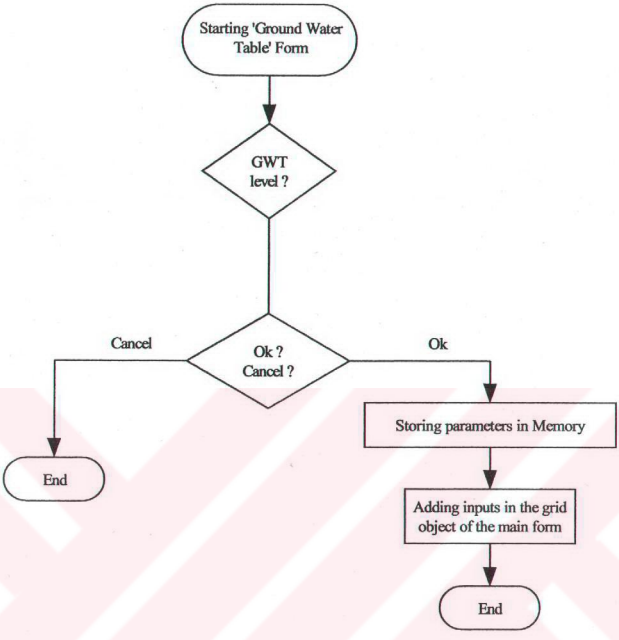


Figure A. 4 Flow Chart of the 'Ground Water Table' Form

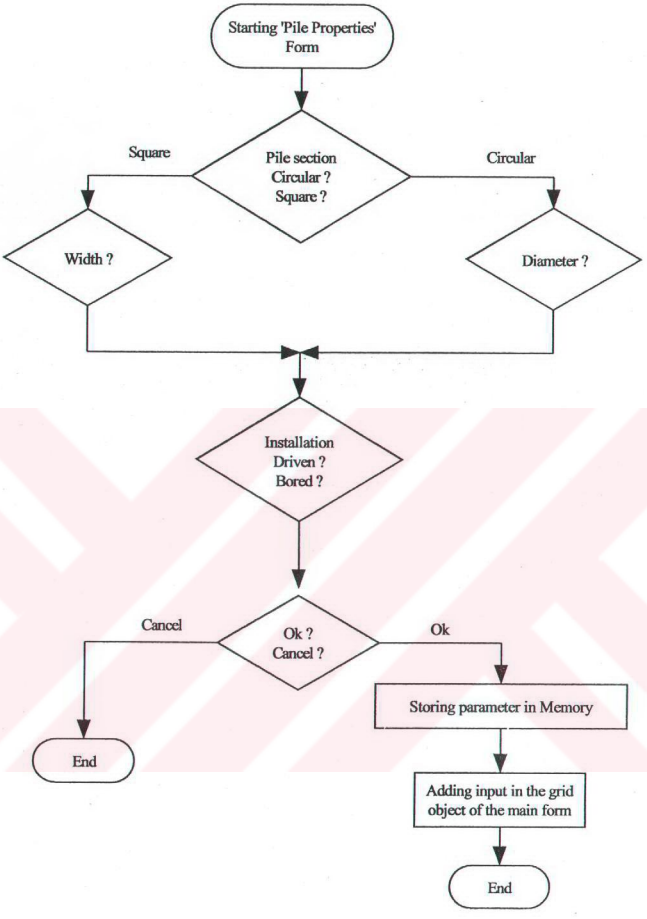


Figure A. 5 Flow Chart of the 'Pile Properties' Form

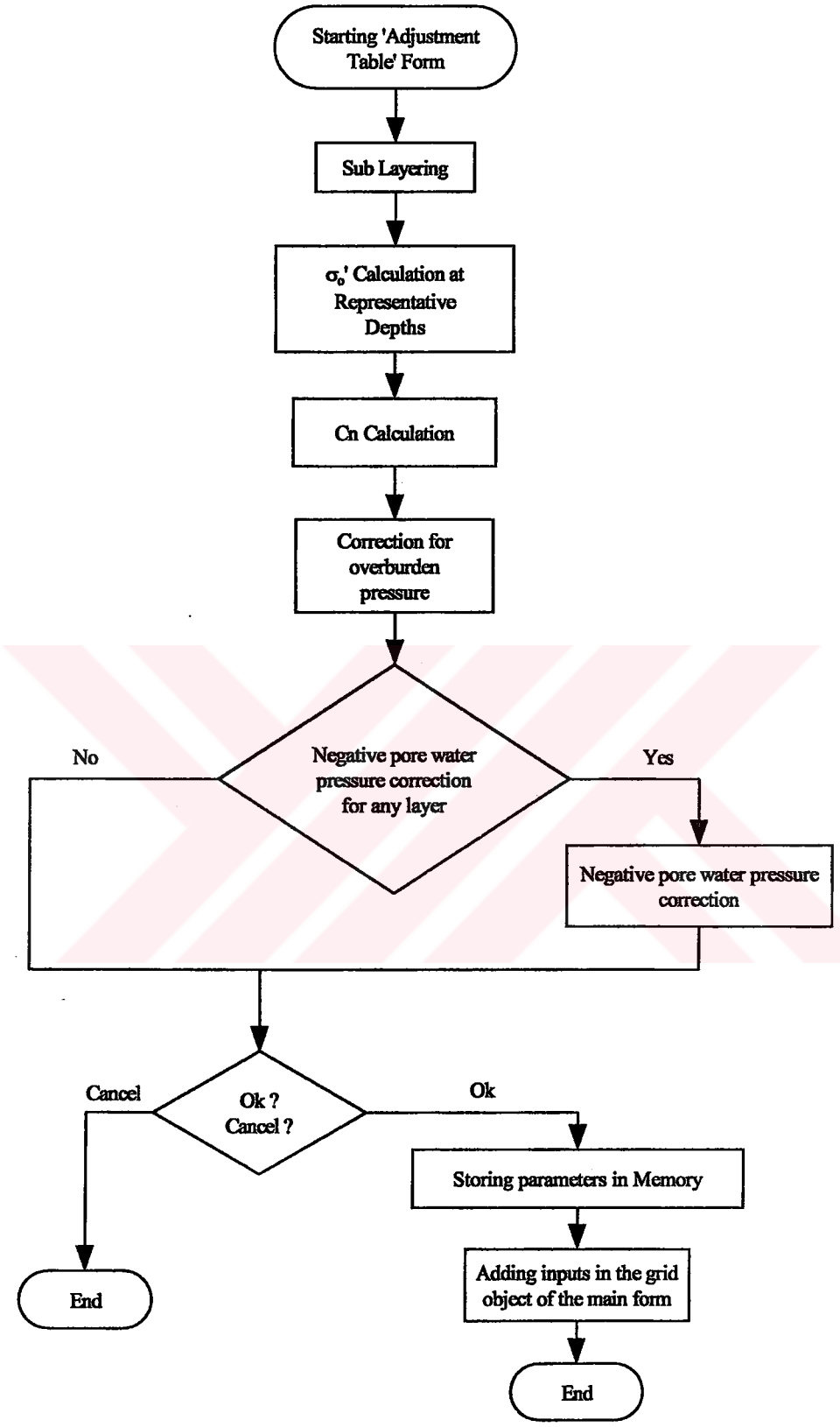


Figure A. 6 Flow Chart of the 'Adjustment Table' Form

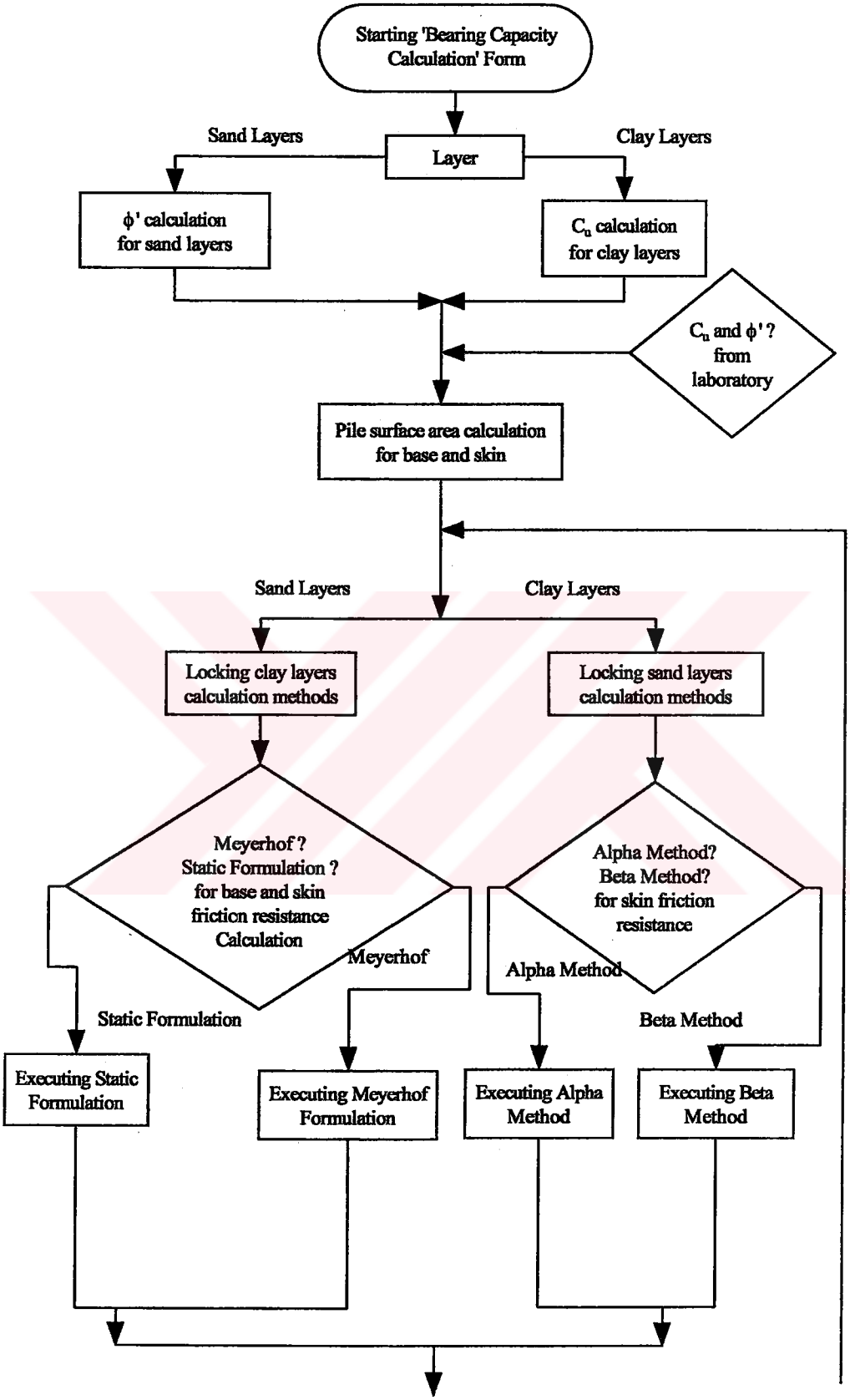


Figure A. 7 Flow Chart of 'Bearing Capacity Calculation' Form

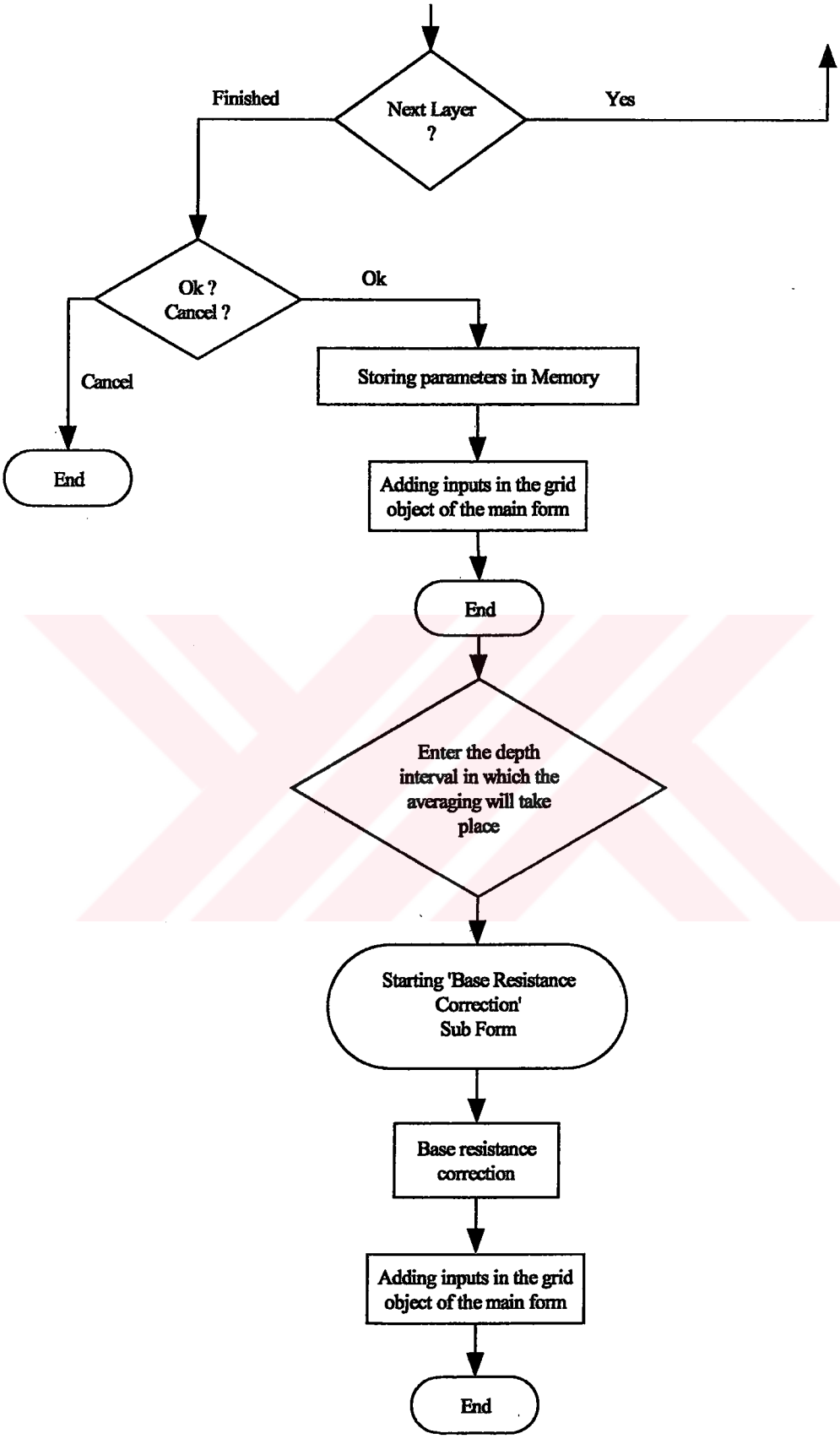


Figure A. 7 Flow Chart of 'Bearing Capacity Calculation' Form (Continued)

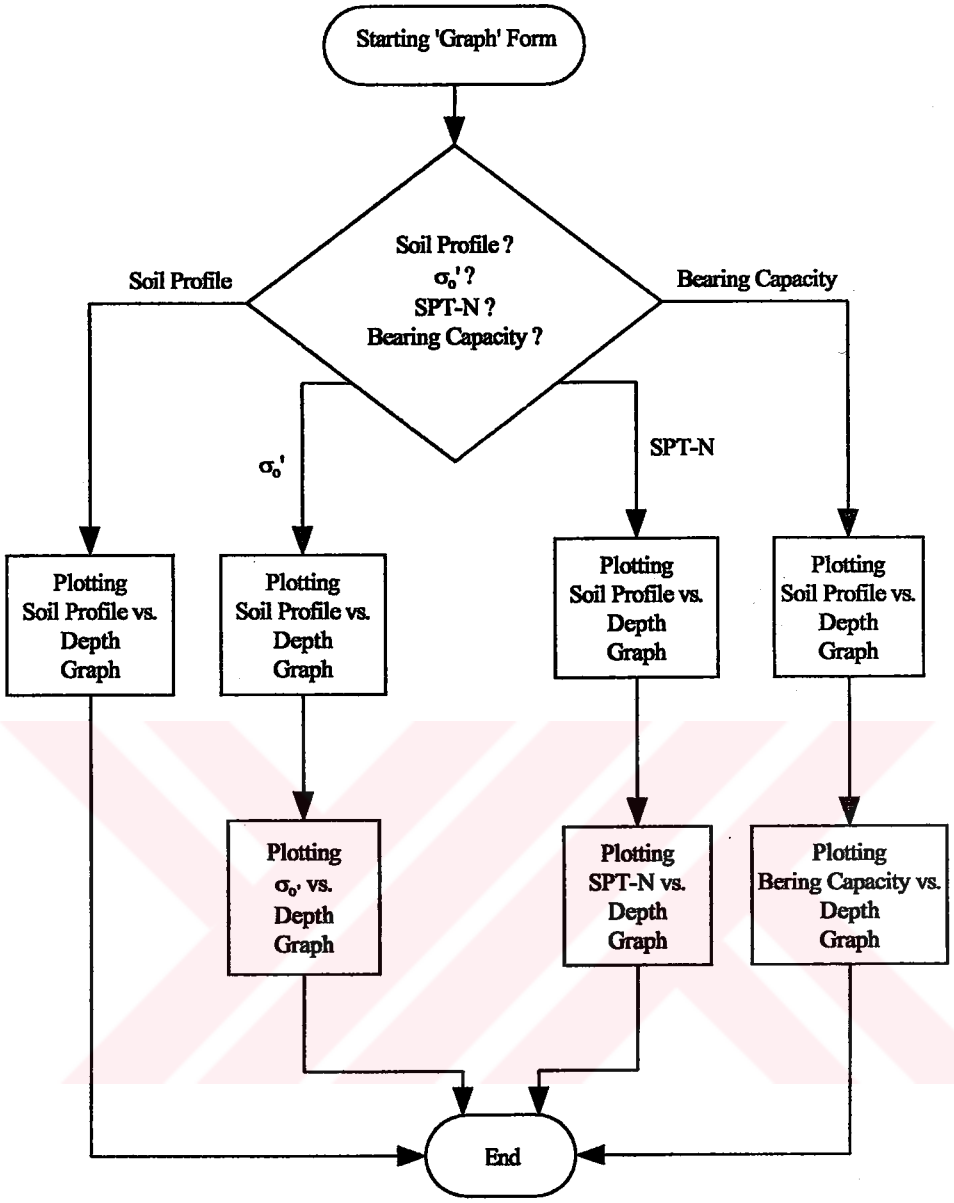


Figure A. 8 Flow Chart of the 'Graph' Form

APPENDIX B

Code of The Computer Program

APPENDIX B Code of the Computer Program

```
Dim SprdSht1Fnctns As Boolean
Dim SprdSht1TxtBxIndx As Integer
Dim SprdSht1TxtLngth() As Integer
Dim SprdSht1LstBxEntry() As Integer
Dim SprdSht1EntryLsts() As String
Dim SprdSht1TmrStrt As Integer
Dim SprdSht1RwsInfnty As Boolean
Dim SprdSht1Frml() As String
Dim SprdSht1FrmlTrmOtOfGrd() As Single
Dim SprdSht1FrmlRsltWrtnOtOfGrd As Boolean
Dim SprdSht1FrmlOtptTyp() As String

Private Sub Check1_Click(Index As Integer)
  Select Case Index
      Case 0
        If Check1(i).Value = 0 Then
          For i = 1 To UBound(Clintrfc())
            If Clintrfc(i) < Slintrfc(1) Then
              Adjustment.SprdSht1.TextMatrix(i, 7) = ""
            End If
          Next
        Else
          For i = 1 To UBound(Clintrfc())
            If Clintrfc(i) < Slintrfc(1) Then

              If N(i) > 15 Then
                N2(i) = 15 + 0.5 * (N(i) - 15)
              Else
                N2(i) = N(i)
              End If
                Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
            End If
          Next
        End If
      Case 1
      If Check1(1).Value = 0 Then
        For i = 1 To UBound(Clintrfc())
          If Slintrfc(1) < Clintrfc(i) And Clintrfc(i) < Slintrfc(2) Then
            Adjustment.SprdSht1.TextMatrix(i, 7) = ""
          End If
        Next
      Else
        For i = 1 To UBound(Clintrfc())
          If Slintrfc(1) < Clintrfc(i) And Clintrfc(i) < Slintrfc(2) Then
            If N(i) > 15 Then
              N2(i) = 15 + 0.5 * (N(i) - 15)
            Else
              N2(i) = N(i)
            End If
              Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
          End If
        Next
      End If
      Case 2
      If Check1(2).Value = 0 Then
        For i = 1 To UBound(Clintrfc())
          If Slintrfc(2) < Clintrfc(i) And Clintrfc(i) < Slintrfc(3) Then
            Adjustment.SprdSht1.TextMatrix(i, 7) = ""
          End If
        Next
      Else
        For i = 1 To UBound(Clintrfc())
          If Slintrfc(2) < Clintrfc(i) And Clintrfc(i) < Slintrfc(3) Then
            If N(i) > 15 Then
              N2(i) = 15 + 0.5 * (N(i) - 15)
            Else
              N2(i) = N(i)
            End If
              Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
          End If
        Next
      End If
      Case 3
      If Check1(3).Value = 0 Then
        For i = 1 To UBound(Clintrfc())
          If Slintrfc(3) < Clintrfc(i) And Clintrfc(i) < Slintrfc(4) Then
            Adjustment.SprdSht1.TextMatrix(i, 7) = ""
          End If
        Next
      Else
        For i = 1 To UBound(Clintrfc())
          If Slintrfc(3) < Clintrfc(i) And Clintrfc(i) < Slintrfc(4) Then
            If N(i) > 15 Then
              N2(i) = 15 + 0.5 * (N(i) - 15)
            Else
              N2(i) = N(i)
            End If
              Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
          End If
        Next
      End If
      Case 4
      If Check1(4).Value = 0 Then
        For i = 1 To UBound(Clintrfc())
          If Slintrfc(4) < Clintrfc(i) And Clintrfc(i) < Slintrfc(5) Then
            Adjustment.SprdSht1.TextMatrix(i, 7) = ""
          End If
        Next
```

```
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(4) < Clintrfc(i) And Clintrfc(i) < Slintrfc(5) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
        Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If
Case 5
If Check1(5).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(5) < Clintrfc(i) And Clintrfc(i) < Slintrfc(6) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(5) < Clintrfc(i) And Clintrfc(i) < Slintrfc(6) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
        Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If
Case 6
If Check1(6).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(6) < Clintrfc(i) And Clintrfc(i) < Slintrfc(7) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(6) < Clintrfc(i) And Clintrfc(i) < Slintrfc(7) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
        Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If
```

```
Case 7
If Check1(7).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(7) < Clintrfc(i) And Clintrfc(i) < Slintrfc(8) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(7) < Clintrfc(i) And Clintrfc(i) < Slintrfc(8) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
        Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If
Case 8
If Check1(8).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(8) < Clintrfc(i) And Clintrfc(i) < Slintrfc(9) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(8) < Clintrfc(i) And Clintrfc(i) < Slintrfc(9) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
        Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If
Case 9
If Check1(9).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(9) < Clintrfc(i) And Clintrfc(i) < Slintrfc(10) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(9) < Clintrfc(i) And Clintrfc(i) < Slintrfc(10) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
```

```
      Else
        N2(i) = N(i)
      End If
        Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If
Case 10
If Check1(10).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(10) < Clintrfc(i) And Clintrfc(i) < Slintrfc(11) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(10) < Clintrfc(i) And Clintrfc(i) < Slintrfc(11) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
        Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If
Case 11
If Check1(11).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(11) < Clintrfc(i) And Clintrfc(i) < Slintrfc(12) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(11) < Clintrfc(i) And Clintrfc(i) < Slintrfc(12) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
        Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If
Case 12
If Check1(12).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(12) < Clintrfc(i) And Clintrfc(i) < Slintrfc(13) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(12) < Clintrfc(i) And Clintrfc(i) < Slintrfc(13) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
        Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If
Case 13
If Check1(13).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(13) < Clintrfc(i) And Clintrfc(i) < Slintrfc(14) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(13) < Clintrfc(i) And Clintrfc(i) < Slintrfc(14) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
        Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If
Case 14
If Check1(14).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(14) < Clintrfc(i) And Clintrfc(i) < Slintrfc(15) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(14) < Clintrfc(i) And Clintrfc(i) < Slintrfc(15) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
        Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
```

```
  Next
End If
Case 15
If Check1(15).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(15) < Clintrfc(i) And Clintrfc(i) < Slintrfc(16) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(15) < Clintrfc(i) And Clintrfc(i) < Slintrfc(16) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
      Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If
Case 16
If Check1(16).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(16) < Clintrfc(i) And Clintrfc(i) < Slintrfc(17) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(16) < Clintrfc(i) And Clintrfc(i) < Slintrfc(17) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
      Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If
Case 17
If Check1(17).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(17) < Clintrfc(i) And Clintrfc(i) < Slintrfc(18) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(17) < Clintrfc(i) And Clintrfc(i) < Slintrfc(18) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
      Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If
Case 18
If Check1(18).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(18) < Clintrfc(i) And Clintrfc(i) < Slintrfc(19) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(18) < Clintrfc(i) And Clintrfc(i) < Slintrfc(19) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
      Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If
Case 19
If Check1(18).Value = 0 Then
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(19) < Clintrfc(i) And Clintrfc(i) < Slintrfc(20) Then
      Adjustment.SprdSht1.TextMatrix(i, 7) = ""
    End If
  Next
Else
  For i = 1 To UBound(Clintrfc())
    If Slintrfc(19).< Clintrfc(i) And Clintrfc(i) < Slintrfc(20) Then
      If N(i) > 15 Then
        N2(i) = 15 + 0.5 * (N(i) - 15)
      Else
        N2(i) = N(i)
      End If
      Adjustment.SprdSht1.TextMatrix(i, 7) = Format(N2(i), "###0.00")
    End If
  Next
End If

End Select
End Sub
```

```
Private Sub Command2_Click()
    Unload Me
End Sub

Private Sub Command3_Click()
If SprdSht1.TextMatrix(1, 2) = "" Then
Else
     ReDim CrrctdSPTN(1 To UBound(Clintrfc))
      For i = 1 To UBound(Clintrfc())
         If SprdSht1.TextMatrix(i, 7) = "" Then
           CrrctdSPTN(i) = N(i)
         Else
           CrrctdSPTN(i) = N2(i)
         End If

      Next
     Geo.Shape6.BackColor = &H8000&
Geo.MSFlexGrid1.Rows = UBound(Slintrfc()) + 2 * UBound(Clintrfc()) + 19

     Unload Me
End If
End Sub

Private Sub Form_Load()

If Slintrfc(UBound(Slintrfc())) < 0.3 + SptnIndpth(UBound(SptnIndpth())) Then
   MsgBox "The last SPT Representative Depth should be less then the final of the soil
profile", vbOKOnly, "Invalid Data Entry"
   GoTo ok:
End If
If Geo.Shape5.BackColor = &H8000& Then
      Call SprdSht1_Options
      Shape1.FillColor = QBColor(6)
      Shape2.FillColor = RGB(255, 125, 20)
      k = 1
      j = 1
      Dim density As Single
      ReDim SigZeroatRD(1 To UBound(Clintrfc()))
      Dim Increase As Single
      density = FiDnsty(j)
     For i = 1 To UBound(Clintrfc())

        If Clintrfc(i) <= Slintrfc(k) Then
           If i = 1 Then
             Increase = (Clintrfc(i)) * density
           Else
             If FirstTurn = False Then

                If Clintrfc(i) >= Gwt Then
                  FirstTurn = True
                  density = density - 9.81
                  Increase = ((Gwt - Clintrfc(i - 1)) * FiDnsty(j)) + ((Clintrfc(i) - Gwt) *
density)
                Else
                  Increase = (Clintrfc(i) - Clintrfc(i - 1)) * density
                End If
             Else
               Increase = (Clintrfc(i) - Clintrfc(i - 1)) * density
             End If
           End If
        Else
           k = k + 1
           j = j + 1
           density = FiDnsty(j)
           If FirstTurn = True Then density = density - 9.81
              If FirstTurn = False Then

                If Clintrfc(i) >= Gwt Then
                  FirstTurn = True
                  density = density - 9.81
                  Increase = ((Gwt - Clintrfc(i - 1)) * FiDnsty(j)) + ((Clintrfc(i) - Gwt) *
density)
                Else
                  Increase = ((Slintrfc(k - 1) - Clintrfc(i - 1)) * FiDnsty(j - 1)) + ((Clintrfc(i) -
Slintrfc(k - 1)) * FiDnsty(j))
                End If
              Else
                 If FirstTurn = True Then Increase = ((Slintrfc(k - 1) - Clintrfc(i - 1)) *
(FiDnsty(j - 1) - 9.81)) + ((Clintrfc(i) - Slintrfc(k - 1)) * (FiDnsty(j) - 9.81)) 'effective alıyo
burayı.
                 If FirstTurn = False Then Increase = ((Slintrfc(k - 1) - Clintrfc(i - 1)) *
(FiDnsty(j - 1))) + ((Clintrfc(i) - Slintrfc(k - 1)) * (FiDnsty(j)))
                End If
        End If
          If i = 1 Then
            SigZeroatRD(i) = Increase
          Else
            SigZeroatRD(i) = SigZeroatRD(i - 1) + Increase
          End If
        Adjustment.SprdSht1.TextMatrix(i, 4) = Format(SigZeroatRD(i), "###0.00")
     Next
     RfrncOvrrdnPrssr = 95.76
     ReDim Cn(1 To UBound(SigZeroatRD()))
     ReDim N(1 To UBound(SigZeroatRD()))
     ReDim N2(1 To UBound(SigZeroatRD()))
   For i = 1 To UBound(SigZeroatRD())
      Cn(i) = (RfrncOvrrdnPrssr / SigZeroatRD(i)) ^ 0.5
      If Cn(i) <= 2 Then
        Cn(i) = Cn(i)
```

```
    Else
      Cn(i) = 2
    End If
  Adjustment.SprdSht1.TextMatrix(i, 5) = Format(Cn(i), "###0.00")
  Adjustment.SprdSht1.TextMatrix(i, 3) = Sptn(i)
    N(i) = Cn(i) * Sptn(i)
    If N(i) > 50 Then N(i) = 50
  Adjustment.SprdSht1.TextMatrix(i, 6) = Format(N(i), "###0.00")
  SprdSht1.TextMatrix(i, 2) = Str(Clintrfc(i))
Next
 k = 1
  For i = 1 To UBound(Clintrfc)
    If Clintrfc(i) <= Slintrfc(k) Then
          SprdSht1.TextMatrix(i, 1) = "Layer " & k
    Else
      k = k + 1
          SprdSht1.TextMatrix(i, 1) = "Layer " & k
    End If
  Next
  For i = 1 To 19
    If i < UBound(Slintrfc) Then
      Check1(i).Visible = True
    Else
      Check1(i).Visible = False
    End If
  Next
  k = 1
  For i = 1 To UBound(Clintrfc)
    If Clintrfc(i) <= Slintrfc(k) Then
      If FiSol(k) = "Sand" Then
        For j = 1 To 7
          SprdSht1.Row = i
          SprdSht1.Col = j
          SprdSht1.CellBackColor = QBColor(6)
        Next
      End If
      If FiSol(k) = "Clay" Then
        For j = 1 To 7
          SprdSht1.Row = i
          SprdSht1.Col = j
          SprdSht1.CellBackColor = RGB(255, 125, 20)
        Next
      End If
    Else
      k = k + 1
      If FiSol(k) = "Sand" Then
        For j = 1 To 7
          SprdSht1.Row = i
          SprdSht1.Col = j
          SprdSht1.CellBackColor = QBColor(6)
        Next
      End If
      If FiSol(k) = "Clay" Then
        For j = 1 To 7
          SprdSht1.Row = i
          SprdSht1.Col = j
          SprdSht1.CellBackColor = RGB(255, 125, 20)
        Next
      End If
    End If
  Next
End If
ok:
End Sub

Public Sub SprdSht1_Options()
    SprdSht1Fnctns = False
    With SprdSht1
      .Rows = UBound(Clintrfc()) + 1
      SprdSht1RwsInfnty = True
     If SprdSht1RwsInfnty = True Then
       .Col = 0
      For i = 1 To .Rows - 1
         .Row = i: .Text = "Spt-N #" & Str(i)
      Next
     Else
       .Col = 0
       .Row = 1: .Text = "1"
       .Row = 2: .Text = "2"
       .Row = 3: .Text = "3"
       .Row = 4: .Text = "4"
       .Row = 5: .Text = "5"
       .Row = 6: .Text = "6"
       .Row = 7: .Text = "7"
       .Row = 8: .Text = "8"
       .Row = 9: .Text = "9"
       .Row = 10: .Text = "10"
     End If
     .Cols = 8
     .Row = 0
     .Col = 1: .Text = "Layer #"
     .Col = 2: .Text = "Repr. Depth"
     .Col = 3: .Text = "Spt-N Value"
     .Col = 4: .Text = "SigZero"
     .Col = 5: .Text = "Cn"
     .Col = 6: .Text = "Corr. Spt-N"
     .Col = 7: .Text = "N""
     .ColWidth(0) = 1000
     .ColWidth(1) = 900
     .ColWidth(2) = 1000
```

```
.ColWidth(3) = 1000
.ColWidth(4) = 900
.ColWidth(5) = 700
.ColWidth(6) = 1000
.ColWidth(7) = 700
.FixedAlignment(1) = flexAlignCenterCenter
.FixedAlignment(2) = flexAlignCenterCenter
.FixedAlignment(3) = flexAlignCenterCenter
.FixedAlignment(4) = flexAlignCenterCenter
.FixedAlignment(5) = flexAlignCenterCenter
.FixedAlignment(6) = flexAlignCenterCenter
.FixedAlignment(7) = flexAlignCenterCenter
.ColAlignment(0) = 1  'Give 1 for Left,4 for Center,7 for Right Alignment
.ColAlignment(1) = 4
.ColAlignment(2) = 4
.ColAlignment(3) = 4
.ColAlignment(4) = 4
.ColAlignment(5) = 4
.ColAlignment(6) = 4
.ColAlignment(7) = 4
ReDim SprdSht1TxtLngth(.Cols - 1)
SprdSht1TxtLngth(1) = 0  'Give -1 for zero length,0 for infitive length
SprdSht1TxtLngth(2) = 0
SprdSht1TxtLngth(3) = 0
SprdSht1TxtLngth(4) = 0
SprdSht1TxtLngth(5) = 0
SprdSht1TxtLngth(6) = 0
SprdSht1TxtLngth(7) = 0
ReDim SprdSht1LstBxEntry(.Cols - 1)
SprdSht1LstBxEntry(1) = -1  'Give -1 for the columns that ListBox aren`t needed
SprdSht1LstBxEntry(2) = -1
SprdSht1LstBxEntry(3) = -1
SprdSht1LstBxEntry(4) = -1
SprdSht1LstBxEntry(5) = -1
SprdSht1LstBxEntry(6) = -1
SprdSht1LstBxEntry(7) = -1
ReDim SprdSht1EntryLsts(2, 5)
SprdSht1EntryLsts(1, 0) = ""
SprdSht1EntryLsts(1, 1) = ""
SprdSht1EntryLsts(1, 2) = ""
SprdSht1EntryLsts(1, 3) = ""
SprdSht1EntryLsts(1, 4) = ""
SprdSht1EntryLsts(1, 5) = ""
SprdSht1EntryLsts(2, 0) = "Sand"
SprdSht1EntryLsts(2, 1) = "Clay"

ReDim SprdSht1Frml(1 To .Cols - 1)
ReDim SprdSht1FrmlTrmOtOfGrd(1 To .Cols - 1)
SprdSht1FrmlTrmOtOfGrd(1) = 0
SprdSht1FrmlTrmOtOfGrd(2) = 0
SprdSht1FrmlTrmOtOfGrd(3) = 0
SprdSht1FrmlTrmOtOfGrd(4) = 0
SprdSht1FrmlTrmOtOfGrd(5) = 0
SprdSht1FrmlTrmOtOfGrd(6) = 0
SprdSht1FrmlTrmOtOfGrd(7) = 0
ReDim SprdSht1FrmlOtptTyp(1 To .Cols)
SprdSht1FrmlOtptTyp(1) = ""
SprdSht1FrmlRsltWrtnOtOfGrd = False
.Col = 1
.Row = 1
SprdSht1TxtBxIndx = Int(.ColAlignment(.Col) / 3)
If SprdSht1TxtLngth(SprdSht1.Col) <> -1 Then
  SprdSht1TxtBx(SprdSht1TxtBxIndx).MaxLength =
SprdSht1TxtLngth(SprdSht1.Col)
Else
  SprdSht1TxtBx(SprdSht1TxtBxIndx).MaxLength = 0
End If
SprdSht1TxtBx(SprdSht1TxtBxIndx).Text = .Text
SprdSht1TmrStrt = 4
End With
SprdSht1Fnctns = True
End Sub

Private Sub SprdSht1_GotFocus()
  SprdSht1LstBx.TabIndex = 0
  SprdSht1.TabIndex = 0
End Sub

Private Sub SprdSht1_LeaveCell()
  If SprdSht1Fnctns = True Then
    SprdSht1TxtBx(SprdSht1TxtBxIndx).Visible = False
    SprdSht1LstBx.Visible = False
  End If
End Sub

Private Sub SprdSht1_RowColChange()
  If SprdSht1Fnctns = True Then
    SprdSht1TxtBxIndx = Int(SprdSht1.ColAlignment(SprdSht1.Col) / 3)
    If SprdSht1TxtLngth(SprdSht1.Col) <> -1 Then
      SprdSht1TxtBx(SprdSht1TxtBxIndx).MaxLength = SprdSht1TxtLngth(SprdSht1.Col)
    Else
      SprdSht1TxtBx(SprdSht1TxtBxIndx).MaxLength = 0
    End If
    SprdSht1Fnctns = False
    SprdSht1TxtBx(SprdSht1TxtBxIndx).Text = SprdSht1.Text
    SprdSht1Fnctns = True
    SprdSht1TmrStrt = 4
  End If
End Sub
```

```
Private Sub SprdSht1_Scroll()
    SprdSht1TxtBx(SprdSht1TxtBxIndx).Visible = False
    SprdSht1LstBx.Visible = False
    SprdSht1TmrStrt = 4
End Sub

Private Sub SprdSht1LstBx_Click()
    SprdSht1.Text = SprdSht1LstBx.Text
    SprdSht1TxtBx(Index).Visible = False
End Sub

Private Sub SprdSht1LstBx_DblClick()
    SprdSht1.SetFocus
    SprdSht1TxtBx(SprdSht1TxtBxIndx).Visible = False
    SprdSht1LstBx.Visible = False
End Sub

Private Sub SprdSht1LstBx_GotFocus()
    SprdSht1.TabIndex = 0
    SprdSht1LstBx.TabIndex = 0
End Sub

Private Sub SprdSht1LstBx_KeyDown(KeyCode As Integer, Shift As Integer)
    If KeyCode = vbKeyReturn Then
      SprdSht1.SetFocus
    End If
End Sub

Private Sub SprdSht1Tmr_Timer()

End Sub

Private Sub SprdSht1TxtBx_Change(Index As Integer)
    SprdSht1.Text = SprdSht1TxtBx(SprdSht1TxtBxIndx)
    If SprdSht1Fnctns = True Then
      Call SprdSht1_Formula_Calculate(SprdSht1.Col, SprdSht1.Row)
    End If
End Sub

Private Sub SprdSht1TxtBx_KeyDown(Index As Integer, KeyCode As Integer, Shift As
Integer)
    If KeyCode = vbKeyReturn Then
      SprdSht1.Text = SprdSht1TxtBx(Index)
      SprdSht1TxtBx(Index).Visible = False
      SprdSht1.SetFocus
    End If
End Sub

Private Sub SprdSht1TxtBx_LostFocus(Index As Integer)
    SprdSht1TxtBx(SprdSht1TxtBxIndx).Visible = False
End Sub

Public Sub SprdSht1_Formula_Calculate(FrmlTrmClmn As Integer, FrmlTrmRw As Integer)
    Dim FrmlTxt As String
    Dim FrmlRsltClmn As Integer
    Dim FrmlRsltRw As Integer
    Dim FrmlCllBgn As Integer
    Dim FrmlCllEnd As Integer
    Dim FrmlCllClmn As Integer
    Dim FrmlCllRw As Integer
    Dim FrmlFnctnsPlc() As Integer
    Dim FrmlFnctn As String
    Dim FrmlFnctnBgn As Integer
    Dim FrmlFnctnEnd As Integer
    For i = 1 To UBound(SprdSht1Frml())
      FrmlTxt = SprdSht1Frml(i)
      f = 0
      Do While InStr(f + 1, FrmlTxt, Mid(Str(FrmlTrmClmn), 2)) <> 0
        f = InStr(f + 1, FrmlTxt, Mid(Str(FrmlTrmClmn), 2))
        If Mid(FrmlTxt, f + 1, 1) = "," Then
          f = f + 1
          FrmlRsltClmn = i
          FrmlRsltRw = FrmlTrmRw - _
              Sgn(Val(Mid(FrmlTxt, f + 1, InStr(f + 1, FrmlTxt, ")") - f - 1))) * _
              Abs(Val(Mid(FrmlTxt, f + 1, InStr(f + 1, FrmlTxt, ")") - f - 1)))
          f = 0
          Do While InStr(f + 1, FrmlTxt, "(") <> 0
            f = InStr(f + 1, FrmlTxt, "(")
            FrmlCllClmn = Val(Mid(FrmlTxt, f + 1, InStr(f + 1, FrmlTxt, ",") - f - 1))
            FrmlCllBgn = f
            f = InStr(f + 1, FrmlTxt, ",")
            FrmlCllRw = FrmlRsltRw + Val(Mid(FrmlTxt, f + 1, InStr(f + 1, FrmlTxt, ")") -
f - 1))
            FrmlCllEnd = InStr(f + 1, FrmlTxt, ")")
            FrmlTxt = Left(FrmlTxt, FrmlCllBgn - 1) & "<" &
SprdSht1.TextMatrix(FrmlCllRw, FrmlCllClmn) & ">" & Mid(FrmlTxt, FrmlCllEnd + 1)
          Loop
          f = 0: j = 0
          Do While InStr(f + 1, FrmlTxt, "[") <> 0
            j = j + 1
            ReDim Preserve FrmlFnctnsPlc(1 To j)
            f = InStr(f + 1, FrmlTxt, "[")
            FrmlFnctnsPlc(j) = f
          Loop
          For f = j To 1 Step -1
            If FrmlFnctnsPlc(f) > 8 Then
              FrmlFnctn = Mid(FrmlTxt, FrmlFnctnsPlc(f) - 8, 8)
            Else
              FrmlFnctn = Left(FrmlTxt, FrmlFnctnsPlc(f) - 1)
            End If
```

```
            If Right(FrmlFnctn, 4) = "phss" Then
              FrmlFnctnBgn = FrmlFnctnsPlc(f) - 4
              FrmlFnctnEnd = InStr(FrmlFnctnsPlc(f) + 1, FrmlTxt, "]")
              FrmlTxt = Left(FrmlTxt, FrmlFnctnBgn - 1) & "<" & _
                    SprdSht1_Operation_Calculate(Mid _
                    (FrmlTxt, FrmlFnctnsPlc(f) + 1, FrmlFnctnEnd - FrmlFnctnsPlc(f) - 1)) &
">" & _
                    Mid(FrmlTxt, FrmlFnctnEnd + 1)
            End If
          Next
          SprdSht1.TextMatrix(FrmlRsltRw, FrmlRsltClmn) = Mid(FrmlTxt, 2,
Len(FrmlTxt) - 2)
          Exit Do
        End If
      Loop
    Next
End Sub

Public Function SprdSht1_Operation_Calculate(OprtnTxt As String) As String
    Dim FrmlOprtnRslt As Single
    Dim FrmlOprtnTrm As Single
    Dim FrmlOprtn As String
    f = InStr(OprtnTxt, "<")
    FrmlOprtnRslt = Val(Mid(OprtnTxt, f + 1, InStr(f + 1, OprtnTxt, ">") - f - 1))
    While InStr(f + 1, OprtnTxt, "<") <> 0
        f = InStr(f + 1, OprtnTxt, ">")
        FrmlOprtn = Mid(OprtnTxt, f + 1, 1)
        f = InStr(f + 1, OprtnTxt, "<")
        FrmlOprtnTrm = Val(Mid(OprtnTxt, f + 1, InStr(f + 1, OprtnTxt, ">") - f - 1))
        Select Case FrmlOprtn
            Case "+"
                FrmlOprtnRslt = FrmlOprtnRslt + FrmlOprtnTrm
            Case "-"
                FrmlOprtnRslt = FrmlOprtnRslt - FrmlOprtnTrm
            Case "*"
                FrmlOprtnRslt = FrmlOprtnRslt * FrmlOprtnTrm
            Case "/"
                FrmlOprtnRslt = FrmlOprtnRslt / FrmlOprtnTrm
        End Select
    Wend
    SprdSht1_Operation_Calculate = Mid(Str(FrmlOprtnRslt), 2)
End Function

Dim q() As Single
Private Sub Command1_Click()
With MSFlexGrid1
  ReDim Qb(1 To UBound(Clintrfc))
  ReDim Qs(1 To UBound(Clintrfc))
  ReDim Qbcr(1 To UBound(Clintrfc))
  ReDim Qbcr2(1 To UBound(Clintrfc))
  ReDim MinQb(1 To UBound(Clintrfc))
    For i = 1 To UBound(Clintrfc())
        Qb(i) = .TextMatrix(i, 4)
        Qs(i) = .TextMatrix(i, 5)
        Qbcr(i) = .TextMatrix(i, 8)
        Qbcr2(i) = .TextMatrix(i, 9)
        MinQb(i) = .TextMatrix(i, 10)
    Next
  ReDim Qu1(1 To UBound(Clintrfc))
  ReDim Qu2(1 To UBound(Clintrfc))
  ReDim MinQu(1 To UBound(Clintrfc))
    For i = 1 To UBound(Clintrfc())
      Qu1(i) = Qbcr(i) + Qs(i)
      Qu2(i) = Qbcr2(i) + Qs(i)
      MinQu(i) = MinQb(i) + Qs(i)
    Next
End With
Geo.Shape7.BackColor = &H8000&
Geo.Shape9.BackColor = &H8000&
Unload Me
End Sub

Private Sub Command3_Click()
 Unload Me
End Sub

Private Sub Form_Load()
  With MSFlexGrid1
  .Rows = UBound(Clintrfc()) + 1
    .Col = 0
    For i = 1 To .Rows - 1
      .Row = i: .Text = "Spt-N #" & Str(i)
    Next
  .Cols = 11
  .Row = 0
  .Col = 1: .Text = "Layer #"
  .Col = 2: .Text = "Soil Type"
  .Col = 3: .Text = "Rpre. Depth"
  .Col = 4: .Text = "Qb (kN)"
  .Col = 5: .Text = "Qs (kN)"
  .Col = 6: .Text = "A"
  .Col = 7: .Text = "B"
  .Col = 8: .Text = "Meyerhof's Method"
  .Col = 9: .Text = "Average Method"
  .Col = 10: .Text = "Min. Base Resistance"
  .ColWidth(0) = 1000
  .ColWidth(1) = 700
  .ColWidth(2) = 800
  .ColWidth(3) = 1000
  .ColWidth(4) = 800
```

```
  .ColWidth(5) = 1000
  .ColWidth(6) = 0
  .ColWidth(7) = 0
  .ColWidth(8) = 1450
  .ColWidth(9) = 1300
  .ColWidth(10) = 1700
  .FixedAlignment(1) = flexAlignCenterCenter
  .FixedAlignment(2) = flexAlignCenterCenter
  .FixedAlignment(3) = flexAlignCenterCenter
  .FixedAlignment(4) = flexAlignCenterCenter
  .FixedAlignment(5) = flexAlignCenterCenter
  .FixedAlignment(6) = flexAlignCenterCenter
  .FixedAlignment(7) = flexAlignCenterCenter
  .FixedAlignment(8) = flexAlignCenterCenter
  .FixedAlignment(9) = flexAlignCenterCenter
  .FixedAlignment(10) = flexAlignCenterCenter
  .ColAlignment(0) = 1  'Give 1 for Left,4 for Center,7 for Right Alignment
  .ColAlignment(1) = 4
  .ColAlignment(2) = 4
  .ColAlignment(3) = 4
  .ColAlignment(4) = 4
  .ColAlignment(5) = 4
  .ColAlignment(6) = 4
  .ColAlignment(7) = 4
  .ColAlignment(8) = 4
  .ColAlignment(9) = 4
  .ColAlignment(10) = 4
  For i = 1 To UBound(Clintrfc)
     .TextMatrix(i, 1) = Bearing_Capacity.MSFlexGrid1.TextMatrix(i, 1)
     .TextMatrix(i, 2) = Bearing_Capacity.MSFlexGrid1.TextMatrix(i, 2)
     .TextMatrix(i, 3) = Bearing_Capacity.MSFlexGrid1.TextMatrix(i, 3)
     .TextMatrix(i, 4) = Format((Bearing_Capacity.MSFlexGrid1.TextMatrix(i, 8) *
Bearing_Capacity.MSFlexGrid1.TextMatrix(i, 9)), "###0.00")
     If i = 1 Then
        .TextMatrix(i, 5) = Format((Area(i) * Bearing_Capacity.MSFlexGrid1.TextMatrix(i,
11)), "###0.00")
     Else
        For P = i To 1 Step -1
           If P - 1 = 0 Then Exit For
           TotalFs = TotalFs + TotalArea(P - 1) * Fs(P - 1)
        Next
        .TextMatrix(i, 5) = Format((TotalFs + Area(i) *
Bearing_Capacity.MSFlexGrid1.TextMatrix(i, 11)), "###0.00")
        TotalFs = 0
     End If
  Next
 With MSFlexGrid1
 k = 1
     For i = 1 To UBound(Clintrfc)
        If Clintrfc(i) <= Slintrfc(k) Then
           If FiSol(k) = "Sand" Then
              For j = 1 To 10
                 .Row = i
                 .Col = j
                 .CellBackColor = QBColor(6)
              Next
           End If
           If FiSol(k) = "Clay" Then
              For j = 1 To 10
                 .Row = i
                 .Col = j
                 .CellBackColor = RGB(255, 125, 20)
              Next
           End If
        Else
           k = k + 1
           If FiSol(k) = "Sand" Then
              For j = 1 To 10
                 .Row = i
                 .Col = j
                 .CellBackColor = QBColor(6)
              Next
           End If
           If FiSol(k) = "Clay" Then
              For j = 1 To 10
                 .Row = i
                 .Col = j
                 .CellBackColor = RGB(255, 125, 20)
              Next
           End If
        End If
     Next
End With
 With MSFlexGrid1
 ReDim Preserve q(1 To UBound(Slintrfc))
 Dim ToplamQb As Single
 k = 0
 On Error Resume Next
 For i = 1 To UBound(Slintrfc)
   L = 0
   Do
     k = k + 1
     ToplamQb = ToplamQb + .TextMatrix(k, 4)
     L = L + 1
   Loop While .TextMatrix(k, 1) = .TextMatrix(k + 1, 1)
   q(i) = ToplamQb / L
   ToplamQb = 0
 Next
 A = 0
 B = 0
```

```
C = 0
k = 0
i = 0
For k = 1 To UBound(Slintrfc())
  Do
  i = i + 1
  If k <> 1 Then
    If .TextMatrix(i, 4) > q(k - 1) Then
      A = q(k - 1) + ((.TextMatrix(i, 4) - q(k - 1)) * (Clintrfc(i) - Slintrfc(k - 1))) / (10 *
PlDmtr)
    End If
  End If

  If k = UBound(Slintrfc()) Then
  Else
    If k = 1 Then
      B = q(k + 1) + ((.TextMatrix(i, 4) - q(k + 1)) * (Slintrfc(k) - Clintrfc(i))) / (10 *
PlDmtr)
      If B < 0 Then B = 0
    Else
      If .TextMatrix(i, 4) > q(k + 1) Then
        B = q(k + 1) + ((.TextMatrix(i, 4) - q(k + 1)) * (Slintrfc(k) - Clintrfc(i))) / (10 *
PlDmtr)
      End If
    End If
  End If
  .TextMatrix(i, 6) = Format((A), "###0.00")
  .TextMatrix(i, 7) = Format((B), "###0.00")
  If A = 0 Then
    If B <= Val(.TextMatrix(i, 4)) Then
      .TextMatrix(i, 8) = Format((B), "###0.00")
    Else
      .TextMatrix(i, 8) = .TextMatrix(i, 4)
    End If
  End If
  If B = 0 Then
    If A <= Val(.TextMatrix(i, 4)) Then
      .TextMatrix(i, 8) = Format((A), "###0.00")
    Else
      .TextMatrix(i, 8) = .TextMatrix(i, 4)
    End If
  End If
  If A = 0 And B = 0 Then
    .TextMatrix(i, 8) = .TextMatrix(i, 4)
  End If
  If A <> 0 And B <> 0 Then
    If A < B Then C = A
    If B <= A Then C = B
    If C <= Val(.TextMatrix(i, 4)) Then
      .TextMatrix(i, 8) = Format((C), "###0.00")
    Else
      .TextMatrix(i, 8) = .TextMatrix(i, 4)
    End If
  End If
  A = 0
  B = 0
  C = 0
  Loop While .TextMatrix(i, 1) = .TextMatrix(i + 1, 1)
Next
  UpD = 5
  DownD = 3
UpD = Val(InputBox("Enter the times of the pile diameter to top of the pile inwhich the
averaging will perform for base resistance correction", "Current Value is" & Str(UpD) & "
times of the pile diameter", Str(UpD)))
DownD = Val(InputBox("Enter the times of the pile diameter to bottom of the pile inwhich
the averaging will perform for base resistance correction", "Current Value is" & Str(DownD)
& " times of the pile diameter", Str(DownD)))
With BC_Adjustment.MSFlexGrid1
  For i = 1 To UBound(Clintrfc())
  Down = Clintrfc(i) - UpD * PlDmtr
  Up = Clintrfc(i) + DownD * PlDmtr
    For j = 1 To UBound(Clintrfc())
      If Clintrfc(j) >= Down Then
        If Clintrfc(j) >= Up Then Exit For
        ToplamDeger = ToplamDeger + Val(Str(.TextMatrix(j, 4)))
        R = R + 1
      Else
      End If
    Next
  Ort = ToplamDeger / R
  .TextMatrix(i, 9) = Format((Ort), "###0.00")
  Ort = 0
  ToplamDeger = 0
  R = 0
Next
End With
End With
For i = 1 To UBound(Clintrfc())
  If Val(.TextMatrix(i, 8)) < Val(.TextMatrix(i, 9)) Then
    .TextMatrix(i, 10) = (.TextMatrix(i, 8))
  Else
    .TextMatrix(i, 10) = (.TextMatrix(i, 9))
  End If
Next
End With
End Sub

Private Sub Command1_Click()
  If Text1 = "" Then
    Ip_Beta = 0
```

```
    Else
      Ip_Beta = Val(Text1)
    End If
    Text1 = ""
    If Text2 = "" Then
      Ocr_Beta = 0
    Else
      Ocr_Beta = Val(Text2)
    End If
    Text2 = ""
With Bearing_Capacity.MSFlexGrid1
Dim pi
pi = 4 * Atn(1)
      For i = Pstn To UBound(Clintrfc())
      On Error Resume Next
    Ocr_Sig = .TextMatrix(i, 7) / (0.11 + 0.0037 * Ip_Beta)
    Ocr = Ocr_Sig / SigZeroatRD(i)
    If Ocr > 1.1 Then
      Ocr_Beta = Ocr
    Else
      Ocr_Beta = 0
    End If
End If
      If .TextMatrix(i, 1) = .TextMatrix(i + 1, 1) Then
            qF(i) = 9 * Cu(i)
            If Ip_Beta = 0 Then
              Ip = ((.TextMatrix(i, 7) / SigZeroatRD(i)) - 0.11) / 0.0037
            Else
              Ip = Ip_Beta
            End If
            FiForBeta = -1.725925926E-09 * Ip ^ 6 + 5.22008547122E-07 * Ip ^ 5 -
6.0843447306933E-05 * Ip ^ 4 + 3.41075563410272E-03 * Ip ^ 3 - 9.04126617943605E-02
* Ip ^ 2 + 0.628295029064852 * Ip + 26.0322222100624
            If Ip < 10 Then FiForBeta = 26.15
            If Ip > 90 Then FiForBeta = 9.9
            If Ocr_Beta = 0 Then
              Fs(i) = ((1 - Sin(FiForBeta * pi / 180)) * Tan(FiForBeta * pi / 180) *
SigZeroatRD(i))
            Else
              Fs(i) = ((1 - Sin(FiForBeta * pi / 180)) * Tan(FiForBeta * pi / 180) *
SigZeroatRD(i)) * Ocr_Beta
            End If
            .TextMatrix(i, 4) = "Hansen"
            .TextMatrix(i, 5) = "Beta"
            .TextMatrix(i, 9) = Format(qF(i), "###0.00")
            .TextMatrix(i, 11) = Format(Fs(i), "###0.00")
        Else
            Pstn = i + 1
            qF(i) = 9 * Cu(i)
            If Ip_Beta = 0 Then
              Ip = ((.TextMatrix(i, 7) / SigZeroatRD(i)) - 0.11) / 0.0037
            Else
              Ip = Ip_Beta
            End If
            FiForBeta = -1.725925926E-09 * Ip ^ 6 + 5.22008547122E-07 * Ip ^ 5 -
6.0843447306933E-05 * Ip ^ 4 + 3.41075563410272E-03 * Ip ^ 3 - 9.04126617943605E-02
* Ip ^ 2 + 0.628295029064852 * Ip + 26.0322222100624
            If Ip < 10 Then FiForBeta = 26.15
            If Ip > 90 Then FiForBeta = 9.9
            If Ocr_Beta = 0 Then
              Fs(i) = ((1 - Sin(FiForBeta * pi / 180)) * Tan(FiForBeta * pi / 180) *
SigZeroatRD(i))
            Else
              Fs(i) = ((1 - Sin(FiForBeta * pi / 180)) * Tan(FiForBeta * pi / 180) *
SigZeroatRD(i)) * Ocr_Beta
            End If
            .TextMatrix(i, 4) = "Hansen"
            .TextMatrix(i, 5) = "Beta"
            .TextMatrix(i, 9) = Format(qF(i), "###0.00")
            .TextMatrix(i, 11) = Format(Fs(i), "###0.00")
        Exit For
      End If
  Next
End With
If Bearing_Capacity.MSFlexGrid1.TextMatrix(Pstn, 2) = "Sand" Then
Bearing_Capacity.Frame2.Enabled = False: Bearing_Capacity.Frame1.Enabled = True
If Bearing_Capacity.MSFlexGrid1.TextMatrix(Pstn, 2) = "Clay" Then
Bearing_Capacity.Frame1.Enabled = False: Bearing_Capacity.Frame2.Enabled = True
BC_Beta.Hide
Option2.Value = False
End Sub

Dim CrtclDpth As Single
Dim Embddddpth As Single
Dim ToplamFi As Single
Dim OrtalamaFi As Single
Dim Nq As Single
Dim ZcDOrani As Single
Dim Zc As Single
Dim KsTanFi As Single
Dim Alfa As Single
Dim Ip As Single
Dim FiForBeta As Single
Dim P As Integer
Dim S As Single

Private Sub Command1_Click()
If MSFlexGrid1.TextMatrix(1, 1) = "" Then
Else
  If MSFlexGrid1.TextMatrix(4, 4) = "" Then
```

```
    Else
      BC_Adjustment.Show
      ReDim Soiltype(1 To UBound(Clintrfc))
      ReDim CalMethod(1 To UBound(Clintrfc))
      For i = 1 To UBound(Clintrfc())
        Soiltype(i) = MSFlexGrid1.TextMatrix(i, 2)
        CalMethod(i) = MSFlexGrid1.TextMatrix(i, 5)
      Next
      Unload Me
    End If
End If
End Sub

Private Sub Command2_Click()
  Unload Me
End Sub

Private Sub Command3_Click()
    With MSFlexGrid1
  For i = Pstn To UBound(Clintrfc())
    On Error Resume Next
    If .TextMatrix(i, 1) = .TextMatrix(i + 1, 1) Then
        Fs(i) = (CrrctdSPTN(i) * EnrgyRt / 55) * 2
          If Pstn = 1 Then
            Embddddpth = Clintrfc(i)
          Else
            Embddddpth = Clintrfc(i) - CL(Pstn - 1)
          End If
        CrtclDpth = Embddddpth / (PlDmtr)
        If CrtclDpth >= 10 Then
          CrtclDpth = 10
        End If
        qF(i) = 40 * (CrrctdSPTN(i) * EnrgyRt / 55) * CrtclDpth
        If PlInstlltnMthd = 2 Then
          qF(i) = qF(i) / 3
          Fs(i) = Fs(i) / 2
        End If
        .TextMatrix(i, 4) = "Meyerhof"
        .TextMatrix(i, 5) = "Meyerhof"
        .TextMatrix(i, 9) = Format(qF(i), "###0.00")
        .TextMatrix(i, 11) = Format(Fs(i), "###0.00")
     Else
        Fs(i) = (CrrctdSPTN(i) * EnrgyRt / 55) * 2 '
          If Pstn = 1 Then
            Embddddpth = Clintrfc(i)
          Else
            Embddddpth = Clintrfc(i) - CL(Pstn - 1)
          End If
        Pstn = i + 1
        CrtclDpth = Embddddpth / (PlDmtr) '
        If CrtclDpth >= 10 Then
          CrtclDpth = 10
        End If
        qF(i) = 40 * (CrrctdSPTN(i) * EnrgyRt / 55) * CrtclDpth
        If PlInstlltnMthd = 2 Then '
          qF(i) = qF(i) / 3
          Fs(i) = Fs(i) / 2
        End If

        .TextMatrix(i, 4) = "Meyerhof"
        .TextMatrix(i, 5) = "Meyerhof"
        .TextMatrix(i, 9) = Format(qF(i), "###0.00")
        .TextMatrix(i, 11) = Format(Fs(i), "###0.00")
      Exit For
    End If
  Next
End With
If MSFlexGrid1.TextMatrix(Pstn, 2) = "Sand" Then Frame2.Enabled = False:
Frame1.Enabled = True
If MSFlexGrid1.TextMatrix(Pstn, 2) = "Clay" Then Frame1.Enabled = False:
Frame2.Enabled = True
S = S + 1
End Sub

Private Sub Command4_Click()
Dim pi
pi = 4 * Atn(1)
With MSFlexGrid1
P = 0
  For k = Pstn To UBound(Clintrfc())
    On Error Resume Next
    If .TextMatrix(k, 1) = .TextMatrix(k + 1, 1) Then
      ToplamFi = ToplamFi + .TextMatrix(k, 6)
    Else
      ToplamFi = ToplamFi + .TextMatrix(k, 6)
    Exit For
    End If
  Next
  If Pstn = 1 Then OrtalamaFi = ToplamFi / k
  If Pstn <> 1 Then OrtalamaFi = ToplamFi / (k - Pstn + 1)
      If PlInstlltnMthd = 1 Then 'driven
        OrtalamaFi = 0.75 * OrtalamaFi + 10
      End If
      If PlInstlltnMthd = 2 Then 'bored da
        OrtalamaFi = OrtalamaFi - 3
      End If
'-------------------------------------------
  For i = Pstn To UBound(Clintrfc())
    On Error Resume Next
    If .TextMatrix(i, 1) = .TextMatrix(i + 1, 1) Then
```

```
        Nq = 2.718281828 ^ (pi * Tan(.TextMatrix(i, 6) * pi / 180)) * (Tan((45 +
.TextMatrix(i, 6) / 2) * pi / 180)) ^ 2
        ZcDOrani = -3.7065703489159E-05 * (OrtalamaFi) ^ 6 + 7.39780305152848E-03 *
(OrtalamaFi) ^ 5 - 0.611594262146106 * (OrtalamaFi) ^ 4 + 26.8172024724446 *
(OrtalamaFi) ^ 3 - 657.955795799946 * (OrtalamaFi) ^ 2 + 8566.39196707057 *
(OrtalamaFi) - 46245.0737576944
          If ZcDOrani < 5 Then ZcDOrani = 5
          If ZcDOrani > 20 Then ZcDOrani = 20
        Zc = ZcDOrani * PlDmtr
        If PlInstlltnMthd = 1 Then
          KsTanFi = -4.80162825778E-07 * (0.75 * .TextMatrix(i, 6) + 10) ^ 6 +
1.0890655192286E-04 * (0.75 * .TextMatrix(i, 6) + 10) ^ 5 - 1.00882457778821E-02 * (0.75
* .TextMatrix(i, 6) + 10) ^ 4 + 0.490286981835297 * (0.75 * .TextMatrix(i, 6) + 10) ^ 3 -
13.2136304267911 * (0.75 * .TextMatrix(i, 6) + 10) ^ 2 + 187.546023293456 * (0.75 *
.TextMatrix(i, 6) + 10) - 1095.71671425429
          If KsTanFi < 1 Then KsTanFi = 1
          If KsTanFi > 2.7 Then KsTanFi = 2.7
        End If
        If PlInstlltnMthd = 2 Then
          KsTanFi = 2.39772058785E-06 * (.TextMatrix(i, 6)) ^ 6 - 4.7462597085435E-04 *
(.TextMatrix(i, 6)) ^ 5 + 3.90262889871078E-02 * (.TextMatrix(i, 6)) ^ 4 -
1.70606276193051 * (.TextMatrix(i, 6)) ^ 3 + 41.8261858202324 * (.TextMatrix(i, 6)) ^ 2 -
545.390374312363 * (.TextMatrix(i, 6)) + 2956.07600973366
          If (.TextMatrix(i, 6)) < 30 Then KsTanFi = 0.057735
          If (.TextMatrix(i, 6)) > 40 Then KsTanFi = 1.006991
        End If

If S = 0 Then
  X = Zc
Else
  X = Slintrfc(S) + Zc
End If
Sig = 0
  If X <= Gwt Then
    If X <= Slintrfc(1) Then
      Sig = X * FiDnsty(1)
    Else
      For P = 1 To UBound(Slintrfc())
        If X <= Slintrfc(P) Then
        current = P
        Exit For
        End If
      Next

      For w = (current - 1) To 1 Step -1
        If w = 1 Then
          B = 0
        Else
          B = Slintrfc(w - 1)
        End If
        Sig = Sig + (Slintrfc(w) - B) * FiDnsty(w)
      Next
      Sig = Sig + (X - Slintrfc(current - 1)) * FiDnsty(current)
    End If
  Else
    If Gwt <= Slintrfc(1) Then
      Sig = Gwt * FiDnsty(1)
    Else
      For P = 1 To UBound(Slintrfc())
        If Gwt <= Slintrfc(P) Then
        currentgwt = P
        Exit For
        End If
      Next
      For w = (currentgwt - 1) To 1 Step -1
        If w - 1 = 0 Then
          C = 0
        Else
          C = Slintrfc(w - 1)
        End If
        Sig = Sig + (Slintrfc(w) - C) * FiDnsty(w)
      Next
      Sig = Sig + (Gwt - Slintrfc(currentgwt - 1)) * FiDnsty(currentgwt)
    End If
    For P = 1 To UBound(Slintrfc())
      If Gwt <= Slintrfc(P) Then
        currentgwt = P
        Exit For
      End If
    Next
    If Gwt < X And X <= Slintrfc(currentgwt) Then
      Sig = Sig + (X - Gwt) * (FiDnsty(currentgwt) - 9.81)
    End If
    If X > Slintrfc(currentgwt) Then
      Sig = Sig + (Slintrfc(currentgwt) - Gwt) * (FiDnsty(currentgwt) - 9.81)
      For P = 1 To UBound(Slintrfc())
        If X <= Slintrfc(P) Then
          CurrentX = P
          Exit For
        End If
      Next
      For w = (CurrentX - 1) To currentgwt + 1 Step -1
        Sig = Sig + (Slintrfc(w) - Slintrfc(w - 1)) * (FiDnsty(w) - 9.81)
      Next
      Sig = Sig + (X - Slintrfc(CurrentX - 1)) * (FiDnsty(CurrentX) - 9.81)

    End If

End If
    If Pstn = 1 Then
```

```
            If Clintrfc(i) <= Zc Then
               qF(i) = SigZeroatRD(i) * Nq
               Fs(i) = SigZeroatRD(i) * KsTanFi
               P = P + 1
            Else
               qF(i) = Sig * Nq
               Fs(i) = Sig * KsTanFi
            End If
         Else
            If (Clintrfc(i) - Slintrfc(S)) <= Zc Then
               qF(i) = SigZeroatRD(i) * Nq
               Fs(i) = SigZeroatRD(i) * KsTanFi
               P = P + 1
            Else
               qF(i) = Sig * Nq
               Fs(i) = Sig * KsTanFi
            End If
         End If
         .TextMatrix(i, 4) = "Static Formula"
         .TextMatrix(i, 5) = "Static Formula"
         .TextMatrix(i, 9) = Format(qF(i), "###0.00")
         .TextMatrix(i, 11) = Format(Fs(i), "###0.00")

      Else
         Nq = 2.718281828 ^ (pi * Tan(.TextMatrix(i, 6) * pi / 180)) * (Tan((45 +
.TextMatrix(i, 6) / 2) * pi / 180)) ^ 2
         ZcDOrani = -3.7065703489159E-05 * (OrtalamaFi) ^ 6 + 7.39780305152848E-03 *
(OrtalamaFi) ^ 5 - 0.611594262146106 * (OrtalamaFi) ^ 4 + 26.8172024724446 *
(OrtalamaFi) ^ 3 - 657.955795799946 * (OrtalamaFi) ^ 2 + 8566.39196707057 *
(OrtalamaFi) - 46245.0737576944
            If ZcDOrani < 5 Then ZcDOrani = 5
            If ZcDOrani > 20 Then ZcDOrani = 20
         Zc = ZcDOrani * PlDmtr
         If PlInstlltnMthd = 1 Then 'driven
            KsTanFi = -4.80162825778E-07 * (0.75 * .TextMatrix(i, 6) + 10) ^ 6 +
1.0890655192286E-04 * (0.75 * .TextMatrix(i, 6) + 10) ^ 5 - 1.00882457778821E-02 * (0.75
* .TextMatrix(i, 6) + 10) ^ 4 + 0.490286981835297 * (0.75 * .TextMatrix(i, 6) + 10) ^ 3 -
13.2136304267911 * (0.75 * .TextMatrix(i, 6) + 10) ^ 2 + 187.546023293456 * (0.75 *
.TextMatrix(i, 6) + 10) - 1095.71671425429
            If KsTanFi < 1 Then KsTanFi = 1
            If KsTanFi > 2.7 Then KsTanFi = 2.7
         End If
         If PlInstlltnMthd = 2 Then ' bored
            KsTanFi = 2.39772058785E-06 * (.TextMatrix(i, 6)) ^ 6 - 4.7462597085435E-04 *
(.TextMatrix(i, 6)) ^ 5 + 3.90262889871078E-02 * (.TextMatrix(i, 6)) ^ 4 -
1.70606276193051 * (.TextMatrix(i, 6)) ^ 3 + 41.8261858202324 * (.TextMatrix(i, 6)) ^ 2 -
545.390374312363 * (.TextMatrix(i, 6)) + 2956.07600973366
            If (.TextMatrix(i, 6)) < 30 Then KsTanFi = 0.057735
            If (.TextMatrix(i, 6)) > 40 Then KsTanFi = 1.006991
         End If
```

```
         If Pstn = 1 Then
            If Clintrfc(i) <= Zc Then
               qF(i) = SigZeroatRD(i) * Nq
               Fs(i) = SigZeroatRD(i) * KsTanFi
               P = P + 1
            Else
If S = 0 Then
   X = Zc
Else
   X = Slintrfc(S) + Zc
End If
Sig = 0
   If X <= Gwt Then
      If X <= Slintrfc(1) Then
         Sig = X * FiDnsty(1)
      Else
         For P = 1 To UBound(Slintrfc())
            If X <= Slintrfc(P) Then
            current = P
            Exit For
            End If
         Next
         For w = (current - 1) To 1 Step -1
            If w = 1 Then
               B = 0
            Else
               B = Slintrfc(w - 1)
            End If
            Sig = Sig + (Slintrfc(w) - B) * FiDnsty(w)
         Next

         Sig = Sig + (X - Slintrfc(current - 1)) * FiDnsty(current)
      End If
   Else
      If Gwt <= Slintrfc(1) Then
         Sig = Gwt * FiDnsty(1)
      Else
         For P = 1 To UBound(Slintrfc())
            If Gwt <= Slintrfc(P) Then
            currentgwt = P
            Exit For
            End If
         Next

         For w = (currentgwt - 1) To 1 Step -1
            If w - 1 = 0 Then
               C = 0
            Else
               C = Slintrfc(w - 1)
            End If
```

```
        Sig = Sig + (Slintrfc(w) - C) * FiDnsty(w)
      Next

      Sig = Sig + (Gwt - Slintrfc(currentgwt - 1)) * FiDnsty(currentgwt)

    End If
    For P = 1 To UBound(Slintrfc())
      If Gwt <= Slintrfc(P) Then
        currentgwt = P
        Exit For
      End If
    Next
    If Gwt < X And X <= Slintrfc(currentgwt) Then
      Sig = Sig + (X - Gwt) * (FiDnsty(currentgwt) - 9.81)
    End If
    If X > Slintrfc(currentgwt) Then
      Sig = Sig + (Slintrfc(currentgwt) - Gwt) * (FiDnsty(currentgwt) - 9.81)
      For P = 1 To UBound(Slintrfc())
        If X <= Slintrfc(P) Then
          CurrentX = P
          Exit For
        End If
      Next
      For w = (CurrentX - 1) To currentgwt + 1 Step -1
        Sig = Sig + (Slintrfc(w) - Slintrfc(w - 1)) * (FiDnsty(w) - 9.81)
      Next
      Sig = Sig + (X - Slintrfc(CurrentX - 1)) * (FiDnsty(CurrentX) - 9.81)
    End If
  End If
          qF(i) = Sig * Nq
          Fs(i) = Sig * KsTanFi
        End If
      Else
        If (Clintrfc(i) - Slintrfc(S)) <= Zc Then
          qF(i) = SigZeroatRD(i) * Nq
          Fs(i) = SigZeroatRD(i) * KsTanFi
          P = P + 1
        Else
          qF(i) = Sig * Nq
          Fs(i) = Sig * KsTanFi
        End If
      End If
      .TextMatrix(i, 4) = "Static Formula"
      .TextMatrix(i, 5) = "Static Formula"
      .TextMatrix(i, 9) = Format(qF(i), "###0.00")
      .TextMatrix(i, 11) = Format(Fs(i), "###0.00")
      S = S + 1
      Pstn = i + 1
      Exit For
    End If
  Next
End With
  ToplamFi = 0
  OrtalamaFi = 0
If MSFlexGrid1.TextMatrix(Pstn, 2) = "Sand" Then Frame2.Enabled = False:
Frame1.Enabled = True
If MSFlexGrid1.TextMatrix(Pstn, 2) = "Clay" Then Frame1.Enabled = False:
Frame2.Enabled = True
End Sub

Private Sub Command5_Click()
    With MSFlexGrid1
  For i = Pstn To UBound(Clintrfc())
    On Error Resume Next
    If .TextMatrix(i, 1) = .TextMatrix(i + 1, 1) Then
      If Cu(i) >= 75 Then Alfa = 0.5
      If 25 < Cu(i) < 75 Then
        Alfa = -0.01 * Cu(i) + 1.25
      End If
      If Cu(i) <= 25 Then
        Alfa = 1
      End If
      qF(i) = 9 * Cu(i)
      Fs(i) = Alfa * Cu(i)
      .TextMatrix(i, 4) = "Hansen"
      .TextMatrix(i, 5) = "Alfa"
      .TextMatrix(i, 9) = Format(qF(i), "###0.00")
      .TextMatrix(i, 11) = Format(Fs(i), "###0.00")
    Else
      Pstn = i + 1
      If Cu(i) >= 75 Then Alfa = 0.5
      If 25 < Cu(i) < 75 Then
        Alfa = -0.01 * Cu(i) + 1.25
      End If
      If Cu(i) <= 25 Then
        Alfa = 1
      End If
      qF(i) = 9 * Cu(i)
      Fs(i) = Alfa * Cu(i)
      .TextMatrix(i, 4) = "Hansen"
      .TextMatrix(i, 5) = "Alfa"
      .TextMatrix(i, 9) = Format(qF(i), "###0.00")
      .TextMatrix(i, 11) = Format(Fs(i), "###0.00")
      Exit For
    End If
  Next
End With
If MSFlexGrid1.TextMatrix(Pstn, 2) = "Sand" Then Frame2.Enabled = False:
Frame1.Enabled = True
```

```
If MSFlexGrid1.TextMatrix(Pstn, 2) = "Clay" Then Frame1.Enabled = False:
Frame2.Enabled = True
S = S + 1
End Sub
Private Sub Command6_Click()
S = S + 1
BC_Beta.Show
End Sub

Private Sub Form_Load()
If Geo.Shape6.BackColor = &H8000& Then
   S = 0
   Pstn = 1
   With MSFlexGrid1
   .Rows = UBound(Clintrfc()) + 1
      .Col = 0
      For i = 1 To .Rows - 1
         .Row = i: .Text = "Spt-N #" & Str(i)
      Next
   .Cols = 12
   .Row = 0
   .Col = 1: .Text = "Layer #"
   .Col = 2: .Text = "Soil Type"
   .Col = 3: .Text = "Rpre. Depth"
   .Col = 4: .Text = "Point"
   .Col = 5: .Text = "Skin"
   .Col = 6: .Text = "Fi"
   .Col = 7: .Text = "Cu (KPa)"
   .Col = 8: .Text = "Ab (m2)"
   .Col = 9: .Text = "qf (kN/m2)"
   .Col = 10: .Text = "As (m2)"
   .Col = 11: .Text = "fs (kN/m2)"
   .ColWidth(0) = 1000
   .ColWidth(1) = 700
   .ColWidth(2) = 800
   .ColWidth(3) = 1000
   .ColWidth(4) = 1100
   .ColWidth(5) = 1100
   .ColWidth(6) = 800
   .ColWidth(7) = 800
   .ColWidth(8) = 800
   .ColWidth(9) = 1000
   .ColWidth(10) = 800
   .ColWidth(11) = 1000
   .FixedAlignment(1) = flexAlignCenterCenter
   .FixedAlignment(2) = flexAlignCenterCenter
   .FixedAlignment(3) = flexAlignCenterCenter
   .FixedAlignment(4) = flexAlignCenterCenter
   .FixedAlignment(5) = flexAlignCenterCenter
   .FixedAlignment(6) = flexAlignCenterCenter
   .FixedAlignment(7) = flexAlignCenterCenter
   .FixedAlignment(8) = flexAlignCenterCenter
   .FixedAlignment(9) = flexAlignCenterCenter
   .FixedAlignment(10) = flexAlignCenterCenter
   .FixedAlignment(11) = flexAlignCenterCenter
   .ColAlignment(0) = 1  'Give 1 for Left,4 for Center,7 for Right Alignment
   .ColAlignment(1) = 4
   .ColAlignment(2) = 4
   .ColAlignment(3) = 4
   .ColAlignment(4) = 4
   .ColAlignment(5) = 4
   .ColAlignment(6) = 4
   .ColAlignment(7) = 4
   .ColAlignment(8) = 4
   .ColAlignment(9) = 4
   .ColAlignment(10) = 4
   .ColAlignment(11) = 4
   End With
   k = 1
   ReDim CL(1 To UBound(Clintrfc()))
     For i = 1 To UBound(Clintrfc())
        If (i + 1) > UBound(Clintrfc()) Then Exit For
          If Clintrfc(i + 1) < Slintrfc(k) Then
            CL(i) = (Clintrfc(i + 1) - Clintrfc(i)) / 2 + Clintrfc(i)
          Else
            CL(i) = Slintrfc(k)
            k = k + 1
          End If
     Next
   CL(UBound(CL)) = SptnFdpth(UBound(SptnFdpth))
    k = 1
     For i = 1 To UBound(Clintrfc)
        If Clintrfc(i) <= Slintrfc(k) Then
              MSFlexGrid1.TextMatrix(i, 1) = "Layer " & k
        Else
           k = k + 1
              MSFlexGrid1.TextMatrix(i, 1) = "Layer " & k
        End If
     Next
   ReDim qF(1 To UBound(CrrctdSPTN()))
   ReDim Fs(1 To UBound(CrrctdSPTN()))
     For i = 1 To UBound(CrrctdSPTN())
        If PlBsShp = 1 Then
          MSFlexGrid1.TextMatrix(i, 8) = Format((((PlDmtr) ^ 2) * 3.1457 / 4), "###0.000")
        Else
          MSFlexGrid1.TextMatrix(i, 8) = Format((((PlDmtr) ^ 2)), "###0.000")
        End If
        MSFlexGrid1.TextMatrix(i, 3) = Str(Clintrfc(i))
     ReDim Preserve Area(UBound(Clintrfc))
     ReDim Preserve TotalArea(UBound(Clintrfc()))
```

```
    With MSFlexGrid1
      If PlBsShp = 1 Then
        If i = 1 Then
          .TextMatrix(i, 10) = Format(((3.1457 * PlDmtr) * Clintrfc(i)), "###0.000")
          Area(1) = .TextMatrix(i, 10)
        Else
          .TextMatrix(i, 10) = Format((3.1457 * PlDmtr) * (Clintrfc(i) - CL(i - 1)),
"###0.000")
          Area(i) = .TextMatrix(i, 10)
        End If
      Else
        If i = 1 Then
          .TextMatrix(i, 10) = Format(((4 * PlDmtr) * Clintrfc(i)), "###0.000")
          Area(1) = .TextMatrix(i, 10)
        Else
          .TextMatrix(i, 10) = Format((4 * PlDmtr) * (Clintrfc(i) - CL(i - 1)), "###0.000")
          Area(i) = .TextMatrix(i, 10)
        End If
      End If
      If PlBsShp = 1 Then
        If i = 1 Then
          TotalArea(1) = Format(((3.1457 * PlDmtr) * CL(i)), "###0.000")
        Else
          TotalArea(i) = Format((3.1457 * PlDmtr) * (CL(i) - CL(i - 1)), "###0.000")
        End If
      Else
        If i = 1 Then
          TotalArea(1) = Format(((4 * PlDmtr) * CL(i)), "###0.000")
        Else
          TotalArea(i) = Format((4 * PlDmtr) * (CL(i) - CL(i - 1)), "###0.000")
        End If
      End If
    End With
  Next
For k = 1 To UBound(Slintrfc())
  For i = 1 To UBound(Clintrfc())
    If MSFlexGrid1.TextMatrix(i, 2) = "" Then
      If Clintrfc(i) < Slintrfc(k) Then
        MSFlexGrid1.TextMatrix(i, 2) = FiSol(k)
      End If
    End If
  Next
Next
ReDim Fi(1 To UBound(CrrctdSPTN()))
For i = 1 To UBound(CrrctdSPTN())
  Fi(i) = (18 * (CrrctdSPTN(i) * EnrgyRt / 70)) ^ 0.5 + 15
  If MSFlexGrid1.TextMatrix(i, 2) = "Clay" Then
    MSFlexGrid1.TextMatrix(i, 6) = "-"
  Else
    MSFlexGrid1.TextMatrix(i, 6) = Format(Fi(i), "###0.00")
  End If
Next
ReDim Cu(1 To UBound(CrrctdSPTN()))
For i = 1 To UBound(CrrctdSPTN())
  Cu(i) = 12 * (CrrctdSPTN(i) * EnrgyRt / 70) / 2
  If MSFlexGrid1.TextMatrix(i, 2) = "Sand" Then
    MSFlexGrid1.TextMatrix(i, 7) = "-"
  Else
    MSFlexGrid1.TextMatrix(i, 7) = Format(Cu(i), "###0.00")
  End If
Next
  k = 1
  For i = 1 To UBound(Clintrfc)

    If Clintrfc(i) <= Slintrfc(k) Then
      If FiSol(k) = "Sand" Then
        For j = 1 To 11
          MSFlexGrid1.Row = i
          MSFlexGrid1.Col = j
          MSFlexGrid1.CellBackColor = QBColor(6)
        Next
      End If
      If FiSol(k) = "Clay" Then
        For j = 1 To 11
          MSFlexGrid1.Row = i
          MSFlexGrid1.Col = j
          MSFlexGrid1.CellBackColor = RGB(255, 125, 20)
        Next
      End If
    Else
      k = k + 1
      If FiSol(k) = "Sand" Then
        For j = 1 To 11
          MSFlexGrid1.Row = i
          MSFlexGrid1.Col = j
          MSFlexGrid1.CellBackColor = QBColor(6)
        Next
      End If
      If FiSol(k) = "Clay" Then
        For j = 1 To 11
          MSFlexGrid1.Row = i
          MSFlexGrid1.Col = j
          MSFlexGrid1.CellBackColor = RGB(255, 125, 20)
        Next
      End If
    End If
  Next
  Me.MSFlexGrid1.Col = 2
  For i = 1 To Me.MSFlexGrid1.Row - 1
    Me.MSFlexGrid1.Row = i
```

```
        If Me.MSFlexGrid1.TextMatrix(i, 2) = "Sand" Then
          If Me.MSFlexGrid1.CellBackColor = RGB(255, 125, 20) Then
            Msg = "The representavite depth of the sub layer number" + Str(i) + " exceeds
the main layer ending depth, inwhich it exist"
            MsgBox Msg, vbOKOnly, "Invalid Data Entry"
          End If
        Else
          If Me.MSFlexGrid1.CellBackColor = QBColor(6) Then
            Msg = "The representavite depth of the sub layer number" + Str(i) + " exceeds
the main layer ending depth, inwhich it exist"
            MsgBox Msg, vbOKOnly, "Invalid Data Entry"
          End If
        End If
      Next
      If MSFlexGrid1.TextMatrix(Pstn, 2) = "Sand" Then Frame2.Enabled = False
      If MSFlexGrid1.TextMatrix(Pstn, 2) = "Clay" Then Frame1.Enabled = False
End If
End Sub


Private Sub MSFlexGrid1_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As
Single)
    X = MSFlexGrid1.ColSel
    Y = MSFlexGrid1.RowSel
    If X = 6 Then
      If MSFlexGrid1.TextMatrix(Y, X) = "-" Then
      Else
        Bearing_Help.Label1.Caption = "Old Fi Value"
        Bearing_Help.Label2.Caption = "Enter New Fi"
        Bearing_Help.Label3.Caption = MSFlexGrid1.TextMatrix(Y, X)
        Bearing_Help.Show
      End If
    End If

    If X = 7 Then
      If MSFlexGrid1.TextMatrix(Y, X) = "-" Then
      Else
        Bearing_Help.Label1.Caption = "Old Cu Value"
        Bearing_Help.Label2.Caption = "Enter New Cu"
        Bearing_Help.Label3.Caption = MSFlexGrid1.TextMatrix(Y, X)
        Bearing_Help.Show
      End If
    End If

End Sub

Private Sub Command1_Click()
  With Bearing_Capacity.MSFlexGrid1
  If Text2 = "" Then
    Text2.SetFocus
  Else
    .TextMatrix(.RowSel, .ColSel) = Text2
    Me.Hide
    Text2 = ""
  End If
  End With
End Sub

Private Sub Text2_KeyUp(KeyCode As Integer, Shift As Integer)
  If KeyCode = vbKeyReturn Then
    Command1.SetFocus
  End If
  If KeyCode = 191 Then
    Uzunluk = Len(Text2)
    Text2 = Left$(Text2, Uzunluk - 1)
    Text2 = Text2 + Chr$(44)
    Text2 = Text2
  End If
End Sub

Private Sub Command1_Click()
  FldIdntEntry = True
  ReDim Slintrfc(MSFlexGrid1.Rows - 1)    'ok katman değişkenlerinin atanması
  ReDim FiInDpth(MSFlexGrid1.Rows - 1)
  ReDim FiSol(MSFlexGrid1.Rows - 1)
  ReDim FiDnsty(MSFlexGrid1.Rows - 1)
  For k = 1 To MSFlexGrid1.Rows - 1
    Slintrfc(k) = Val((MSFlexGrid1.TextMatrix(k, 2)))
  Next
  For k = 1 To MSFlexGrid1.Rows - 1
    FiInDpth(k) = Val((MSFlexGrid1.TextMatrix(k, 1)))
  Next
  For k = 1 To MSFlexGrid1.Rows - 1
    FiSol(k) = (MSFlexGrid1.TextMatrix(k, 3))
  Next
  For k = 1 To MSFlexGrid1.Rows - 1
    FiDnsty(k) = Val((MSFlexGrid1.TextMatrix(k, 4)))
  Next
  On Error GoTo ErrorHandler:
    If Slintrfc(1) Then
      Geo.Shape1.BackColor = &H8000&
        If Geo.Shape1.BackColor = &H8000& And Geo.Shape2.BackColor = &H8000&
And Geo.Shape3.BackColor = &H8000& And Geo.Shape4.BackColor = &H8000& Then
          Geo.Shape5.BackColor = &H8000&
        Else
          Geo.Shape5.BackColor = &H80&
        End If
      Unload Me
      Exit Sub
    End If
```

```
ErrorHandler:
        Geo.Shape1.BackColor = &H80&
          If Geo.Shape1.BackColor = &H8000& And Geo.Shape2.BackColor = &H8000&
And Geo.Shape3.BackColor = &H8000& And Geo.Shape4.BackColor = &H8000& Then
             Geo.Shape5.BackColor = &H8000&
          Else
             Geo.Shape5.BackColor = &H80&
          End If
End Sub

Private Sub Command2_Click()
   Unload Me
   Unload Field_Identification_Sub
End Sub

Private Sub Command3_Click()
   Field_Identification_Sub.Caption = "Add Clay Layer"
   Field_Identification_Sub.Show
   If MSFlexGrid1.Row = 0 Then
      Field_Identification_Sub.Text1.Text = "0"
      Field_Identification_Sub.Text2.SetFocus
   Else
      Field_Identification_Sub.Text1.Text =
Field_Identification.MSFlexGrid1.TextMatrix(Field_Identification.MSFlexGrid1.Rows - 1,
2)
      Field_Identification_Sub.Text2.SetFocus
   End If

End Sub

Private Sub Command4_Click()
   Field_Identification_Sub.Caption = "Add Sand Layer"
   Field_Identification_Sub.Show
   If MSFlexGrid1.Row = 0 Then
      Field_Identification_Sub.Text1.Text = "0"
      Field_Identification_Sub.Text2.SetFocus
   Else
      Field_Identification_Sub.Text1.Text =
Field_Identification.MSFlexGrid1.TextMatrix(Field_Identification.MSFlexGrid1.Rows - 1,
2)
      Field_Identification_Sub.Text2.SetFocus
   End If

End Sub

Private Sub Command5_Click()
   If Field_Identification.MSFlexGrid1.Rows <> 1 Then
      Field_Identification.MSFlexGrid1.Rows = Field_Identification.MSFlexGrid1.Rows - 1
   End If
End Sub
```

```
Private Sub Form_Load()
   With MSFlexGrid1
   .Rows = 1
   .Cols = 5
   .Row = 0
   .Col = 1: .Text = "Initial Depth (m)"
   .Col = 2: .Text = "Final Depth (m)"
   .Col = 3: .Text = "Soil Type"
   .Col = 4: .Text = "Unit Weight (KN/m^3)"

   .ColWidth(0) = 1000
   .ColWidth(1) = 1600
   .ColWidth(2) = 1600
   .ColWidth(3) = 1400
   .ColWidth(4) = 1800

   .FixedAlignment(1) = flexAlignCenterCenter
   .FixedAlignment(2) = flexAlignCenterCenter
   .FixedAlignment(3) = flexAlignCenterCenter
   .FixedAlignment(4) = flexAlignCenterCenter

   .ColAlignment(0) = 1  'Give 1 for Left,4 for Center,7 for Right Alignment
   .ColAlignment(1) = 4
   .ColAlignment(2) = 4
   .ColAlignment(3) = 4
   .ColAlignment(4) = 4
   End With



   If FldIdntEntry = True Then
      MSFlexGrid1.Rows = UBound(Slintrfc) + 1
      For i = 1 To UBound(Slintrfc)
         MSFlexGrid1.TextMatrix(i, 2) = Str(Slintrfc(i))
      Next
      For i = 1 To UBound(FiInDpth)
         MSFlexGrid1.TextMatrix(i, 1) = Str(FiInDpth(i))
      Next
      For i = 1 To UBound(FiSol)
         MSFlexGrid1.TextMatrix(i, 3) = FiSol(i)
      Next
      For i = 1 To UBound(FiDnsty())
         MSFlexGrid1.TextMatrix(i, 4) = Str(FiDnsty(i))
      Next

      For i = 1 To UBound(FiSol)
         If FiSol(i) = "Sand" Then
               MSFlexGrid1.Col = 3
```

```
            MSFlexGrid1.Row = i
            MSFlexGrid1.CellBackColor = QBColor(6)
        End If
        If FiSol(i) = "Clay" Then
            MSFlexGrid1.Col = 3
            MSFlexGrid1.Row = i
            MSFlexGrid1.CellBackColor = RGB(255, 125, 20)
        End If
      Next
      MSFlexGrid1.Col = 0
      For i = 1 To Field_Identification.MSFlexGrid1.Rows - 1
        MSFlexGrid1.Row = i
        MSFlexGrid1.Text = "Layer #" & Str(i)
      Next
    End If
End Sub

Private Sub Form_Unload(Cancel As Integer)
    Unload Field_Identification_Sub
End Sub

Private Sub Command1_Click()
If Field_Identification_Sub.Text1.Text = "" Or Field_Identification_Sub.Text2.Text = "" Or
Field_Identification_Sub.Text3.Text = "" Then
    Field_Identification_Sub.Show
    If Text1 = "" Then
        Text1.SetFocus
    ElseIf Text2 = "" Then
        Text2.SetFocus
    ElseIf Text3 = "" Then
        Text3.SetFocus
    End If
Else
    Field_Identification.MSFlexGrid1.Rows = Field_Identification.MSFlexGrid1.Rows + 1
    Field_Identification.MSFlexGrid1.Col = 0
    For i = 1 To Field_Identification.MSFlexGrid1.Rows - 1
        Field_Identification.MSFlexGrid1.Row = i
        Field_Identification.MSFlexGrid1.Text = "Layer #" & Str(i)
    Next
    Field_Identification.MSFlexGrid1.Col = 3
    If Me.Caption = "Add Sand Layer" Then
        Field_Identification.MSFlexGrid1.CellBackColor = QBColor(6)
        Field_Identification.MSFlexGrid1.Text = "Sand"
    Else
        Field_Identification.MSFlexGrid1.CellBackColor = RGB(255, 125, 20)
        Field_Identification.MSFlexGrid1.Text = "Clay"
    End If
    Field_Identification.MSFlexGrid1.TextMatrix(Field_Identification.MSFlexGrid1.Rows - 1,
1) = Text1.Text
    Field_Identification.MSFlexGrid1.TextMatrix(Field_Identification.MSFlexGrid1.Rows - 1,
2) = Text2.Text
    Field_Identification.MSFlexGrid1.TextMatrix(Field_Identification.MSFlexGrid1.Rows - 1,
4) = Text3.Text
    Text1.Text =
Field_Identification.MSFlexGrid1.TextMatrix(Field_Identification.MSFlexGrid1.Rows - 1,
2)
    Text2.Text = ""
    Text3.Text = ""
    Text2.SetFocus
End If
End Sub
Private Sub Text1_KeyDown(KeyCode As Integer, Shift As Integer)
    If KeyCode = vbKeyReturn Then
        Text2.SetFocus
    End If
End Sub
Private Sub Text2_KeyDown(KeyCode As Integer, Shift As Integer)
    If KeyCode = vbKeyReturn Then
        Text3.SetFocus
    End If
End Sub

Private Sub Text3_KeyDown(KeyCode As Integer, Shift As Integer)
    If KeyCode = vbKeyReturn Then
        Command1.SetFocus
    End If
End Sub

Dim FirstTurn As Boolean
Dim RfrncOvrrdnPrssr As Single
Private Sub A1_Click()
    Adjustment.Show 'menülerle formların gösterilmesi
End Sub

Private Sub C2_Click()
    Bearing_Capacity.Show 'menülerle formların gösterilmesi
End Sub
Private Sub Command1_Click()
    Field_Identification.Show
End Sub
Private Sub Command2_Click()
    Sptn_Values.Show
End Sub
Private Sub Command3_Click()
    GWT_Form.Show
End Sub
Private Sub Command4_Click()
    Pile_Properties.Show
End Sub
```

```
Private Sub Command5_Click()
    Adjustment.Show
End Sub
Private Sub Command6_Click()
    Bearing_Capacity.Show
End Sub
Private Sub Command7_Click()
  If Geo.Shape7.BackColor = &H80& Then
    Exit Sub
  End If
    Graph_Master.Show
End Sub
Private Sub Command8_Click()
    CommonDialog1.HelpFile = "VB5.HLP"
    CommonDialog1.HelpCommand = cdlHelpContents
    CommonDialog1.ShowHelp
End Sub
Private Sub Command9_Click()
If Geo.Shape9.BackColor = &H8000& Then
Printer.PaperSize = vbPRPSA4
Printer.FontSize = 5
Printer.Print "Soil Profile:"
Printer.Print "Initial Depth"; Tab(15); "Final Depth"; Tab(30); "Soil Type"; Tab(45); "Unit
Weight"
For i = 1 To UBound(Slintrfc())
  Printer.Print Tab(6); MSFlexGrid1.TextMatrix(3 + i, 0); Tab(19);
MSFlexGrid1.TextMatrix(3 + i, 1); Tab(32); MSFlexGrid1.TextMatrix(3 + i, 2); Tab(48);
MSFlexGrid1.TextMatrix(3 + i, 3)
Next
Printer.Print ""
Printer.Print "Ground Water Level="; MSFlexGrid1.TextMatrix(3 + UBound(Slintrfc()) + 2,
1) + " m."
Printer.Print "Pile Diameter="; MSFlexGrid1.TextMatrix(3 + UBound(Slintrfc()) + 2 + 2, 1)
+ " m."
Printer.Print "Pile Shape= "; MSFlexGrid1.TextMatrix(3 + UBound(Slintrfc()) + 2 + 3, 1)
Printer.Print "Installation Method= "; MSFlexGrid1.TextMatrix(3 + UBound(Slintrfc()) + 2 +
4, 1)
Printer.Print ""
Printer.Print "SPT:"
Printer.Print "Initial"; Tab(10); "Final"; Tab(17); "Repre. Depth"; Tab(30); "SPT-N"
For i = 1 To UBound(Clintrfc())
  Printer.Print Tab(2); MSFlexGrid1.TextMatrix(3 + UBound(Slintrfc()) + 2 + 4 + 3 + i, 0);
Tab(9); MSFlexGrid1.TextMatrix(3 + UBound(Slintrfc()) + 2 + 4 + 3 + i, 1); Tab(19);
MSFlexGrid1.TextMatrix(3 + UBound(Slintrfc()) + 2 + 4 + 3 + i, 2); Tab(31);
MSFlexGrid1.TextMatrix(3 + UBound(Slintrfc()) + 2 + 4 + 3 + i, 3)
Next
Printer.Print ""
Printer.Print "Energy Ratio= "; MSFlexGrid1.TextMatrix(3 + UBound(Slintrfc()) + 2 + 4 + 3
+ UBound(Clintrfc()) + 2, 1)
Printer.Print ""
Printer.Print "Calculation Results"
Printer.Print "Soil Type"; Tab(12); "Rep. Depth"; Tab(25); "Sig"; Tab(36); "Cn"; Tab(44); "C
SPT-N"; Tab(57); "Cal. Method"; Tab(77); "Fi"; Tab(85); "Cu"; Tab(98); "Qb"; Tab(106);
"C. Qb 1"; Tab(117); "C. Qb 2"; Tab(130); "Qs"; Tab(139); "Qu 1"; Tab(149); "Qu 2"
ok = UBound(Slintrfc())
For i = 1 To UBound(Clintrfc())
  With MSFlexGrid1
  Printer.Print Tab(3); .TextMatrix(ok + UBound(Clintrfc()) + 18 + i, 0); Tab(14);
.TextMatrix(ok + UBound(Clintrfc()) + 18 + i, 1); Tab(23); .TextMatrix(ok +
UBound(Clintrfc()) + 18 + i, 2); Tab(34); .TextMatrix(ok + UBound(Clintrfc()) + 18 + i, 3);
Tab(43); .TextMatrix(ok + UBound(Clintrfc()) + 18 + i, 4); Tab(56); .TextMatrix(ok +
UBound(Clintrfc()) + 18 + i, 5); Tab(75); .TextMatrix(ok + UBound(Clintrfc()) + 18 + i, 6);
Tab(85); .TextMatrix(ok + UBound(Clintrfc()) + 18 + i, 7); Tab(95); .TextMatrix(ok +
UBound(Clintrfc()) + 18 + i, 8); Tab(105); .TextMatrix(ok + UBound(Clintrfc()) + 18 + i, 9);
Tab(116); .TextMatrix(ok + UBound(Clintrfc()) + 18 + i, 10); Tab(127); .TextMatrix(ok +
UBound(Clintrfc()) + 18 + i, 11); Tab(137); .TextMatrix(ok + UBound(Clintrfc()) + 18 + i,
12); Tab(148); .TextMatrix(ok + UBound(Clintrfc()) + 18 + i, 13)
  End With
Next
Printer.EndDoc
Else
End If
End Sub

Private Sub E1_Click()
    Field_Identification.Show
End Sub

Private Sub E2_Click()
    Sptn_Values.Show
End Sub

Private Sub E3_Click()
    Gwt = Val(InputBox("Enter The GWT Level (m)", "Current GWT Level is" & Str(Gwt)
& " m.", Str(Gwt)))
End Sub



Private Sub E5_Click()
    Pile_Properties.Show
End Sub

Private Sub F1_Click()
    CommonDialog1.CancelError = True
    On Error GoTo errhandler
    CommonDialog1.Flags = cdlOFNHideReadOnly
    CommonDialog1.Filter = "Geo files|*.geo"
    CommonDialog1.ShowOpen
    Call Load_Geo(CommonDialog1.filename)
```

```
errhandler:
    Exit Sub
End Sub

Private Sub F2_Click()
    CommonDialog1.CancelError = True
    On Error GoTo errhandler
    Call Save_Geo
    CommonDialog1.Flags = cdlOFNHideReadOnly
    CommonDialog1.Filter = "Geo files|*.geo"
    CommonDialog1.ShowSave
    RichTextBox1.SaveFile CommonDialog1.filename, rtfRTF
errhandler:
    Exit Sub
End Sub

Private Sub F3_Click()
  End
End Sub

Private Sub Form_Unload(Cancel As Integer)
    End
End Sub

Private Sub G0_Click()
  Graph_Master.Show
End Sub

Private Sub H1_Click()
    CommonDialog1.HelpFile = "VB5.HLP"
    CommonDialog1.HelpCommand = cdlHelpContents
    CommonDialog1.ShowHelp
End Sub

Public Sub Load_Geo(GeoNm As String)
    MSFlexGrid1.Clear
    FldIdntEntry = True
    SptnVlsEntry = True
    FnClcltn = False
    Erase Slintrfc
    Erase Sptn
    Erase Clintrfc
    Erase FiInDpth
    Erase FiSol
    Erase FiDnsty
    Erase SptnIndpth
    Erase SptnFdpth
    RichTextBox1.LoadFile GeoNm
    Text1 = RichTextBox1.Text
    f = InStr(RichTextBox1.Text, "Field Identification Initial Depth:")
    f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    i = 0
    Do While Mid(RichTextBox1.Text, f + 1, 1) <> Chr(13)
      i = i + 1
      f = f + 1
      ReDim Preserve FiInDpth(1 To i)
      FiInDpth(i) = Val(Mid(RichTextBox1.Text, f, InStr(f, RichTextBox1.Text, Chr(13)) -
f))
      f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    Loop
    f = InStr(RichTextBox1.Text, "Soil Layer Interface:")
    f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    i = 0
    Do While Mid(RichTextBox1.Text, f + 1, 1) <> Chr(13)
      i = i + 1
      f = f + 1
      ReDim Preserve Slintrfc(1 To i)
      Slintrfc(i) = Val(Mid(RichTextBox1.Text, f, InStr(f, RichTextBox1.Text, Chr(13)) - f))
      f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    Loop
    f = InStr(RichTextBox1.Text, "Sort of Soil")
    f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    i = 0
    Do While Mid(RichTextBox1.Text, f + 1, 1) <> Chr(13)
      i = i + 1
      f = f + 1
      ReDim Preserve FiSol(1 To i)
      FiSol(i) = (Mid(RichTextBox1.Text, f, InStr(f, RichTextBox1.Text, Chr(13)) - f))
      f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    Loop
    f = InStr(RichTextBox1.Text, "Density of the layer:")
    f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    i = 0
    Do While Mid(RichTextBox1.Text, f + 1, 1) <> Chr(13)
      i = i + 1
      f = f + 1
      ReDim Preserve FiDnsty(1 To i)
      FiDnsty(i) = Val(Mid(RichTextBox1.Text, f, InStr(f, RichTextBox1.Text, Chr(13)) - f))
      f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    Loop
    f = InStr(RichTextBox1.Text, "Sptn Initial Depth:")
    f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    i = 0
    Do While Mid(RichTextBox1.Text, f + 1, 1) <> Chr(13)
      i = i + 1
      f = f + 1
      ReDim Preserve SptnIndpth(1 To i)
      SptnIndpth(i) = Val(Mid(RichTextBox1.Text, f, InStr(f, RichTextBox1.Text, Chr(13)) -
f))
      f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
```

```
    Loop
    f = InStr(RichTextBox1.Text, "Sptn Final Depth:")
    f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    i = 0
    Do While Mid(RichTextBox1.Text, f + 1, 1) <> Chr(13)
      i = i + 1
      f = f + 1
      ReDim Preserve SptnFdpth(1 To i)
      SptnFdpth(i) = Val(Mid(RichTextBox1.Text, f, InStr(f, RichTextBox1.Text, Chr(13)) -
f))
      f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    Loop
    f = InStr(RichTextBox1.Text, "Calculation Layer Interface:")
    f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    i = 0
    Do While Mid(RichTextBox1.Text, f + 1, 1) <> Chr(13)
      i = i + 1
      f = f + 1
      ReDim Preserve Clintrfc(1 To i)
      Clintrfc(i) = Val(Mid(RichTextBox1.Text, f, InStr(f, RichTextBox1.Text, Chr(13)) - f))
      f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    Loop
    f = InStr(RichTextBox1.Text, "Standart Penetration Test:")
    f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    i = 0
    Do While Mid(RichTextBox1.Text, f + 1, 1) <> Chr(13)
      i = i + 1
      f = f + 1
      ReDim Preserve Sptn(1 To i)
      Sptn(i) = Val(Mid(RichTextBox1.Text, f, InStr(f, RichTextBox1.Text, Chr(13)) - f))
      f = InStr(f, RichTextBox1.Text, Chr(13)) + 1
    Loop
    f = InStr(RichTextBox1.Text, "Ground Water Table:")
    f = InStr(f, RichTextBox1.Text, Chr(13)) + 2
    Gwt = Val(Mid(RichTextBox1.Text, f, InStr(f, RichTextBox1.Text, Chr(13)) - f))
    f = InStr(RichTextBox1.Text, "Pile Diameter:")
    f = InStr(f, RichTextBox1.Text, Chr(13)) + 2
    PlDmtr = Val(Mid(RichTextBox1.Text, f, InStr(f, RichTextBox1.Text, Chr(13)) - f))
    f = InStr(RichTextBox1.Text, "Energy Ratio:")
    f = InStr(f, RichTextBox1.Text, Chr(13)) + 2
    EnrgyRt = Val(Mid(RichTextBox1.Text, f, InStr(f, RichTextBox1.Text, Chr(13)) - f))
    f = InStr(RichTextBox1.Text, "PileBaseShape:")
    f = InStr(f, RichTextBox1.Text, Chr(13)) + 2
    PlBsShp = Val(Mid(RichTextBox1.Text, f, InStr(f, RichTextBox1.Text, Chr(13)) - f))
    f = InStr(RichTextBox1.Text, "Pile Installation Method:")
    f = InStr(f, RichTextBox1.Text, Chr(13)) + 2
    PlInstlltnMthd = Val(Mid(RichTextBox1.Text, f, InStr(f, RichTextBox1.Text, Chr(13)) -
f))
    EnrgyRtEntry = True
    PlDmtrEntry = True
    Shape1.BackColor = &H8000&
    Shape2.BackColor = &H8000&
    Shape3.BackColor = &H8000&
    Shape4.BackColor = &H8000&
    Shape5.BackColor = &H8000&
    Shape6.BackColor = &H80&
    Shape7.BackColor = &H80&
    Shape9.BackColor = &H80&
    MSFlexGrid1.Rows = UBound(Slintrfc()) + 2 * UBound(Clintrfc()) + 19
End Sub
Public Sub Save_Geo()
    RichTextBox1.Text = ""
    RichTextBox1.Text = "Field Identification Initial Depth:" + Chr(13)
    For i = 1 To UBound(FiInDpth())
      RichTextBox1.Text = RichTextBox1.Text + Str(FiInDpth(i)) + Chr(13)
    Next
    RichTextBox1.Text = RichTextBox1.Text + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + "Soil Layer Interface:" + Chr(13)
   For i = 1 To UBound(Slintrfc())
     RichTextBox1.Text = RichTextBox1.Text + Str(Slintrfc(i)) + Chr(13)
   Next
    RichTextBox1.Text = RichTextBox1.Text + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + "Sort of Soil:" + Chr(13)
   For i = 1 To UBound(FiSol())
     RichTextBox1.Text = RichTextBox1.Text + (FiSol(i)) + Chr(13)
   Next
    RichTextBox1.Text = RichTextBox1.Text + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + "Density of the layer:" + Chr(13)
   For i = 1 To UBound(FiDnsty())
     RichTextBox1.Text = RichTextBox1.Text + Str(FiDnsty(i)) + Chr(13)
   Next
    RichTextBox1.Text = RichTextBox1.Text + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + "Sptn Initial Depth:" + Chr(13)
   For i = 1 To UBound(SptnIndpth())
     RichTextBox1.Text = RichTextBox1.Text + Str(SptnIndpth(i)) + Chr(13)
   Next
    RichTextBox1.Text = RichTextBox1.Text + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + "Sptn Final Depth:" + Chr(13)
   For i = 1 To UBound(SptnFdpth())
     RichTextBox1.Text = RichTextBox1.Text + Str(SptnFdpth(i)) + Chr(13)
   Next
    RichTextBox1.Text = RichTextBox1.Text + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + "Calculation Layer Interface:" + Chr(13)
   For i = 1 To UBound(Clintrfc())
     RichTextBox1.Text = RichTextBox1.Text + Str(Clintrfc(i)) + Chr(13)
   Next
    RichTextBox1.Text = RichTextBox1.Text + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + "Standart Penetration Test:" + Chr(13)
   For i = 1 To UBound(Sptn())
     RichTextBox1.Text = RichTextBox1.Text + Str(Sptn(i)) + Chr(13)
```

```
    Next
    RichTextBox1.Text = RichTextBox1.Text + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + "Ground Water Table:" + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + Str(Gwt) + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + "Pile Diameter:" + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + Str(PlDmtr) + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + "Energy Ratio:" + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + Str(EnrgyRt) + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + "PileBaseShape:" + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + Str(PlBsShp) + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + "Pile Installation Method:" + Chr(13)
    RichTextBox1.Text = RichTextBox1.Text + Str(PlInstlltnMthd) + Chr(13)
End Sub

Private Sub Timer1_Timer()
On Error GoTo errhand
With MSFlexGrid1
  .ColWidth(0) = 1350
  .ColWidth(1) = 1150
  .ColWidth(2) = 950
  .ColWidth(3) = 1160
  .ColWidth(4) = 850
  .ColWidth(5) = 1100
  .ColWidth(6) = 530
  .ColWidth(7) = 530
  .ColWidth(8) = 700
  .ColWidth(9) = 800
  .ColWidth(10) = 800
  .ColWidth(11) = 800
  .ColWidth(12) = 800
  .ColWidth(13) = 800
  .FixedAlignment(1) = flexAlignCenterCenter
  .FixedAlignment(2) = flexAlignCenterCenter
  .FixedAlignment(3) = flexAlignCenterCenter
  .FixedAlignment(4) = flexAlignCenterCenter
  .FixedAlignment(5) = flexAlignCenterCenter
  .FixedAlignment(6) = flexAlignCenterCenter
  .FixedAlignment(7) = flexAlignCenterCenter
  .FixedAlignment(8) = flexAlignCenterCenter
  .FixedAlignment(9) = flexAlignCenterCenter
  .FixedAlignment(10) = flexAlignCenterCenter
  .FixedAlignment(11) = flexAlignCenterCenter
  .FixedAlignment(12) = flexAlignCenterCenter
  .FixedAlignment(13) = flexAlignCenterCenter
  .ColAlignment(0) = 4  'Give 1 for Left,4 for Center,7 for Right Alignment
  .ColAlignment(1) = 4
  .ColAlignment(2) = 4
  .ColAlignment(3) = 4
  .ColAlignment(4) = 4
  .ColAlignment(5) = 4
  .ColAlignment(6) = 4
  .ColAlignment(7) = 4
  .ColAlignment(8) = 4
  .ColAlignment(9) = 4
  .ColAlignment(10) = 4
  .ColAlignment(11) = 4
  .ColAlignment(12) = 4
  .ColAlignment(13) = 4
  .Col = 0
  .Row = 2
  .CellFontBold = True
  .CellAlignment = 1
  For i = 0 To 3
    .Row = 3
    .Col = i
    .CellFontBold = True
    .CellFontUnderline = True
  Next
  .Col = 0
  .Row = UBound(Slintrfc()) + 5
  .CellFontBold = True
  .CellAlignment = 7
  .Row = UBound(Slintrfc()) + 7
  .CellFontBold = True
  .CellAlignment = 7
  .Row = UBound(Slintrfc()) + 8
  .CellFontBold = True
  .CellAlignment = 7
  .Row = UBound(Slintrfc()) + 9
  .CellFontBold = True
  .CellAlignment = 7
  .Row = UBound(Slintrfc()) + 11
  .CellFontBold = True
  .CellAlignment = 1
  For i = 0 To 3
    .Col = i
    .Row = UBound(Slintrfc()) + 12
    .CellFontBold = True
    .CellFontUnderline = True
  Next
  .Col = 0 'ENERGY
  .Row = UBound(Slintrfc()) + 14 + UBound(Clintrfc())
  .CellFontBold = True
  .CellAlignment = 7
  .Row = UBound(Slintrfc()) + 16 + UBound(Clintrfc())
  .CellFontBold = True
  .CellAlignment = 7
  .Col = 1
  .Row = UBound(Slintrfc()) + 16 + UBound(Clintrfc())
```

```
    .CellFontBold = True
    .CellAlignment = 1
    For i = 0 To 13
      .Col = i
      .Row = UBound(Slintrfc()) + 17 + UBound(Clintrfc())
      .CellFontBold = True
    Next
If Slintrfc(1) Then
    .TextMatrix(2, 0) = "Soil Profile:"
    .TextMatrix(3, 0) = "Initial Depth:"
    .TextMatrix(3, 1) = "Final Depth:"
    .TextMatrix(3, 2) = "Soil Type:"
    .TextMatrix(3, 3) = "Unit Weight:"
    For i = 1 To UBound(Slintrfc())
    .TextMatrix(i + 3, 0) = Str(FiInDpth(i))
    .TextMatrix(i + 3, 1) = Str(Slintrfc(i))
    .TextMatrix(i + 3, 2) = (FiSol(i))
    .TextMatrix(i + 3, 3) = Str(FiDnsty(i))
    Next
      If Gwt Then
        .TextMatrix(UBound(Slintrfc()) + 5, 0) = "GWT Level:"
        .TextMatrix(UBound(Slintrfc()) + 5, 1) = Str(Gwt)
      End If
      If PlDmtr Then
        .TextMatrix(UBound(Slintrfc()) + 7, 0) = "Pile Diameter:"
        .TextMatrix(UBound(Slintrfc()) + 7, 1) = Str(PlDmtr)
        If PlBsShp = 1 Then 'circular
          .TextMatrix(UBound(Slintrfc()) + 8, 0) = "Pile Shape:"
          .TextMatrix(UBound(Slintrfc()) + 8, 1) = "Circular"
        Else ' square
          .TextMatrix(UBound(Slintrfc()) + 8, 0) = "Pile Shape:"
          .TextMatrix(UBound(Slintrfc()) + 8, 1) = "Square"
        End If
        If PlInstlltnMthd = 1 Then
          .TextMatrix(UBound(Slintrfc()) + 9, 0) = "Install Method:"
          .TextMatrix(UBound(Slintrfc()) + 9, 1) = "Driven"
        End If
        If PlInstlltnMthd = 2 Then
          .TextMatrix(UBound(Slintrfc()) + 9, 0) = "Install Method:"
          .TextMatrix(UBound(Slintrfc()) + 9, 1) = "Bored"
        End If
      End If
      If Sptn(1) Then
        .TextMatrix(UBound(Slintrfc()) + 11, 0) = "Spt-N:"
        .TextMatrix(UBound(Slintrfc()) + 12, 0) = "Initial"
        .TextMatrix(UBound(Slintrfc()) + 12, 1) = "Final"
        .TextMatrix(UBound(Slintrfc()) + 12, 2) = "Repre. D."
        .TextMatrix(UBound(Slintrfc()) + 12, 3) = "Spt-N Value"
        For i = 1 To UBound(Clintrfc())
          .TextMatrix((UBound(Slintrfc()) + 12 + i), 2) = Str(Clintrfc(i))
          .TextMatrix((UBound(Slintrfc()) + 12 + i), 3) = Str(Sptn(i))
        Next
        If FnClcltn = False Then
          For i = 1 To UBound(Clintrfc())
            .TextMatrix((UBound(Slintrfc()) + 12 + i), 0) = Str(Format((Clintrfc(i) - 0.3),
"###0.00"))
            .TextMatrix((UBound(Slintrfc()) + 12 + i), 1) = Str(Format((Clintrfc(i) +
0.15), "###0.00"))
          Next
        End If
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 14), 0) = "Energy Ratio:"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 14), 1) = Str(EnrgyRt)
      End If
      For i = 0 To 13
        .Row = (UBound(Slintrfc()) + UBound(Clintrfc()) + 17)
        .Col = i
        .CellFontUnderline = True
      Next
      If Qb(1) Then
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 16), 0) = "Calculation"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 16), 1) = "Table:"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 17), 0) = "Soil Type"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 17), 1) = "Repre. D."
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 17), 2) = "Sig Zero'"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 17), 3) = "Cn"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 17), 4) = "C. Spt-N"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 17), 5) = "Cal. Method"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 17), 6) = "Fi"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 17), 7) = "Cu"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 17), 8) = "Qb"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 17), 9) = "C. Qb 1"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 17), 10) = "C Qb 2"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 17), 11) = "Qs"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 17), 12) = "Qu 1"
        .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 17), 13) = "Qu 2"
        For i = 1 To UBound(Clintrfc())
          .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 0) = Soiltype(i)
          .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 1) =
Str(Format((Clintrfc(i)), "###0.00"))
          .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 2) =
Str(Format(SigZeroatRD(i), "###0.00"))
          .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 3) =
Str(Format(Cn(i), "###0.00"))
          .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 4) =
Str(Format(CrrctdSPTN(i), "###0.00"))
          .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 5) =
CalMethod(i)
          If .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 0) = "Sand"
Then
```

```
                .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 6) =
Str(Format(Fi(i), "###0.00"))
              Else
                .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 6) = "-"
              End If
              If .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 0) = "Clay"
Then
                .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 7) =
Str(Format(Cu(i), "###0.00"))
              Else
                .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 7) = "-"
              End If
              .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 8) =
Str(Format(Qb(i), "###0.00"))
              .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 9) =
Str(Format(Qbcr(i), "###0.00"))
              .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 10) =
Str(Format(Qbcr2(i), "###0.00"))
              .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 11) =
Str(Format(Qs(i), "###0.00"))
              .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 12) =
Str(Format((Qbcr(i) + Qs(i)), "###0.00"))
              .TextMatrix((UBound(Slintrfc()) + UBound(Clintrfc()) + 18) + i, 13) =
Str(Format((Qbcr2(i) + Qs(i)), "###0.00"))
            Next
          End If
End If
End With
errhand:
  Exit Sub
End Sub

Private Sub Toolbar1_ButtonClick(ByVal Button As ComctlLib.Button)
    If Button.Index = 1 Then
      CommonDialog1.CancelError = True
    On Error GoTo errhandler
    CommonDialog1.Flags = cdlOFNHideReadOnly
    CommonDialog1.Filter = "Geo files|*.geo"
    CommonDialog1.ShowOpen
    Call Load_Geo(CommonDialog1.filename)
errhandler:
    Exit Sub
    End If
    If Button.Index = 2 Then
      CommonDialog1.CancelError = True
    On Error GoTo errhandler
      Call Save_Geo
      CommonDialog1.Flags = cdlOFNHideReadOnly
      CommonDialog1.Filter = "Geo files|*.geo"
      CommonDialog1.ShowSave
      RichTextBox1.SaveFile CommonDialog1.filename, rtfRTF
    End If
End Sub


Dim ZoomVertical As Integer
Dim ZoomHorizontal As Integer
Dim P As Integer
Dim Kat As Integer
Dim MaxQu As Single
Private Sub B_Click()
  Check1.Visible = True
  Check2.Visible = True
  Check3.Visible = True
  Check4.Visible = True
  Check5.Visible = True
  Check6.Visible = True
  Check7.Visible = True
  Me.Cls
  ZoomVertical = (ScaleHeight - 20 / 100 * ScaleHeight) / Slintrfc(UBound(Slintrfc))
  ZoomHorizontal = (ScaleWidth - 60 / 100 * ScaleWidth) / 50
  FillStyle = vbFSTransparent
  ScaleMode = 3
  ForeColor = QBColor(0)
  CurrentX = ScaleHeight * 10 / 100
  CurrentY = ScaleWidth * 5 / 100
  For i = 1 To UBound(Slintrfc())
    FillColor = QBColor(i + 1)
    k = (i) Mod 2
    Line (CurrentX, CurrentY)-((ZoomHorizontal * 15 * k + (ScaleHeight * 10 / 100)),
((ScaleWidth * 5 / 100) + (ZoomVertical * Slintrfc(i)))), , B '(150 * k + 50)bir 200 bir 50 ,
(50(üstteki bölümü ekliyo) + (10(scale) * a(I)))), , B
  Next
  CurrentX = (ScaleHeight * 10 / 100) + (ScaleHeight * 8 / 100) - 47
  CurrentY = (ScaleWidth * 5 / 100) - 30
  Print "Soil Profile vs. Depth" ' label
  CurrentX = (ScaleHeight * 10 / 100) - 35
  CurrentY = (ScaleWidth * 5 / 100) - 7
  Print "0 m."
  For i = 1 To UBound(Slintrfc())
    CurrentX = (ScaleHeight * 10 / 100) - 35
    CurrentY = ((ScaleWidth * 5 / 100) - 7 + ZoomVertical * Slintrfc(i))
    Print Slintrfc(i) & " m."
  Next
  For i = 1 To UBound(Slintrfc()) ' sand clay yazdırma
    If i = 1 Then
      CurrentX = (ScaleHeight * 10 / 100) + 33
      CurrentY = ((ScaleWidth * 5 / 100) - 6 + ZoomVertical * Slintrfc(i) / 2)
      Print FiSol(i)
    Else
```

```
        CurrentX = (ScaleHeight * 10 / 100) + 33
        CurrentY = ((ScaleWidth * 5 / 100) - 6 + ZoomVertical * (Slintrfc(i) - (Slintrfc(i) -
Slintrfc(i - 1)) / 2))
        Print FiSol(i)
      End If
    Next
    CurrentX = (ScaleHeight * 10 / 100) - 15
    CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Slintrfc(UBound(Slintrfc)))
    Print "Depth"
    DrawWidth = 1                'GWT
    DrawStyle = 4
    CurrentX = (ScaleHeight * 10 / 100) - 15
    CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Gwt)
    Line (CurrentX, CurrentY)-((ZoomHorizontal * 5 * 1 + (ScaleHeight * 20 / 100)) + 15,
CurrentY)
    CurrentX = CurrentX + 5
    CurrentY = CurrentY - 5
    Print "GWT"
    DrawStyle = 0
    Kat = 0.0117442 * MaxQu - 1.07876547
    CurrentX = (ScaleHeight * 43 / 100)
    CurrentY = (ScaleWidth * 5 / 100)
    DrawWidth = 3                'kalın axlar
    Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + (ZoomHorizontal / Kat *
MaxQu)), (ScaleWidth * 5 / 100)) ' x axis
    Line ((ScaleHeight * 43 / 100), (ScaleWidth * 5 / 100))-((ScaleHeight * 43 / 100),
(((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(UBound(Clintrfc))))) ' y axis
    DrawWidth = 1
    CurrentX = (ScaleHeight * 43 / 100) + (ScaleHeight * 8 / 100) + 50
    CurrentY = (ScaleWidth * 5 / 100) - 30
    Print "Bearing Capacity vs. Depth"    ' label
    CurrentX = (ScaleHeight * 43 / 100) - 37            'origin label
    CurrentY = (ScaleWidth * 5 / 100) - 7
    Print "0 m."
    If Clintrfc(UBound(Clintrfc())) > 20 Then
      M = 2
    Else
      M = 1
    End If
    For i = 1 To Clintrfc(UBound(Clintrfc())) Step M   'axis label
      CurrentX = (ScaleHeight * 43 / 100) - 37
      CurrentY = ((ScaleWidth * 5 / 100) - 7 + ZoomVertical * i)
      Print i & " m."
    Next
    CurrentX = (ScaleHeight * 43 / 100) - 25 'sondaki label
    CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
    Print "Depth"
    CurrentX = ((ScaleHeight * 43 / 100) + 5 + ZoomHorizontal / Kat * MaxQu) 'sağdaki label
    CurrentY = (ScaleWidth * 5 / 100) - 5
    Print "Q"
    SkalaKat = (Val(MaxQu / 1000) + 1) * 100
    For i = 1 To 9
      CurrentX = ((ScaleHeight * 43 / 100) - 11.7 + ZoomHorizontal / Kat * (SkalaKat) * i)
      CurrentY = (ScaleWidth * 5 / 100) - 15
      Print Val((SkalaKat) * i)
    Next
End Sub
Private Sub Check1_Click()
If Check1.Value = 1 Then
    DrawWidth = 1
    DrawStyle = 0
    Me.ForeColor = QBColor(5)
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * MinQu(1))
    CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
    For i = 2 To UBound(Clintrfc)
      Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat *
MinQu(i)), ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
    Next
    DrawStyle = 0
    CurrentX = (ScaleHeight * 43 / 100) - 3 + ZoomHorizontal / Kat * MinQu(UBound(Qu1))
    CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
    Print "Min. Qu"
Else
    DrawWidth = 1
    DrawStyle = 0
    Me.ForeColor = RGB(255, 255, 255)
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * MinQu(1))
    CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
    For i = 2 To UBound(Clintrfc)
      Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat *
MinQu(i)), ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
    Next
    DrawStyle = 0
    CurrentX = (ScaleHeight * 43 / 100) - 3 + ZoomHorizontal / Kat * MinQu(UBound(Qu1))
    CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
    Print "Min Qu"
End If
End Sub

Private Sub Check2_Click()
If Check2.Value = 1 Then
    DrawWidth = 1
    DrawStyle = 0
    Me.ForeColor = QBColor(2)
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qbcr(1))
    CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
    For i = 2 To UBound(Clintrfc)
      Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat *
Qbcr(i)), ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
    Next
```

```
    DrawStyle = 0
    CurrentX = (ScaleHeight * 43 / 100) - 3 + ZoomHorizontal / Kat * Qbcr(UBound(Qbcr))
    CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
    Print "Qb c.1"
  Else
    DrawWidth = 1
    DrawStyle = 0
    Me.ForeColor = RGB(255, 255, 255)
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qbcr(1))
    CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
    For i = 2 To UBound(Clintrfc)
      Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat *
Qbcr(i)), ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
    Next
    DrawStyle = 0
    CurrentX = (ScaleHeight * 43 / 100) - 3 + ZoomHorizontal / Kat * Qbcr(UBound(Qbcr))
    CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
    Print "Qb c.1"
End If
End Sub

Private Sub Check3_Click()
If Check3.Value = 1 Then
    DrawWidth = 1
    DrawStyle = 0
    Me.ForeColor = QBColor(10)
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qbcr2(1))
    CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
    For i = 2 To UBound(Clintrfc)
      Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat *
Qbcr2(i)), ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
    Next
    DrawStyle = 0
    CurrentX = (ScaleHeight * 43 / 100) - 3 + ZoomHorizontal / Kat * Qbcr2(UBound(Qbcr2))
    CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
    Print "Qb c.2"
Else
    DrawWidth = 1
    DrawStyle = 0
    Me.ForeColor = RGB(255, 255, 255)
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qbcr2(1))
    CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
    For i = 2 To UBound(Clintrfc)
      Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat *
Qbcr2(i)), ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
    Next
    DrawStyle = 0
    CurrentX = (ScaleHeight * 43 / 100) - 3 + ZoomHorizontal / Kat * Qbcr2(UBound(Qbcr2))
    CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
    Print "Qb c.2"
End If
End Sub

Private Sub Check4_Click()
If Check4.Value = 1 Then
    DrawWidth = 1
    DrawStyle = 0
    Me.ForeColor = QBColor(9)
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qs(1))
    CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
    For i = 2 To UBound(Clintrfc)
      Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qs(i)),
((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
    Next
    DrawStyle = 0
    CurrentX = (ScaleHeight * 43 / 100) - 3 + ZoomHorizontal / Kat * Qs(UBound(Qs))
    CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
    Print "Qs"
Else
    DrawWidth = 1
    DrawStyle = 0
    Me.ForeColor = RGB(255, 255, 255)
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qs(1))
    CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
    For i = 2 To UBound(Clintrfc)
      Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qs(i)),
((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
    Next
    DrawStyle = 0
    CurrentX = (ScaleHeight * 43 / 100) - 3 + ZoomHorizontal / Kat * Qs(UBound(Qs))
    CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
    Print "Qs"
End If
End Sub

Private Sub Check5_Click()
If Check5.Value = 1 Then
    DrawWidth = 1
    DrawStyle = 0
    Me.ForeColor = QBColor(4)
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qu1(1))
    CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
    For i = 2 To UBound(Clintrfc)
      Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat *
Qu1(i)), ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
    Next
    DrawStyle = 0
    CurrentX = (ScaleHeight * 43 / 100) - 3 + ZoomHorizontal / Kat * Qu1(UBound(Qu1))
    CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
    Print "Qu 1"
```

```
Else
   DrawWidth = 1
   DrawStyle = 0
   Me.ForeColor = RGB(255, 255, 255)
   CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qu1(1))
   CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
   For i = 2 To UBound(Clintrfc)
      Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat *
Qu1(i)), ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
   Next
   DrawStyle = 0
   CurrentX = (ScaleHeight * 43 / 100) - 3 + ZoomHorizontal / Kat * Qu1(UBound(Qu1))
   CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
   Print "Qu 1"
End If
End Sub
Private Sub Check6_Click()
If Check6.Value = 1 Then
   DrawWidth = 1
   DrawStyle = 0
   Me.ForeColor = QBColor(12)
   CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qu2(1))
   CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
   For i = 2 To UBound(Clintrfc)
      Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat *
Qu2(i)), ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
   Next
   DrawStyle = 0
   CurrentX = (ScaleHeight * 43 / 100) - 3 + ZoomHorizontal / Kat * Qu2(UBound(Qu1))
   CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
   Print "Qu 2"
Else
   DrawWidth = 1
   DrawStyle = 0
   Me.ForeColor = RGB(255, 255, 255)
   CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qu2(1))
   CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
   For i = 2 To UBound(Clintrfc)
      Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat *
Qu2(i)), ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
   Next
   DrawStyle = 0
   CurrentX = (ScaleHeight * 43 / 100) - 3 + ZoomHorizontal / Kat * Qu2(UBound(Qu1))
   CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
   Print "Qu 2"
End If
End Sub

Private Sub Check7_Click()
 If Check7.Value = 1 Then
   DrawWidth = 1
   DrawStyle = 0
   Me.ForeColor = RGB(0, 0, 0)
   CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qb(1))
   CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
   For i = 2 To UBound(Clintrfc)
      Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qb(i)),
((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
   Next
Else
   DrawWidth = 1
   DrawStyle = 0
   Me.ForeColor = RGB(255, 255, 255)
   CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qb(1))
   CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
   For i = 2 To UBound(Clintrfc)
      Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / Kat * Qb(i)),
((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
   Next
End If
End Sub

Private Sub Form_Load()
Me.BackColor = &H80000005
MaxQu = Qu1(1)
For i = 1 To UBound(Qu1())
   If Qu1(i) > MaxQu Then MaxQu = Qu1(i)
Next
For i = 1 To UBound(Qu2())
   If Qu2(i) > MaxQu Then MaxQu = Qu2(i)
Next
End Sub

Private Sub Form_Resize()
   Me.Cls
End Sub
Private Sub P0_Click()
   Check1.Visible = False
   Check2.Visible = False
   Check3.Visible = False
   Check4.Visible = False
   Check5.Visible = False
   Check6.Visible = False
   Check7.Visible = False
   Me.Cls
   ZoomVertical = (ScaleHeight - 20 / 100 * ScaleHeight) / Slintrfc(UBound(Slintrfc))
   ZoomHorizontal = (ScaleWidth - 60 / 100 * ScaleWidth) / 50
   FillStyle = vbFSTransparent
   ScaleMode = 3
   ForeColor = QBColor(0)
```

```
  CurrentX = ScaleHeight * 10 / 100
  CurrentY = ScaleWidth * 5 / 100
  For i = 1 To UBound(Slintrfc())
     FillColor = QBColor(i + 1)
     k = (i) Mod 2
     Line (CurrentX, CurrentY)-((ZoomHorizontal * 15 * k + (ScaleHeight * 10 / 100)),
((ScaleWidth * 5 / 100) + (ZoomVertical * Slintrfc(i)))), , B
  Next
  CurrentX = (ScaleHeight * 10 / 100) + (ScaleHeight * 8 / 100) - 47
  CurrentY = (ScaleWidth * 5 / 100) - 30
  Print "Soil Profile vs. Depth" ' label
  CurrentX = (ScaleHeight * 10 / 100) - 35
  CurrentY = (ScaleWidth * 5 / 100) - 7
  Print "0 m."
  For i = 1 To UBound(Slintrfc())
     CurrentX = (ScaleHeight * 10 / 100) - 35
     CurrentY = ((ScaleWidth * 5 / 100) - 7 + ZoomVertical * Slintrfc(i))
     Print Slintrfc(i) & " m."
  Next
  For i = 1 To UBound(Slintrfc()) ' sand clay yazdırma
     If i = 1 Then
        CurrentX = (ScaleHeight * 10 / 100) + 33
        CurrentY = ((ScaleWidth * 5 / 100) - 6 + ZoomVertical * Slintrfc(i) / 2)
        Print FiSol(i)
     Else
        CurrentX = (ScaleHeight * 10 / 100) + 33
        CurrentY = ((ScaleWidth * 5 / 100) - 6 + ZoomVertical * (Slintrfc(i) - (Slintrfc(i) -
Slintrfc(i - 1)) / 2))
        Print FiSol(i)
     End If
  Next
  CurrentX = (ScaleHeight * 10 / 100) - 15
  CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Slintrfc(UBound(Slintrfc)))
  Print "Depth"
  DrawWidth = 1                'GWT
  DrawStyle = 4
  CurrentX = (ScaleHeight * 10 / 100) - 15
  CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Gwt)
  Line (CurrentX, CurrentY)-((ZoomHorizontal * 5 * 1 + (ScaleHeight * 20 / 100)) + 15,
CurrentY)
  CurrentX = CurrentX + 5
  CurrentY = CurrentY - 5
  Print "GWT"
  DrawStyle = 0
End Sub
Private Sub P1_Click()
  Printer.Orientation = 2
  Me.PrintForm
  End Sub
```

```
Private Sub S_Click()
  Check1.Visible = False
  Check2.Visible = False
  Check3.Visible = False
  Check4.Visible = False
  Check5.Visible = False
  Check6.Visible = False
  Check7.Visible = False
  Me.Cls
  ZoomVertical = (ScaleHeight - 20 / 100 * ScaleHeight) / Slintrfc(UBound(Slintrfc))
  ZoomHorizontal = (ScaleWidth - 60 / 100 * ScaleWidth) / 50
  FillStyle = vbFSTransparent
  ScaleMode = 3
  ForeColor = QBColor(0)
  CurrentX = ScaleHeight * 10 / 100
  CurrentY = ScaleWidth * 5 / 100
  For i = 1 To UBound(Slintrfc())
     FillColor = QBColor(i + 1)
     k = (i) Mod 2
     Line (CurrentX, CurrentY)-((ZoomHorizontal * 15 * k + (ScaleHeight * 10 / 100)),
((ScaleWidth * 5 / 100) + (ZoomVertical * Slintrfc(i)))), , B
  Next
  CurrentX = (ScaleHeight * 10 / 100) + (ScaleHeight * 8 / 100) - 47
  CurrentY = (ScaleWidth * 5 / 100) - 30
  Print "Soil Profile vs. Depth" ' label
  CurrentX = (ScaleHeight * 10 / 100) - 35
  CurrentY = (ScaleWidth * 5 / 100) - 7
  Print "0 m."
  For i = 1 To UBound(Slintrfc())
     CurrentX = (ScaleHeight * 10 / 100) - 35
     CurrentY = ((ScaleWidth * 5 / 100) - 7 + ZoomVertical * Slintrfc(i))
     Print Slintrfc(i) & " m."
  Next
  For i = 1 To UBound(Slintrfc()) ' sand clay yazdırma
     If i = 1 Then
        CurrentX = (ScaleHeight * 10 / 100) + 33
        CurrentY = ((ScaleWidth * 5 / 100) - 6 + ZoomVertical * Slintrfc(i) / 2)
        Print FiSol(i)
     Else
        CurrentX = (ScaleHeight * 10 / 100) + 33
        CurrentY = ((ScaleWidth * 5 / 100) - 6 + ZoomVertical * (Slintrfc(i) - (Slintrfc(i) -
Slintrfc(i - 1)) / 2))
        Print FiSol(i)
     End If
  Next
  CurrentX = (ScaleHeight * 10 / 100) - 15
  CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Slintrfc(UBound(Slintrfc)))
  Print "Depth"
  DrawWidth = 1                'GWT
  DrawStyle = 4
```

```
   CurrentX = (ScaleHeight * 10 / 100) - 15
   CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Gwt)
   Line (CurrentX, CurrentY)-((ZoomHorizontal * 5 * 1 + (ScaleHeight * 20 / 100)) + 15,
CurrentY)
   CurrentX = CurrentX + 5
   CurrentY = CurrentY - 5
   Print "GWT"
   DrawStyle = 0
   CurrentX = (ScaleHeight * 43 / 100)
   CurrentY = (ScaleWidth * 5 / 100)
   DrawWidth = 3                'kalın axlar
   Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + (ZoomHorizontal / 7 * 550)),
(ScaleWidth * 5 / 100)) ' x axis
   Line ((ScaleHeight * 43 / 100), (ScaleWidth * 5 / 100))-((ScaleHeight * 43 / 100),
(((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(UBound(Clintrfc))))) ' y axis
   DrawWidth = 1
CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * 0)
CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * 0)
For X = 0 To Clintrfc(UBound(Clintrfc())) Step 0.1
Sig = 0
   If X <= Gwt Then
      If X <= Slintrfc(1) Then
         Sig = X * FiDnsty(1)

      Else
         For P = 1 To UBound(Slintrfc())
            If X <= Slintrfc(P) Then
            current = P
            Exit For
            End If
         Next

         For w = (current - 1) To 1 Step -1
            If w = 1 Then
               B = 0
            Else
               B = Slintrfc(w - 1)
            End If
            Sig = Sig + (Slintrfc(w) - B) * FiDnsty(w)
         Next

         Sig = Sig + (X - Slintrfc(current - 1)) * FiDnsty(current)
      End If
   Else
      If Gwt <= Slintrfc(1) Then
         Sig = Gwt * FiDnsty(1)
      Else
         For P = 1 To UBound(Slintrfc())
            If Gwt <= Slintrfc(P) Then
            currentgwt = P
            Exit For
            End If
         Next
         For w = (currentgwt - 1) To 1 Step -1
            If w - 1 = 0 Then
               C = 0
            Else
               C = Slintrfc(w - 1)
            End If
            Sig = Sig + (Slintrfc(w) - C) * FiDnsty(w)
         Next
         Sig = Sig + (Gwt - Slintrfc(currentgwt - 1)) * FiDnsty(currentgwt)
      End If
      For P = 1 To UBound(Slintrfc())
         If Gwt <= Slintrfc(P) Then
            currentgwt = P
            Exit For
         End If
      Next
      If Gwt < X And X <= Slintrfc(currentgwt) Then
         Sig = Sig + (X - Gwt) * (FiDnsty(currentgwt) - 9.81)
      End If
      If X > Slintrfc(currentgwt) Then
         Sig = Sig + (Slintrfc(currentgwt) - Gwt) * (FiDnsty(currentgwt) - 9.81)
         For P = 1 To UBound(Slintrfc())
            If X <= Slintrfc(P) Then
               CurrentXvalue = P
               Exit For
            End If
         Next
         For w = (CurrentXvalue - 1) To currentgwt + 1 Step -1
            Sig = Sig + (Slintrfc(w) - Slintrfc(w - 1)) * (FiDnsty(w) - 9.81)
         Next
         Sig = Sig + (X - Slintrfc(CurrentXvalue - 1)) * (FiDnsty(CurrentXvalue) - 9.81)
      End If
   End If
Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * Sig),
((ScaleWidth * 5 / 100) + ZoomVertical * X))
Next
   CurrentX = (ScaleHeight * 43 / 100) + (ScaleHeight * 8 / 100) + 35
   CurrentY = (ScaleWidth * 5 / 100) - 30
   Print "Effective Sigma Zero  vs. Depth"    ' label
   CurrentX = (ScaleHeight * 43 / 100) - 37          'origin label
   CurrentY = (ScaleWidth * 5 / 100) - 7
   Print "0 m."
   If Clintrfc(UBound(Clintrfc())) > 20 Then
      M = 2
   Else
      M = 1
   End If
```

```
    For i = 1 To Clintrfc(UBound(Clintrfc())) Step M   'axis label
        CurrentX = (ScaleHeight * 43 / 100) - 37
        CurrentY = ((ScaleWidth * 5 / 100) - 7 + ZoomVertical * i)
        Print i & " m."
    Next
    CurrentX = (ScaleHeight * 43 / 100) - 15 'sondaki label
    CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
    Print "Depth"
    CurrentX = ((ScaleHeight * 43 / 100) - 45 + ZoomHorizontal / 7 * 550) - 30 'sağdaki label
    CurrentY = (ScaleWidth * 5 / 100) + 10
    Print "Sigma Zero (KN/m^2)"
CurrentX = (ScaleHeight * 43 / 100)
CurrentY = (ScaleWidth * 5 / 100)
For i = 0 To 5.5 Step 0.5
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * 100 * i)
    CurrentY = (ScaleWidth * 5 / 100)
    Line (CurrentX, CurrentY)-(CurrentX, CurrentY - 5)
Next
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * 0) - 3
    CurrentY = (ScaleWidth * 5 / 100) - 15
    Print "0"
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * 50) - 5
    CurrentY = (ScaleWidth * 5 / 100) - 15
    Print "50"
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * 100) - 9
    CurrentY = (ScaleWidth * 5 / 100) - 15
    Print "100"
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * 150) - 9
    CurrentY = (ScaleWidth * 5 / 100) - 15
    Print "150"
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * 200) - 9
    CurrentY = (ScaleWidth * 5 / 100) - 15
    Print "200"
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * 250) - 9
    CurrentY = (ScaleWidth * 5 / 100) - 15
    Print "250"
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * 300) - 9
    CurrentY = (ScaleWidth * 5 / 100) - 15
    Print "300"
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * 350) - 9
    CurrentY = (ScaleWidth * 5 / 100) - 15
    Print "350"
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * 400) - 9
    CurrentY = (ScaleWidth * 5 / 100) - 15
    Print "400"
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * 450) - 9
    CurrentY = (ScaleWidth * 5 / 100) - 15
    Print "450"
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * 500) - 9
    CurrentY = (ScaleWidth * 5 / 100) - 15
    Print "500"
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal / 7 * 550) - 9
    CurrentY = (ScaleWidth * 5 / 100) - 15
    Print "550"
End Sub

Private Sub SPT_Click()
    Check1.Visible = False
    Check2.Visible = False
    Check3.Visible = False
    Check4.Visible = False
    Check5.Visible = False
    Check6.Visible = False
    Check7.Visible = False
    Me.Cls
    ZoomVertical = (ScaleHeight - 20 / 100 * ScaleHeight) / Slintrfc(UBound(Slintrfc))
    ZoomHorizontal = (ScaleWidth - 60 / 100 * ScaleWidth) / 50
    FillStyle = vbFSTransparent
    ScaleMode = 3
    ForeColor = QBColor(0)
    CurrentX = ScaleHeight * 10 / 100
    CurrentY = ScaleWidth * 5 / 100
    For i = 1 To UBound(Slintrfc())
        FillColor = QBColor(i + 1)
        k = (i) Mod 2
        Line (CurrentX, CurrentY)-((ZoomHorizontal * 15 * k + (ScaleHeight * 10 / 100)),
((ScaleWidth * 5 / 100) + (ZoomVertical * Slintrfc(i)))), , B '(150 * k + 50)bir 200 bir 50 ,
(50(üstteki bölümü ekliyo) + (10(scale) * a(I)))), , B
    Next
    CurrentX = (ScaleHeight * 10 / 100) + (ScaleHeight * 8 / 100) - 47
    CurrentY = (ScaleWidth * 5 / 100) - 30
    Print "Soil Profile vs. Depth" ' label
    CurrentX = (ScaleHeight * 10 / 100) - 35
    CurrentY = (ScaleWidth * 5 / 100) - 7
    Print "0 m."
    For i = 1 To UBound(Slintrfc())
        CurrentX = (ScaleHeight * 10 / 100) - 35
        CurrentY = ((ScaleWidth * 5 / 100) - 7 + ZoomVertical * Slintrfc(i))
        Print Slintrfc(i) & " m."
    Next
    For i = 1 To UBound(Slintrfc()) ' sand clay yazdırma
        If i = 1 Then
            CurrentX = (ScaleHeight * 10 / 100) + 33
            CurrentY = ((ScaleWidth * 5 / 100) - 6 + ZoomVertical * Slintrfc(i) / 2)
            Print FiSol(i)
        Else
            CurrentX = (ScaleHeight * 10 / 100) + 33
            CurrentY = ((ScaleWidth * 5 / 100) - 6 + ZoomVertical * (Slintrfc(i) - (Slintrfc(i) -
Slintrfc(i - 1)) / 2))
            Print FiSol(i)
```

```
    End If
  Next
  CurrentX = (ScaleHeight * 10 / 100) - 15
  CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Slintrfc(UBound(Slintrfc)))
  Print "Depth"
  DrawWidth = 1                'GWT
  DrawStyle = 4
  CurrentX = (ScaleHeight * 10 / 100) - 15
  CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Gwt)
  Line (CurrentX, CurrentY)-((ZoomHorizontal * 5 * 1 + (ScaleHeight * 20 / 100)) + 15,
CurrentY)
  CurrentX = CurrentX + 5
  CurrentY = CurrentY - 5
  Print "GWT"
  DrawStyle = 0
  CurrentX = (ScaleHeight * 43 / 100)
  CurrentY = (ScaleWidth * 5 / 100)
  DrawWidth = 3                'kalın axlar
  Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + (ZoomHorizontal * 0.9 * 50)),
(ScaleWidth * 5 / 100)) ' x axis
  Line ((ScaleHeight * 43 / 100), (ScaleWidth * 5 / 100))-((ScaleHeight * 43 / 100),
(((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(UBound(Clintrfc))))) ' y axis
  DrawWidth = 1
  CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 * Sptn(1))
  CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
  For i = 2 To UBound(Clintrfc)
    Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 *
Sptn(i)), ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
  Next
  CurrentX = CurrentX - 4
  Print "N"
  CurrentX = (ScaleHeight * 43 / 100) + (ScaleHeight * 8 / 100) + 55
  CurrentY = (ScaleWidth * 5 / 100) - 30
  Print "SPT Blows vs. Depth"    ' label
  CurrentX = (ScaleHeight * 43 / 100) - 37          'origin label
  CurrentY = (ScaleWidth * 5 / 100) - 7
  Print "0 m."
  If Clintrfc(UBound(Clintrfc())) > 20 Then
    M = 2
  Else
    M = 1
  End If
  For i = 1 To Clintrfc(UBound(Clintrfc())) Step M   'axis label
    CurrentX = (ScaleHeight * 43 / 100) - 37
    CurrentY = ((ScaleWidth * 5 / 100) - 7 + ZoomVertical * i)
    Print i & " m."
  Next
  CurrentX = (ScaleHeight * 43 / 100) - 15 'sondaki label
  CurrentY = ((ScaleWidth * 5 / 100) + 5 + ZoomVertical * Clintrfc(UBound(Clintrfc)))
  Print "Depth"
  CurrentX = ((ScaleHeight * 43 / 100) + 5 + ZoomHorizontal * 0.9 * 50) 'sağdaki label
  CurrentY = (ScaleWidth * 5 / 100) - 5
  Print "N"
  Me.ForeColor = QBColor(9)
  DrawWidth = 1                'Csptn grafiği
  DrawStyle = 0
  CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 * CrrctdSPTN(1))
  CurrentY = ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(1))
  For i = 2 To UBound(Clintrfc)
    Line (CurrentX, CurrentY)-(((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 *
CrrctdSPTN(i)), ((ScaleWidth * 5 / 100) + ZoomVertical * Clintrfc(i)))
  Next
  CurrentX = CurrentX - 28
  Print "Corrected N"
Me.ForeColor = QBColor(0)
CurrentX = (ScaleHeight * 43 / 100)
CurrentY = (ScaleWidth * 5 / 100)
For i = 0 To 50 Step 5
  CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 * i)
  CurrentY = (ScaleWidth * 5 / 100)
  Line (CurrentX, CurrentY)-(CurrentX, CurrentY - 5)
Next
  CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 * 0) - 3
  CurrentY = (ScaleWidth * 5 / 100) - 15
  Print "0"
  CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 * 5) - 3
  CurrentY = (ScaleWidth * 5 / 100) - 15
  Print "5"
  CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 * 10) - 5
  CurrentY = (ScaleWidth * 5 / 100) - 15
  Print "10"
  CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 * 15) - 5
  CurrentY = (ScaleWidth * 5 / 100) - 15
  Print "15"
  CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 * 20) - 5
  CurrentY = (ScaleWidth * 5 / 100) - 15
  Print "20"
  CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 * 25) - 5
  CurrentY = (ScaleWidth * 5 / 100) - 15
  Print "25"
  CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 * 30) - 5
  CurrentY = (ScaleWidth * 5 / 100) - 15
  Print "30"
  CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 * 35) - 5
  CurrentY = (ScaleWidth * 5 / 100) - 15
  Print "35"
  CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 * 40) - 5
  CurrentY = (ScaleWidth * 5 / 100) - 15
  Print "40"
  CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 * 45) - 5
```

```
    CurrentY = (ScaleWidth * 5 / 100) - 15
    Print "45"
    CurrentX = ((ScaleHeight * 43 / 100) + ZoomHorizontal * 0.9 * 50) - 5
    CurrentY = (ScaleWidth * 5 / 100) - 15
    Print "50"
End Sub

Private Sub Command1_Click()
If Text1.Text = "" Then
   Text1.SetFocus
Else
   Gwt = Val(Text1)
   Geo.Shape3.BackColor = &H8000&
        If Geo.Shape1.BackColor = &H8000& And Geo.Shape2.BackColor = &H8000& And
Geo.Shape3.BackColor = &H8000& And Geo.Shape4.BackColor = &H8000& Then
            Geo.Shape5.BackColor = &H8000&
        Else
            Geo.Shape5.BackColor = &H80&
        End If
        Unload Me
End If
End Sub

Private Sub Command2_Click()
   Unload Me
End Sub
Private Sub Text1_KeyDown(KeyCode As Integer, Shift As Integer)
   If KeyCode = vbKeyReturn Then
      Command1.SetFocus
   End If
End Sub

Private Sub Command1_Click()
If Text1 <> "" Or Text2 <> "" Then
   If Option3.Value = True Or Option4.Value = True Then
   If Option1.Value = True Then
        PlDmtr = Val(Text1)
        PlBsShp = 1
      End If
      If Option2.Value = True Then
        PlDmtr = Val(Text2)
        PlBsShp = 2
      End If
      If Option3.Value = True Then PlInstlltnMthd = 1
      If Option4.Value = True Then PlInstlltnMthd = 2
   PlDmtrEntry = True
   On Error GoTo ErrorHandler:
      If PlDmtr Then
        Geo.Shape4.BackColor = &H8000&
        If Geo.Shape1.BackColor = &H8000& And Geo.Shape2.BackColor = &H8000&
And Geo.Shape3.BackColor = &H8000& And Geo.Shape4.BackColor = &H8000& Then
            Geo.Shape5.BackColor = &H8000&
        Else
            Geo.Shape5.BackColor = &H80&
        End If

        Unload Me
        Exit Sub
      End If
ErrorHandler:
        Geo.Shape4.BackColor = &H80&
        If Geo.Shape1.BackColor = &H8000& And Geo.Shape2.BackColor = &H8000&
And Geo.Shape3.BackColor = &H8000& And Geo.Shape4.BackColor = &H8000& Then
            Geo.Shape5.BackColor = &H8000&
        Else
            Geo.Shape5.BackColor = &H80&
        End If
   Unload Me
   End If
End If
End Sub

Private Sub Command2_Click()
   Unload Me
End Sub

Private Sub Form_Load()
   If PlDmtrEntry = True Then
      If PlBsShp = 1 Then
        Option1.Value = True
        Text1 = Str(PlDmtr)
      End If
      If PlBsShp = 2 Then
        Option2.Value = True
        Text2 = Str(PlDmtr)
      End If
      If PlInstlltnMthd = 1 Then Option3.Value = True
      If PlInstlltnMthd = 2 Then Option4.Value = True
   End If
End Sub

Private Sub Command1_Click()
If MSFlexGrid1.Rows = 1 Then
Else
   If Text1 = "" Then
      Dim Msg, Style, Title, Ctxt, Response, MyString
      Msg = "Please enter a Energy Ratio"   ' Define message.
      Style = vbOKOnly ' Define buttons.
      Title = "Energy Ratio?"  ' Define title.
```

```
    Ctxt = 1000 ' Define topic
    ok = MsgBox(Msg, Style, Title, Help, Ctxt)
    Text1.SetFocus
    GoTo Shortcut
  Else
    EnrgyRtEntry = True
    EnrgyRt = Val(Text1)
  End If
  SptnVlsEntry = True
  Dim InflncDpthErr As Boolean
  Dim InflncDpthErrExstnc As Boolean
  ReDim Clintrfc(MSFlexGrid1.Rows - 1)
  MSFlexGrid1.Col = 3
    For i = 1 To MSFlexGrid1.Rows - 1
      MSFlexGrid1.Row = i
      Clintrfc(i) = Val(MSFlexGrid1.Text)
    Next
    If Clintrfc(1) = 0 Then
      M = MsgBox(" SPT-N Values is invalid!", vbCritical, "Invalid Data Entry")
     If M = 1 Then
       InflncDpthErr = True
     End If
    End If
    For i = 2 To MSFlexGrid1.Rows - 1
      If Clintrfc(i) < Clintrfc(i - 1) Then
        For j = i To MSFlexGrid1.Rows - 1
          If Clintrfc(j) <> 0 Then
            InflncDpthErrExstnc = True
          End If
        Next
        If InflncDpthErrExstnc = True Then
          M = MsgBox(Str(i) & ". SPT-N Value is invalid!", vbCritical, "Invalid Data Entry")
         If M = 1 Then
           InflncDpthErr = True
         End If
          InflncDpthErrExstnc = False
        End If
      End If
    Next
    If InflncDpthErr = True Then
      M = MsgBox("Please, Correct the SPT-N Value Error(s)!", vbCritical, "Invalid Data
Entry")
    Else
      For i = 1 To MSFlexGrid1.Rows - 1
        If Clintrfc(i) = 0 Then
          i = i - 1
          Exit For
        End If
      Next
  If Clintrfc(MSFlexGrid1.Rows - 1) <> 0 Then i = i - 1
  ReDim Preserve Clintrfc(i)
  ReDim SptnIndpth(1 To i)
  ReDim SptnFdpth(1 To i)
  For k = 1 To i
    SptnIndpth(k) = Val(MSFlexGrid1.TextMatrix(k, 1))
  Next
  For k = 1 To i
    SptnFdpth(k) = Val(MSFlexGrid1.TextMatrix(k, 2))
  Next
  For i = 1 To UBound(Clintrfc())
    If Val(MSFlexGrid1.TextMatrix(i, 4)) = 0 Then
      InflncDpthErr = True
      M = MsgBox(Str(i) & ". SPTN Value is invalid!", vbCritical, "Invalid Data Entry")
    End If
  Next
  For i = UBound(Clintrfc()) + 1 To MSFlexGrid1.Rows - 1
    If MSFlexGrid1.TextMatrix(i, 4) <> "" Then
      InflncDpthErr = True
      M = MsgBox(Str(i) & ". SPTN Value is invalid!", vbCritical, "Invalid Data Entry")
    End If
  Next
  ReDim Sptn(MSFlexGrid1.Rows - 1)
  MSFlexGrid1.Col = 4
    For i = 1 To MSFlexGrid1.Rows - 1
      MSFlexGrid1.Row = i
      Sptn(i) = Val(MSFlexGrid1.Text)
    Next
    For i = 1 To MSFlexGrid1.Rows - 1
      If Sptn(i) = 0 Then
        i = i - 1
        Exit For
      End If
    Next
  If Sptn(MSFlexGrid1.Rows - 1) <> 0 Then i = i - 1 'ok
    ReDim Preserve Sptn(i)
    If InflncDpthErr = True Then
    M = MsgBox("Please, Correct the SPT-N Value Error(s)!", vbCritical, "Invalid Data
Entry")
  Else
    Unload Me
  End If
End If
  On Error GoTo ErrorHandler:
    If Clintrfc(1) Then
      Geo.Shape2.BackColor = &H8000&
        If Geo.Shape1.BackColor = &H8000& And Geo.Shape2.BackColor = &H8000&
And Geo.Shape3.BackColor = &H8000& And Geo.Shape4.BackColor = &H8000& Then
          Geo.Shape5.BackColor = &H8000&
        Else
          Geo.Shape5.BackColor = &H80&
```

```
            End If
          Unload Me
          Exit Sub
        End If
ErrorHandler:
          Geo.Shape2.BackColor = &H80&
            If Geo.Shape1.BackColor = &H8000& And Geo.Shape2.BackColor = &H8000&
And Geo.Shape3.BackColor = &H8000& And Geo.Shape4.BackColor = &H8000& Then
              Geo.Shape5.BackColor = &H8000&
            Else
              Geo.Shape5.BackColor = &H80&
            End If
Shortcut:
End If
End Sub

Private Sub Command2_Click()
  Unload Me
  Unload Sptn_Values2
End Sub

Private Sub Command3_Click()
  Sptn_Values2.Show
  If Sptn_Values.MSFlexGrid1.Rows - 1 = 0 Then
    Sptn_Values2.Text1.Text = ""
    Sptn_Values2.Text2.Text = ""
    Sptn_Values2.Text1.SetFocus
  Else
    Sptn_Values2.Text1.Text =
Str(Val(Sptn_Values.MSFlexGrid1.TextMatrix(Sptn_Values.MSFlexGrid1.Rows - 1, 1)) +
1.5)
    Sptn_Values2.Text2.Text = ""
    Sptn_Values2.Text1.SetFocus
    'Sptn_Values2.Text1.
  End If
End Sub

Private Sub Command4_Click()
  If MSFlexGrid1.Rows <> 1 Then
    MSFlexGrid1.Rows = MSFlexGrid1.Rows - 1
  End If
End Sub

Private Sub Command5_Click()
Geo.Shape6.BackColor = &H80&
Geo.Shape7.BackColor = &H80&
FnClcItn = True
  k = 1
  ReDim CL(1 To UBound(Clintrfc()))
    For i = 1 To UBound(Clintrfc())
      If (i + 1) > UBound(Clintrfc()) Then Exit For
        If Clintrfc(i + 1) < Slintrfc(k) Then
          CL(i) = (Clintrfc(i + 1) - Clintrfc(i)) / 2 + Clintrfc(i)
        Else
          CL(i) = Slintrfc(k)

          k = k + 1
        End If
    Next
CL(UBound(CL)) = SptnFdpth(UBound(SptnFdpth))
ReDim Sptn(MSFlexGrid1.Rows - 1)
  MSFlexGrid1.Col = 4
  For i = 1 To MSFlexGrid1.Rows - 1
    MSFlexGrid1.Row = i
    Sptn(i) = Val(MSFlexGrid1.Text)
  Next
  For i = 1 To MSFlexGrid1.Rows - 1
    If Sptn(i) = 0 Then
      i = i - 1
      Exit For
    End If
  Next
If Sptn(MSFlexGrid1.Rows - 1) <> 0 Then i = i - 1
  ReDim Preserve Sptn(i)

ReDim FnRpDpth(1 To 500)
FnRpDpth(1) = 0.125
k = 1
For i = 2 To UBound(FnRpDpth())
  FnRpDpth(i) = FnRpDpth(i - 1) + 0.25

  If CL(UBound(CL())) - FnRpDpth(i) < 0.25 Then Exit For
  If (CL(k) - (FnRpDpth(i))) < 0.25 Then

    FnRpDpth(i + 1) = FnRpDpth(i) + ((CL(k) - (FnRpDpth(i))) / 2)
    FnRpDpth(i + 2) = CL(k) + 0.125
    i = i + 2
    k = k + 1
  End If
Next
ReDim Preserve FnRpDpth(1 To (i))
ReDim FnSptn(1 To UBound(FnRpDpth()))
k = 1
For i = 1 To UBound(FnRpDpth())
  If FnRpDpth(i) <= CL(k) Then
    FnSptn(i) = Sptn(k)
  Else
    k = k + 1
    FnSptn(i) = Sptn(k)
  End If
```

```
    Next
ReDim Clintrfc(1 To UBound(FnRpDpth()))
ReDim Sptn(1 To UBound(FnRpDpth()))
For i = 1 To UBound(FnRpDpth())
    Clintrfc(i) = FnRpDpth(i)
    Sptn(i) = FnSptn(i)
Next
Geo.MSFlexGrid1.Clear
Geo.MSFlexGrid1.Rows = UBound(Slintrfc()) + 2 * UBound(Clintrfc()) + 19
Unload Me
End Sub

Private Sub Form_Load()
    With MSFlexGrid1
    .Rows = 1
    .Cols = 5
    .Row = 0
    .Col = 1: .Text = "Initial Depth (m)"
    .Col = 2: .Text = "Final Depth (m)"
    .Col = 3: .Text = "Representative Depth"
    .Col = 4: .Text = "SPT-N Value"
    .ColWidth(0) = 1100
    .ColWidth(1) = 1700
    .ColWidth(2) = 1700
    .ColWidth(3) = 2000
    .ColWidth(4) = 1500
    .FixedAlignment(1) = flexAlignCenterCenter
    .FixedAlignment(2) = flexAlignCenterCenter
    .FixedAlignment(3) = flexAlignCenterCenter
    .FixedAlignment(4) = flexAlignCenterCenter
    .ColAlignment(0) = 1  'Give 1 for Left,4 for Center,7 for Right Alignment
    .ColAlignment(1) = 4
    .ColAlignment(2) = 4
    .ColAlignment(3) = 4
    .ColAlignment(4) = 4
    End With
    If EnrgyRtEntry = True Then
        Text1 = EnrgyRt
    End If
    If SptnVlsEntry = True Then
        MSFlexGrid1.Rows = UBound(Clintrfc) + 1
        For i = 1 To UBound(Clintrfc)
            MSFlexGrid1.TextMatrix(i, 0) = "SPT-N #" & Str(i)
            MSFlexGrid1.TextMatrix(i, 3) = Str(Clintrfc(i))
            MSFlexGrid1.TextMatrix(i, 4) = Str(Sptn(i))
            If FnClcItn = False Then
                MSFlexGrid1.TextMatrix(i, 1) = Str(SptnIndpth(i))
                MSFlexGrid1.TextMatrix(i, 2) = Str(SptnFdpth(i))
            Else
            End If
        Next
        If FnClcItn = True Then  'detailed mod için açıklama
            Dim Msg, Style, Title, Ctxt, Response, MyString
            Msg = "In Detailed Calculation Mode Spt-N Initial and Final Depths are not
avaible."   ' Define message.
            Style = vbOKOnly ' Define buttons.
            Title = "Program Information"   ' Define title.
            Ctxt = 1000 ' Define topic
            ok = MsgBox(Msg, Style, Title, Help, Ctxt)
        End If
    k = 1
        For i = 1 To UBound(Clintrfc)
            If Clintrfc(i) <= Slintrfc(k) Then
                If FiSol(k) = "Sand" Then
                    For j = 1 To 4
                        MSFlexGrid1.Row = i
                        MSFlexGrid1.Col = j
                        MSFlexGrid1.CellBackColor = QBColor(6)
                    Next
                End If
                If FiSol(k) = "Clay" Then
                    For j = 1 To 4
                        MSFlexGrid1.Row = i
                        MSFlexGrid1.Col = j
                        MSFlexGrid1.CellBackColor = RGB(255, 125, 20)
                    Next
                End If
            Else
                k = k + 1
                If FiSol(k) = "Sand" Then
                    For j = 1 To 4
                        MSFlexGrid1.Row = i
                        MSFlexGrid1.Col = j
                        MSFlexGrid1.CellBackColor = QBColor(6)
                    Next
                End If
                If FiSol(k) = "Clay" Then
                    For j = 1 To 4
                        MSFlexGrid1.Row = i
                        MSFlexGrid1.Col = j
                        MSFlexGrid1.CellBackColor = RGB(255, 125, 20)
                    Next
                End If
            End If
        Next
    End If
End Sub
Private Sub Form_Unload(Cancel As Integer)
    Unload Sptn_Values2
End Sub
```

```
Private Sub Command1_Click()
If Text1.Text = "" Or Text2.Text = "" Then
   Me.Show
   If Text1 = "" Then
      Text1.SetFocus
   ElseIf Text2 = "" Then
      Text2.SetFocus
   End If
Else
   If FldIdntEntry = True Then
      If (Val(Text1.Text) + 0.3) > Slintrfc(UBound(Slintrfc())) Then
         MsgBox "The SPT Representative Depth should be less then the final of the soil
profile", vbOKOnly, "Invalid Data Entry"
         GoTo ok:
      End If
   Else
         MsgBox "The last SPT Representative Depth should be iess then the final of the soil
profile", vbOKOnly, "Invalid Data Entry"
   End If
   Sptn_Values.MSFlexGrid1.Rows = Sptn_Values.MSFlexGrid1.Rows + 1
   Sptn_Values.MSFlexGrid1.Col = 0
   For i = 1 To Sptn_Values.MSFlexGrid1.Rows - 1
      Sptn_Values.MSFlexGrid1.Row = i
      Sptn_Values.MSFlexGrid1.Text = "SPT-N #" & Str(i)
   Next
   Sptn_Values.MSFlexGrid1.TextMatrix(Sptn_Values.MSFlexGrid1.Rows - 1, 1) =
Text1.Text
   Sptn_Values.MSFlexGrid1.TextMatrix(Sptn_Values.MSFlexGrid1.Rows - 1, 2) =
Str(Val(Text1.Text) + 0.45)
   Sptn_Values.MSFlexGrid1.TextMatrix(Sptn_Values.MSFlexGrid1.Rows - 1, 3) =
Str(Val(Text1.Text) + 0.3)
   Sptn_Values.MSFlexGrid1.TextMatrix(Sptn_Values.MSFlexGrid1.Rows - 1, 4) =
Text2.Text
   Text2.Text = ""
   Text1.SetFocus
   Text1.Text =
Str(Val(Sptn_Values.MSFlexGrid1.TextMatrix(Sptn_Values.MSFlexGrid1.Rows - 1, 1)) +
1.5)
End If
ok:
End Sub
Private Sub Text1_KeyDown(KeyCode As Integer, Shift As Integer)
   If KeyCode = vbKeyReturn Then
      Text2.SetFocus
   End If
End Sub
Private Sub Text2_KeyDown(KeyCode As Integer, Shift As Integer)
   If KeyCode = vbKeyReturn Then
      Command1.SetFocus
   End If
End Sub
Global Gwt As Single
Global PlDmtr As Single
Global Slintrfc() As Single
Global Sptn() As Integer
Global Clintrfc() As Single
Global FiInDpth() As Single
Global FiSol() As String
Global FiDnsty() As Single
Global SptnIndpth() As Single
Global SptnFdpth() As Single
Global FldIdntEntry As Boolean
Global SptnVlsEntry As Boolean
Global CL() As Single
Global SigZeroatRD() As Single
Global Cn() As Single
Global N() As Single
Global N2() As Single
Global CrrctdSPTN() As Single
Global Fs() As Single
Global qF() As Single
Global Fi() As Single
Global Cu() As Single
Global EnrgyRt As Integer
Global EnrgyRtEntry As Boolean
Global PlDmtrEntry As Boolean
Global PlBsAr As Single
Global PlBsShp As Integer
Global FnClcltn As Boolean
Global FnRpDpth() As Single
Global FnSptn() As Single
Global Ip_Beta As Single
Global Ocr_Beta As Single
Global Pstn As Integer
Global Qb() As Single
Global Qs() As Single
Global Qbcr() As Single
Global Qbcr2() As Single
Global MinQb() As Single
Global Soiltype() As String
Global Qu1() As Single
Global Qu2() As Single
Global MinQu() As Single
Global CalMethod() As String
Global Area() As Single
Global TotalArea() As Single
Global PlInstlltnMthd As Integer
Global UpD As Integer
Global DownD As Integer
```

APPENDIX C User's Manuel

C.1 Installation and Execution of the Computer Program

The computer program can be installed by clicking the setup icon from the program installation folder. A standard windows installation program guide is provided throughout the setup process (Figure C.1).

In this section and in Section 4.4 (sample section), the figures, which are the forms of the program, have been printed as black and white for convenience of presentation, although they are colored on the screen.

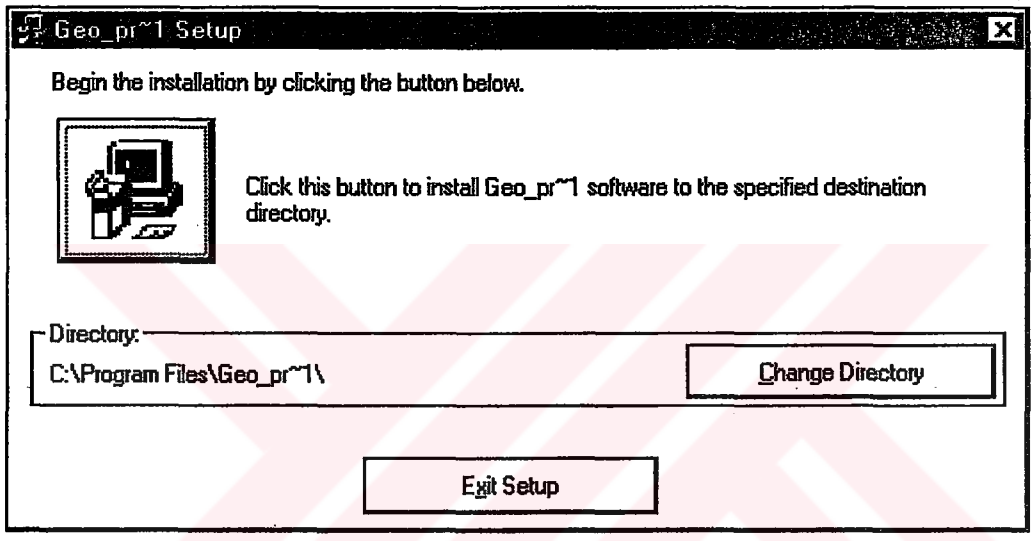


Figure C. 1 Program installation form

In the program installation form (Figure C.1), the directory in which the installation is performed can be changed. Program installation begins by clicking the 'Install' button. After a few seconds a form is displayed which informs that the program installation is done successfully.

The program can be executed by pressing the 'Start' button, programs section, and the icon that refers to the Geo program.

To uninstall the program, the user should press the 'Start' button, and select 'Control Panel' from the 'Settings' menu. In the following window, 'Add & Remove Programs' icon should be clicked. A window is displayed in which the

programs are listed which can be uninstalled. By selecting the 'Geo Program' icon and pressing the 'Add & Remove' button, uninstallation is executed.

C.2 The Overall Appearance of the Main Form

In the main form, shown in Figure C.2, the input and calculation forms can be accessed with the help of some buttons and pull-down menus. Buttons and pull-down menus have the same function, but buttons have been designed for quick access to the forms. Buttons in the tools frame have been separated into two parts. The first upper four buttons are for inputting, and the other two lower buttons are for calculation.

All buttons on that form have colored frames in order to display to the user whether the parameters in that form have been entered or the calculations have been done or not. If the necessary parameters are entered for the input buttons, the color of the button's frame turns to green. To be ready for the calculations, these four buttons should be green. Otherwise calculations will not be performed.

The calculation buttons, located under input buttons, can be accessed only if the color of their frames is green. These buttons should be pressed in the order of their placement, because the program needs the results of the first form in order to calculate the others. Otherwise no calculation is performed.

The main form has a grid object for displaying the inputs and the outputs. The grid object is filled following inputting or calculation phases.

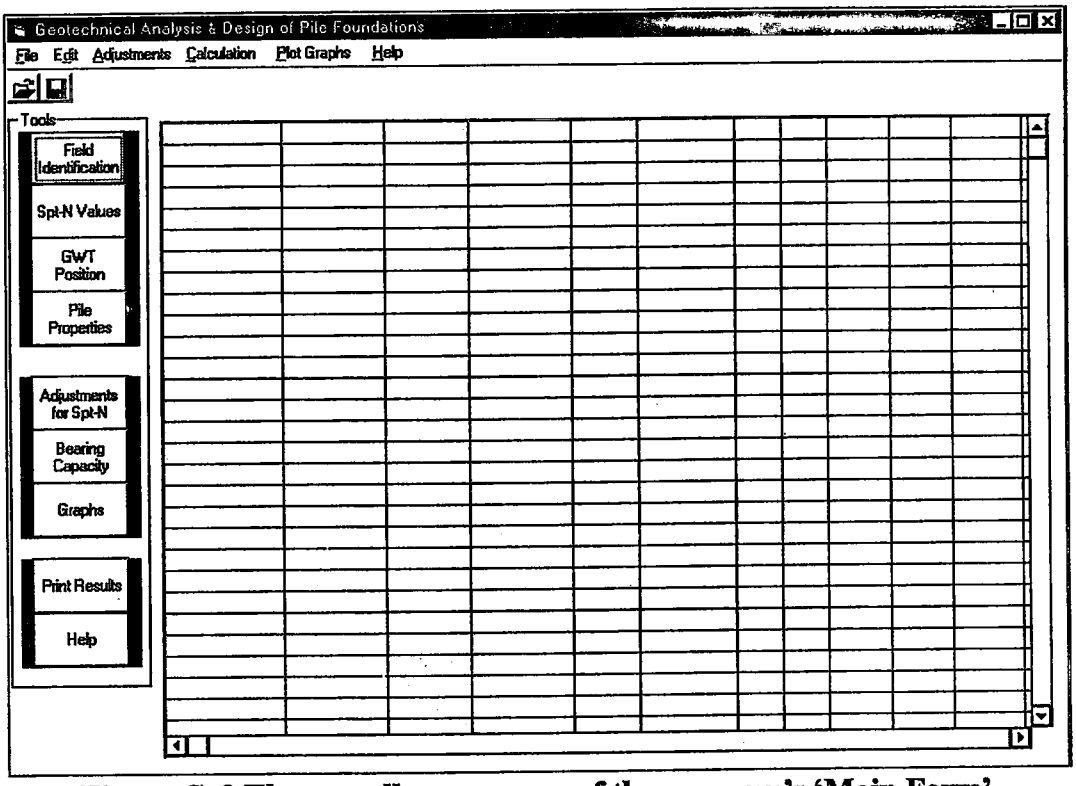


Figure C. 2 The overall appearance of the program's 'Main Form'.

C.3 Inputting

Inputting operation can be done either loading an existing ".geo" file or entering the parameters manually.

C.3.1 Loading a Geo File

The program is capable of running data files in the hard drive. User can load these files into the program. The files, created by the program, have '.geo' extension. '.geo' files are actually text files. They may be edited with a text editor and can be changed and saved again.

To load a ' .geo' file, 'load geo file' command should be selected from the file pull-down menu or the 'load' button at the toolbar should be clicked. A standard load window is displayed which is a common 'Windows' application. In this window only '.geo' files are shown. Double clicking a file or pressing load button after selecting a '.geo' file will load the file.

After loading a '.geo' file, this file can be edited with the program and saved again. At this stage the program is ready to execute the calculations.

C.3.2 Manuel Data Entry

C.3.2.1 Soil Profile

Pressing the 'Field Identification' button in the main form provides an access to the form shown in Figure C.3. This form is designed for entering the soil profile characteristics. To add the desired layer, the user should click the 'Add Clay Layer' or 'Add Sand Layer' buttons. By pressing these buttons, a new form is shown on which initial depth, final depth and unit weight of the layer is asked. Properties of the newly added layer are immediately inserted in the grid object of the 'Field Identification Form'. The clay layers are shown as orange and sand layers are shown as green to distinguish the soil profile easily.

In entering the properties of a subsequent layer, the initial depth is filled automatically as the final depth of the previous layer. The 'Remove Last Layer' button can be used to delete the last layer.

Once the soil profile is created successfully, the 'Ok' button on the 'Field Identification Form' should be clicked to store the parameters in the memory.

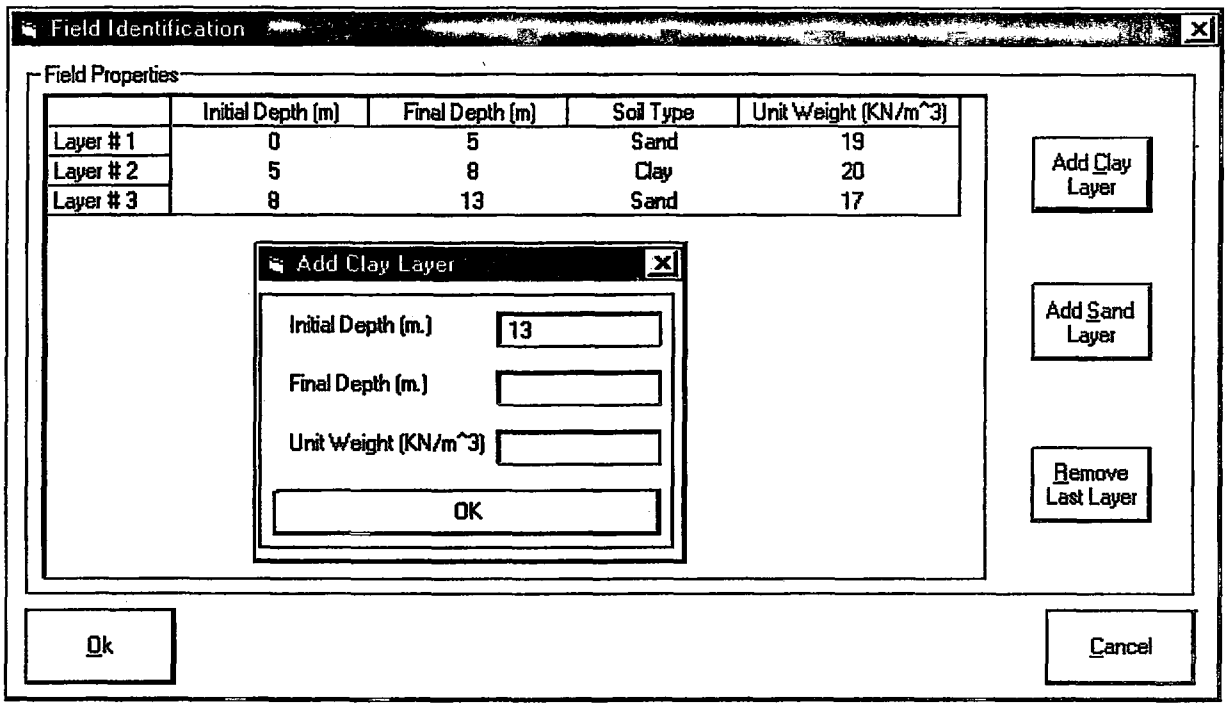


Figure C. 3 'Field Identification' form and its sub form.

C.3.2.2 SPT Values

Pressing the 'SPT-N Values' button in the 'Main Form' or selecting the same command from the pulldown menu provides access to the 'SPT-Properties' form, which is shown in Figure C.4. This form is designed to enter the SPT data values. By clicking the 'Add SPT Data' button, a sub form will be shown on which the initial depth and SPT-N Value is required. To approve the entered values, the 'Ok' button should be clicked. Once the first SPT data entered, the initial depth section of the next SPT data is filled automatically. If the entered value is not correct, simply entering a new value will delete the previous one.

The 'Remove SPT Data' button should be pressed to correct a SPT data, which is entered before.

After all values have been entered, the Detailed Calculation button can be pressed in order to subdivide the sub layers into 25 cm thick pieces. This operation provides more detailed results as already mentioned in the preceding sections.

The energy ratio of the SPT equipment under consideration is entered in percent in the 'Energy Ratio' frame of this form.

To approve the values that have been recently entered, 'Ok' button on the 'SPT Properties' form should be pressed.

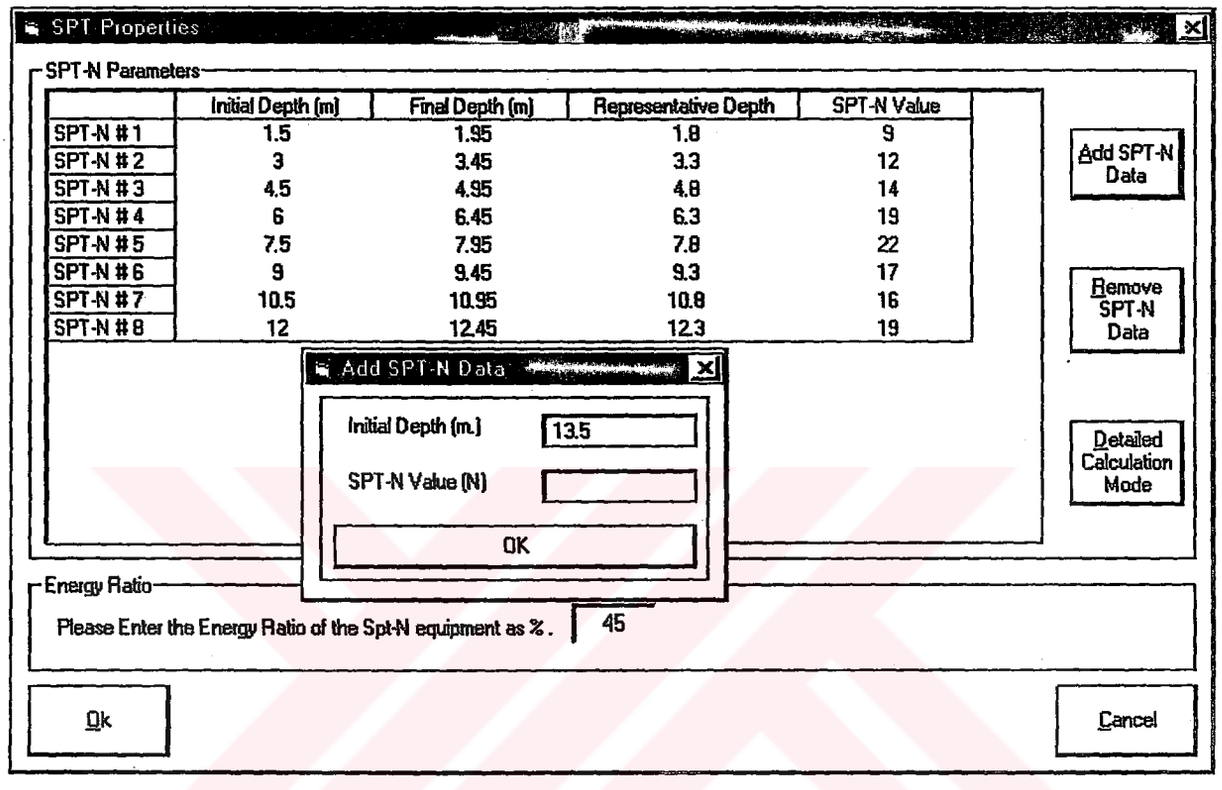


Figure C. 4 The 'SPT Properties' form and its sub form.

C.3.2.3 Ground Water Table

Information for the ground water table level is entered with the 'Ground Water Table Level' form shown in Figure C.5. Clicking the 'GWT Position' button in the 'Main Form' or selecting it from the pull down menu activates this form. Entering a value, and pressing 'Ok' will store the value in the memory.

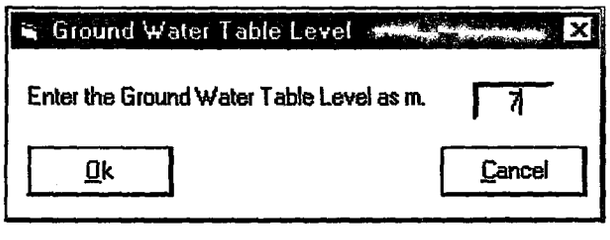


Figure C. 5 Ground water level entering form.

C.3.2.4 Pile Properties

'Pile Properties' form can be accessed by pressing the 'Pile Properties' button on the 'Main Form' or selecting it from the pull-down menu. Pile sectional characteristics and installation method is entered in this form as displayed in Figure C.6. In the 'Pile Section' frame, two radio buttons are available for piles having circular or square cross section. After selecting the 'Section', the diameter or width of the pile should be entered in meters in the corresponding text box.

In the 'Installation' section, selecting the radio buttons as driven or bored enters pile installation method.

After all parameters are entered, 'Ok' button should be pressed in order to store them in the memory.

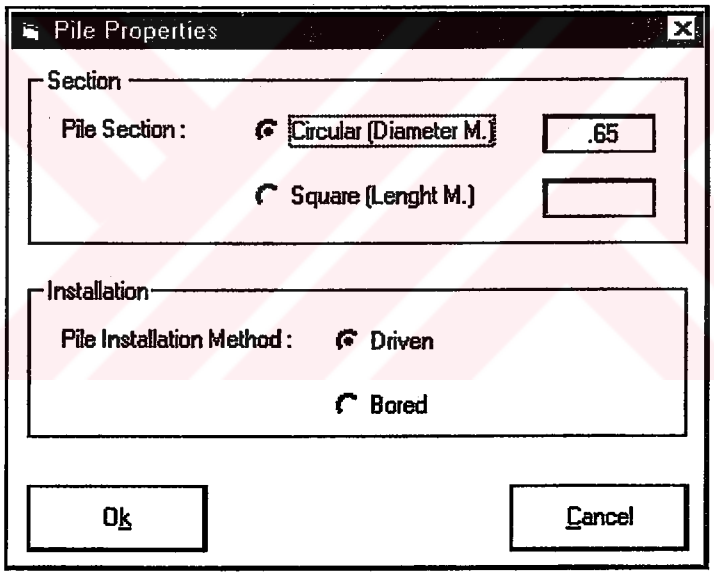
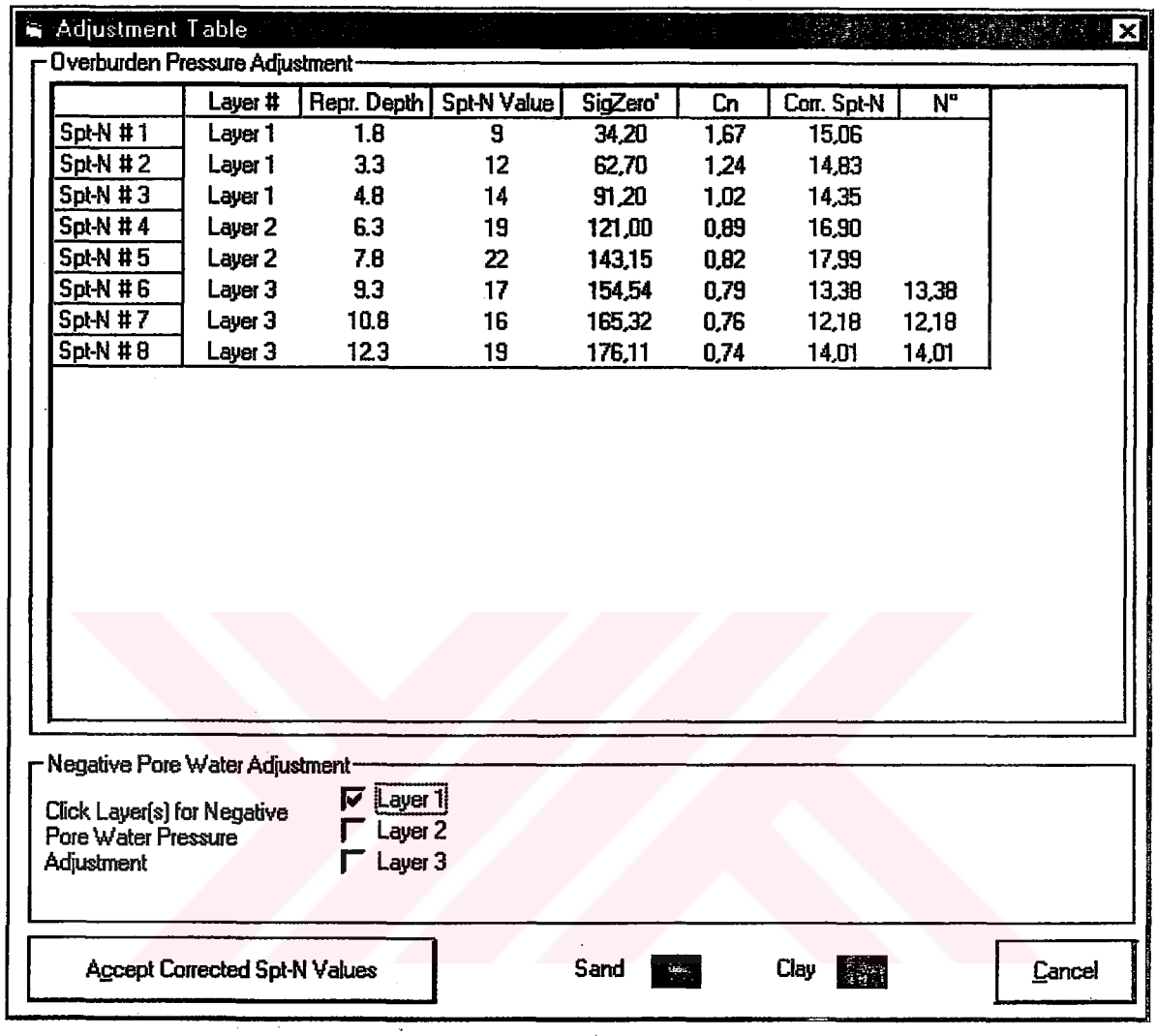


Figure C. 6 'Pile Properties' form

C.4 Adjustment for SPT-N

Pressing the 'Adjustment Table' button, only if the color of its frame is green that means all inputs are entered can access the form shown in Figure B.7. In this form, there are the layer numbers, representative depths, SPT-N Values, effective overburden pressures at representative depths, C_N values, corrected SPT-N values. If required, negative pore water pressure adjustment on SPT-N can be made for

each main layer by pressing the check boxes. If 'Accept Corrected SPT-N' button is pressed, the corrected SPT-N values will be stored in the memory.



	Layer #	Repr. Depth	Spt-N Value	SigZero'	Cn	Corr. Spt-N	N"
Spt-N # 1	Layer 1	1.8	9	34,20	1,67	15,06	
Spt-N # 2	Layer 1	3.3	12	62,70	1,24	14,83	
Spt-N # 3	Layer 1	4.8	14	91,20	1,02	14,35	
Spt-N # 4	Layer 2	6.3	19	121,00	0,89	16,90	
Spt-N # 5	Layer 2	7.8	22	143,15	0,82	17,99	
Spt-N # 6	Layer 3	9.3	17	154,54	0,79	13,38	13,38
Spt-N # 7	Layer 3	10.8	16	165,32	0,76	12,18	12,18
Spt-N # 8	Layer 3	12.3	19	176,11	0,74	14,01	14,01

Figure C. 7 Adjustment Table for SPT-N Values

C.5 Bearing Capacity Calculation

By pressing the 'Bearing Capacity Calculation' button, the form shown in Figure C.8 is displayed. Pressing the buttons on the two different calculation method frames provides the execution of the calculations. The clay layer calculation method frame is unavailable in case of a sand layer, and vice versa. There are two base and skin resistance calculation methods for a sand layer, and two skin resistance calculation methods for a clay layer. In order to execute the desired method for any layer, the corresponding button should be clicked.

If the user wants to change the ϕ' and c_u values, which are estimated by the program, the desired cell should be clicked. This will display a new form on the screen. The new value is entered in this form as displayed in Figure C.8. This operation must be performed before the bearing capacity calculation; otherwise the program will process existing values.

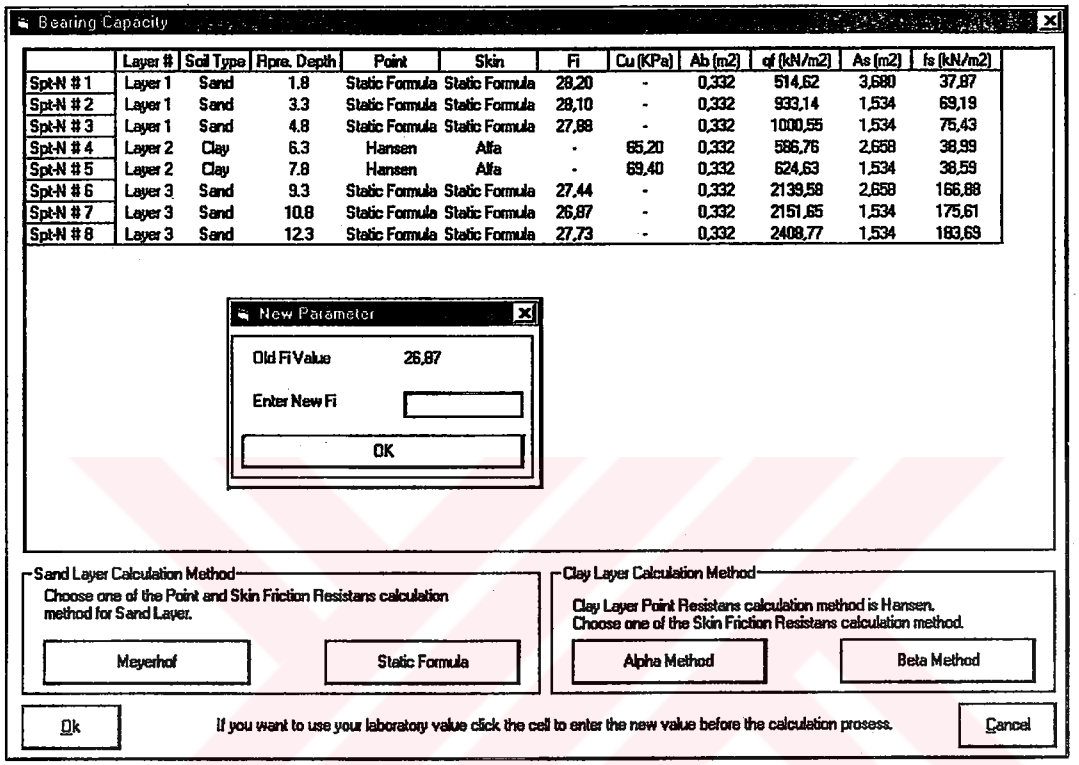


Figure C. 8 'Bearing Capacity Calculation' form and its sub form.

C.6 Base Resistance Correction

'Base Resistance Correction Form' is displayed automatically after pressing 'Ok' button in the 'Bearing Capacity Form'. Before displaying this form, two message boxes will ask the boundaries for average base resistance calculation. In this form the base resistance correction will be automatically made with Average and Meyerhof's methods. The corrected base resistances and the smaller of these can be seen in the grid object of the 'Base Resistance Correction Form'. Pressing 'Ok' button will store the results in the memory.

C.7 Graphs

After all calculations have been done, the resulting graphs can be plotted by pressing 'Graphs' button. Only if the bearing capacity calculation has been done, the frame around the button will turn to green, which means it is available for plotting the graphs.

By pressing the 'Graph' button, a new form is displayed. Soil profile will be shown fixed for all at the left side of the screen. Effective overburden pressure, raw and corrected SPT-N values, and bearing capacity versus depth graphs can be plotted by selecting the desired command from the menu.

In the bearing capacity versus depth graph, the base resistance, corrected base resistances with two correction methods, skin friction, and ultimate bearing capacity of the pile can be plotted by selecting the corresponding boxes.

After plotting the graphs, it can be sent to printer by clicking the 'Print Page' button in the menu.

C.8 Printing Results

Clicking the print results button provides the printing of the results. A printer, attached to the system, will print the results page. This results page will contain all the data in the 'Grid Object' of the 'Main Form'.

APPENDIX D Sample Application

D.1 Problem Description

To show the usage of the computer program prepared in this study, a sample problem has been processed. Sample problem has been selected as simple as possible, The detailed calculation mode has not been performed in order to be able to show all the results in figures, in which the forms of the program has been displayed.

The soil profile in the sample problem consists of three layers. A clay layer lies between two sand layers. Upper sand layer extends down to 5 m; the thickness of the clay layer is 3 m. The ground water table lies at the depth of 7 m. Totally 8 Standard Penetration Test results are given down to 12.45 m depth,

The pile is a constant cross section circular driven pile with a diameter of 0.65 m.

The main layers, unit weights, Standard Penetration Test initial and final depths, SPT-N values, and ground water table level is shown Figure 4.4.

D.2 Inputting

After starting the program, the main form of the program is displayed on the screen as shown in Figure C.2. Inputting has been performed in the same order as in the 'User's Manual' section.

In order to input the characteristics of the main layers in to the program, the 'Field Identification' button is clicked. The first layer is sand. After clicking the 'Add Sand Layer' button, initial depth, final depth, and unit weight of the main layer is asked in the window shown in Figure C.3. By pressing the 'Ok' button, the inputs are placed in the 'Grid Object' of the 'Field Identification Form'. The layers as dislayed in Figure C.3 have been entered to the program, 'Ok' button is pressed, and the parameters are stored in the memory of the computer.

The next button of the 'Tools' menu, which is SPT value button, has been clicked to enter the SPT results into the program. The 'Add SPT Data' button has been clicked to enter the beginning depth and SPT-N value of the first SPT. The values, which are displayed in the 'Grid Object' of the 'SPT Properties' form in Figure C.4, has been entered into the program. As the energy ratio of the SPT equipment 45 percent value has been entered. Then 'Ok' button has been clicked to store the inputs into the memory.

Third button of the tools menu is the GWT Position button. As displayed in Figure C.5, 7 m is entered by typing the value in the text cell and pressed 'Ok' button.

'Pile Properties' button has been clicked as the last input button. In the 'Pile Properties' form, 'circular' is selected with a diameter of 0.65 m in the 'Section' frame, 'driven' is selected in the 'Installation' frame as shown in Figure C.6. Then 'Ok' button is clicked to store the input values in the memory.

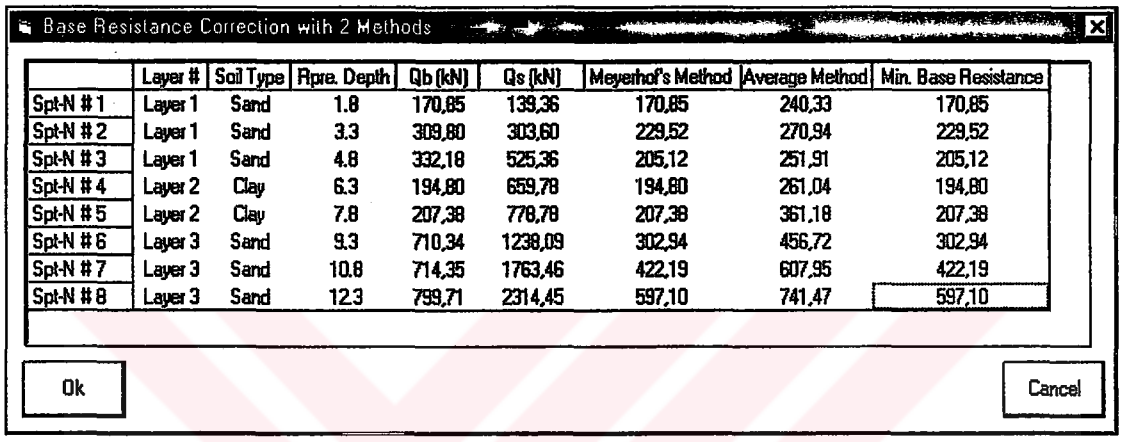
At this stage, all of the input parameters have been entered, and these parameters can be saved to the disk as a file with '.geo' extension, by clicking the save button on the toolbar. If required, these files can be loaded and run in the future by pressing the load button on the toolbar.

D.3 Calculations

The 'Adjustment Table' button, which is the first of the two calculation buttons, has been pressed to process the SPT correction. On the 'Adjustment Table' form, negative pore water correction for the last sand layer has been applied, and 'Accept Corrected SPT-N Values' button has been pressed to continue. (Figure C.7)

Static Formula Method for sand layers, and Alpha Method for clay layers have been adopted. By pressing the 'Bearing Capacity' button on the 'Main Form', which is the last button for calculation process, the program waits for user response for calculation method selection for sand and clay layers. For the first sand layer 'Static

Formula' button, for the clay layer 'Alpha Method', and for the second sand layer again 'Static Formula' button has been pressed. The results are shown on 'Bearing Capacity' form (Figure C.8). To continue, 'Ok' button has been pressed and two message boxes, which ask for the boundaries of average base resistance method, are displayed. On these message boxes default values are 3 and 5, respectively. After pressing 'Ok' buttons on these boxes, 'Base Resistance Correction' form shown in Figure D.1 has been displayed.



	Layer #	Soil Type	Rpre. Depth	Qb (kN)	Qs (kN)	Meyerhof's Method	Average Method	Min. Base Resistance
Spt-N # 1	Layer 1	Sand	1.8	170,85	139,36	170,85	240,33	170,85
Spt-N # 2	Layer 1	Sand	3.3	309,80	303,60	229,52	270,94	229,52
Spt-N # 3	Layer 1	Sand	4.8	332,18	525,36	205,12	251,91	205,12
Spt-N # 4	Layer 2	Clay	6.3	194,80	659,78	194,80	261,04	194,80
Spt-N # 5	Layer 2	Clay	7.8	207,38	776,78	207,38	361,18	207,38
Spt-N # 6	Layer 3	Sand	9.3	710,34	1238,09	302,94	456,72	302,94
Spt-N # 7	Layer 3	Sand	10.8	714,35	1763,46	422,19	607,95	422,19
Spt-N # 8	Layer 3	Sand	12.3	799,71	2314,45	597,10	741,47	597,10

Figure D. 1 'Base Resistance Correction' form

The base resistance correction has been applied with two methods automatically. 'Ok' button has been pressed to continue, and returned to the 'Main Form'.

D.4 Outputs

At the end of the above mentioned operations, the inputs or results has been placed into the 'Grid Object' of the 'Main Form' to prepare an output report. This report has been also used to form the printer output. To get the printed output, 'Print Results' button has been clicked, and the page, shown in Figure D.2, has been obtained from the printer.

Soil Profile:

Initial Depth	Final Depth	Soil Type	Unit Weight
0	5	Sand	19
5	8	Clay	20
8	13	Sand	17

Ground Water Level= 7 m.
Pile Diameter= .55 m.
Pile Shape= Circular
Installation Method= Driven

SPT:

Initial	Final	Repre. Depth	SPT-N
1.5	1.95	1.8	9
3	3.45	3.3	12
4.5	4.95	4.8	14
6	6.45	6.3	19
7.5	7.95	7.8	22
9	9.45	9.3	17
10.5	10.95	10.8	16
12	12.45	12.3	19

Energy Ratio= 45

Calculation Results

Soil Type	Rep. Depth	Sig	Cn	CSPT-N	Cal.Method	Fi	Cu	Qb	C. Qb 1	C. Qb 2	Qs	Qu 1	Qu 2
Sand	1.8	34.2	1.57	15.06	StaticFormula	28.2	-	170.85	170.85	240.33	139.36	310.21	379.69
Sand	3.3	62.7	1.24	14.89	StaticFormula	28.1	-	309.8	229.52	270.94	303.6	533.12	574.54
Sand	4.8	91.2	1.02	14.35	StaticFormula	27.88	-	332.18	205.12	251.91	525.36	730.48	777.27
Clay	6.3	121	.89	16.9	Alfa	-	65.2	194.8	194.8	261.04	698.85	893.65	959.89
Clay	7.8	143.15	.82	17.99	Alfa	-	69.4	207.38	207.38	361.18	867.82	1075.2	1229
Sand	9.3	154.54	.79	13.38	StaticFormula	27.44	-	710.34	302.94	450.64	1334.41	1637.35	1785.05
Sand	10.8	165.92	.76	12.18	StaticFormula	26.87	-	690.05	411.72	591.36	1850.63	2262.35	2441.99
Sand	12.3	176.11	.74	14.01	StaticFormula	27.73	-	757.68	569.3	719.36	2377.65	2946.95	3097.01

Figure D. 2 Printer Output

Clicking the 'Graphs' button in the 'Main Form' has plotted the resulting graphs. A new form has been displayed on the screen. The desired graphs have been plotted by selecting from the menu. Soil profile, σ_o', raw and corrected SPT-N values, base resistances, corrected base resistances with Average and Meyerhof's methods, skin friction, and ultimate bearing capacity versus depth graphs have been plotted as shown in Figure D.3. These graphs can also be printed from the printer by pressing the 'Print Page' button in the menu of the 'Graph' form.

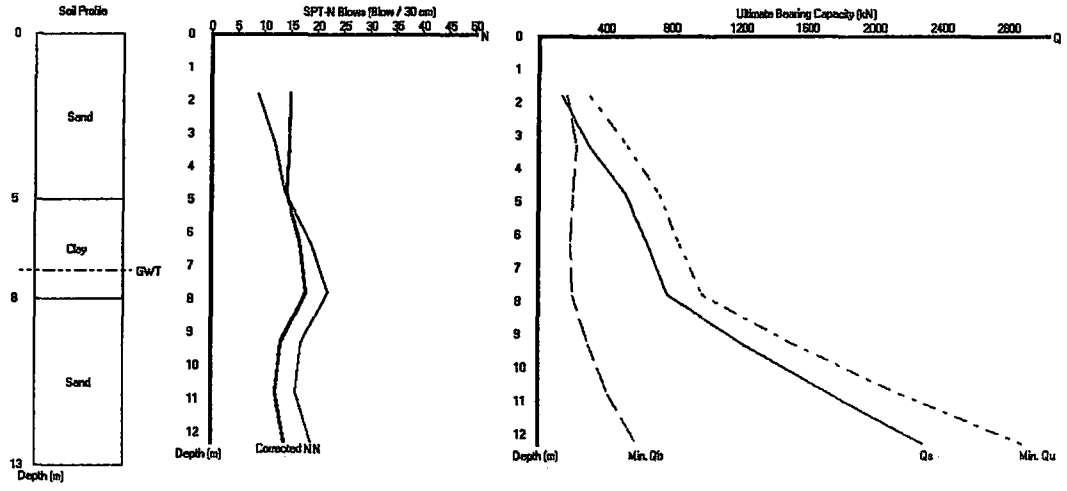


Figure D. 3 The resulting graphs of the sample application

APPENDIX E Resulting Graphs of Application Section

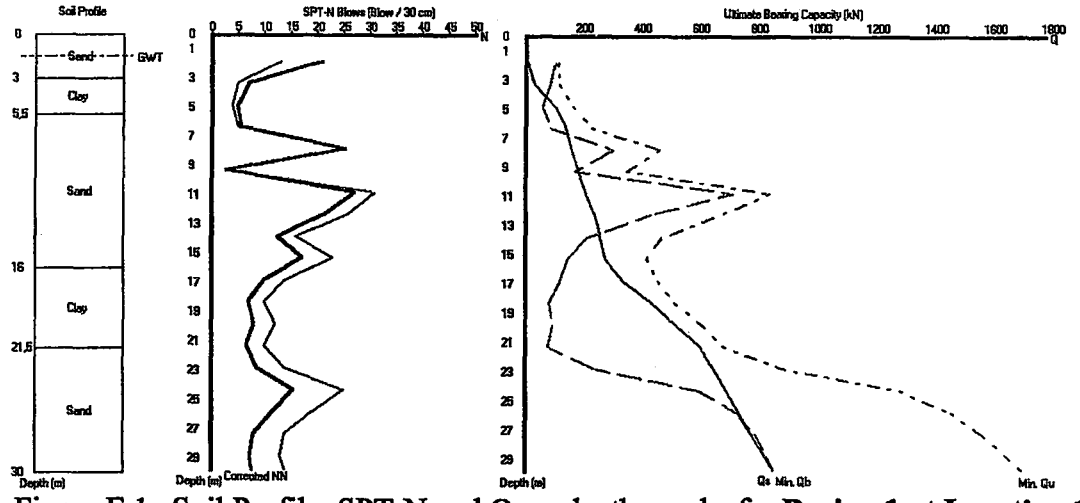


Figure E.1a Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 1 at Location 1**

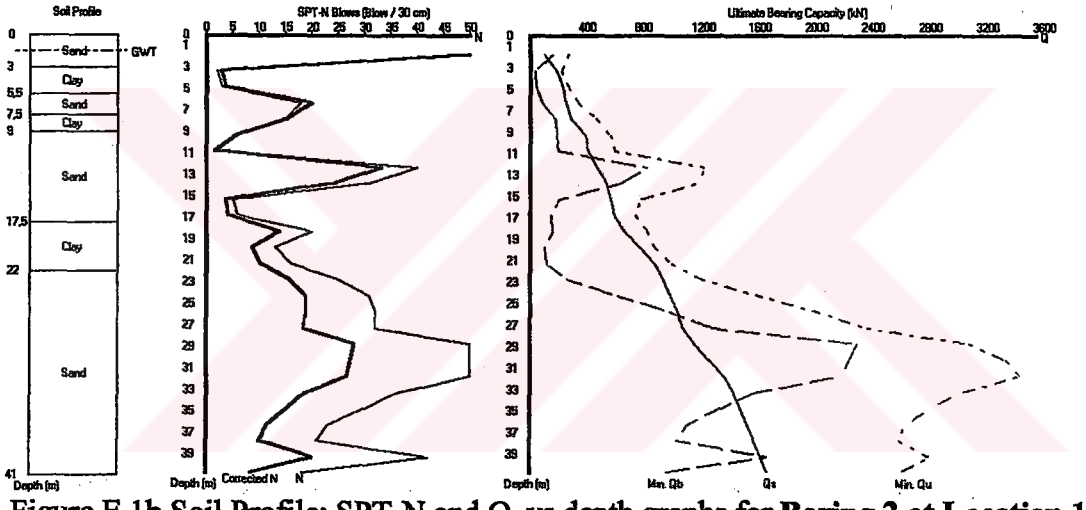


Figure E.1b Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 2 at Location 1**

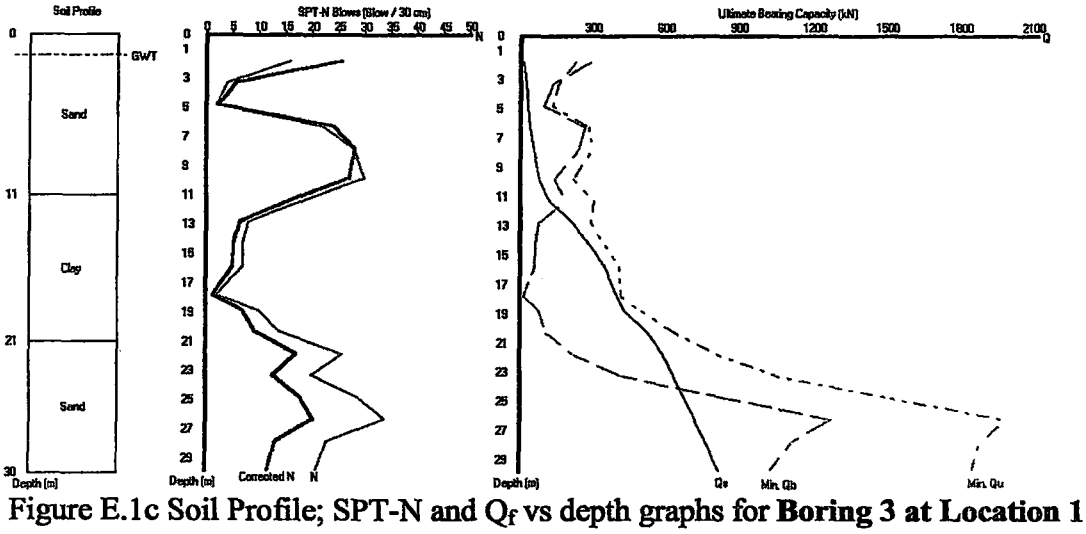


Figure E.1c Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 3 at Location 1**

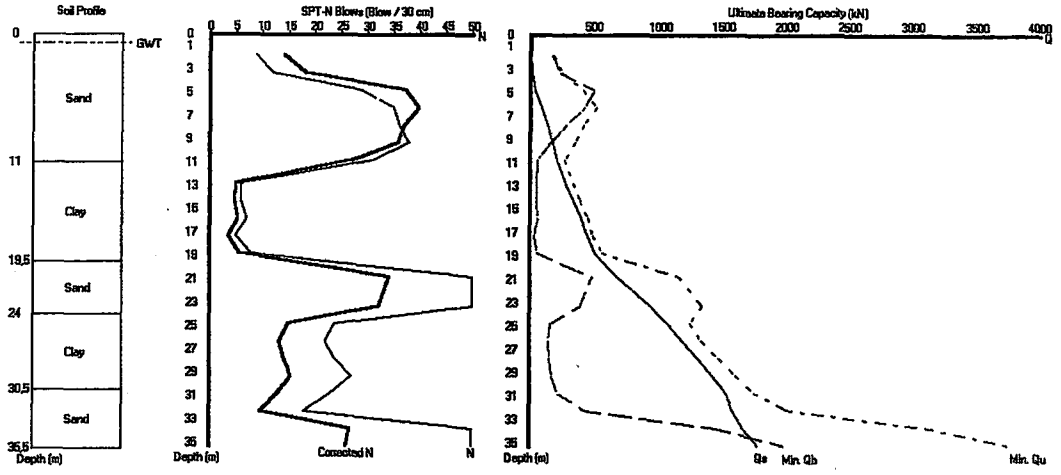


Figure E.2a Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 1 at Location 2**

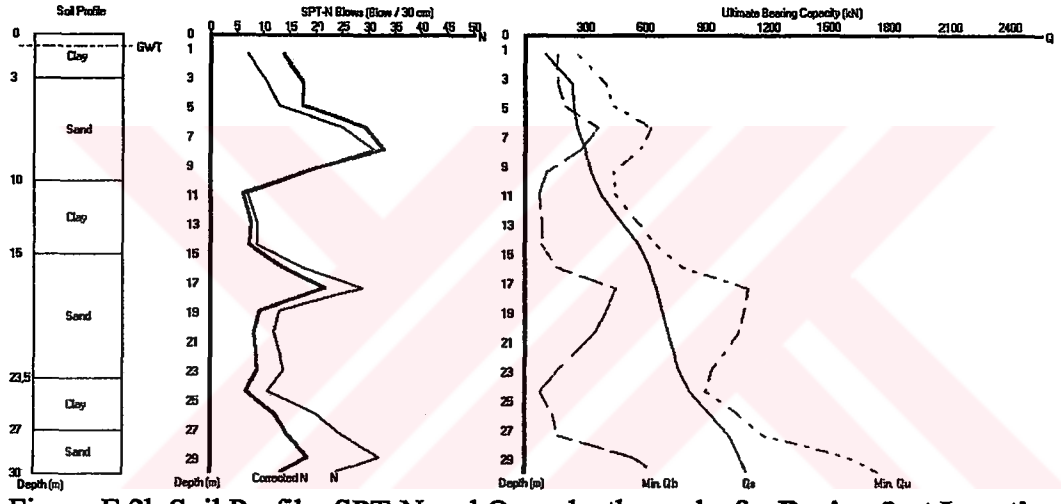


Figure E.2b Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 2 at Location 2**

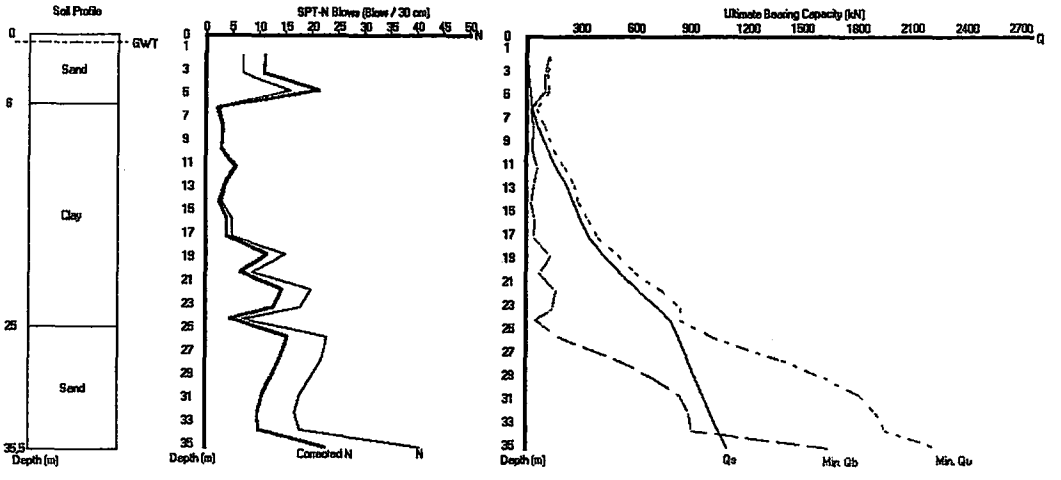


Figure E.2c Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 3 at Location 2**

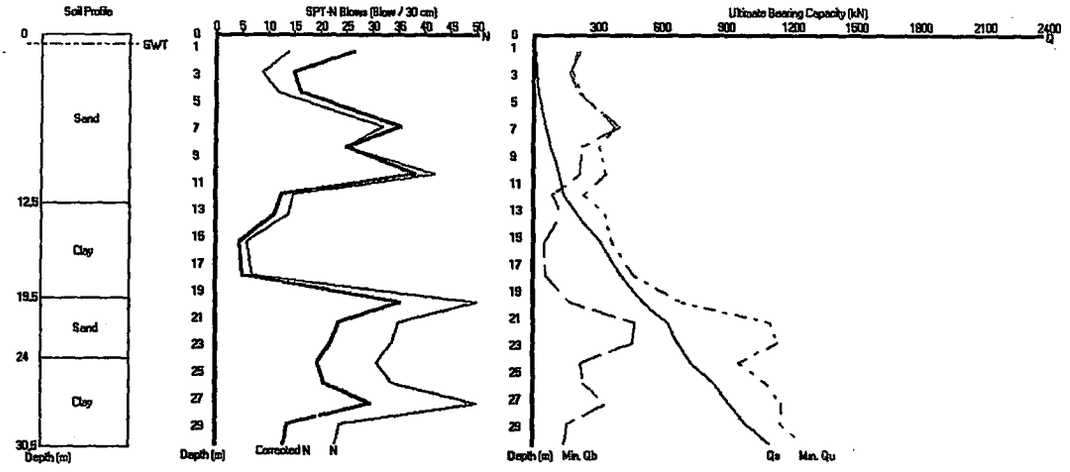


Figure E.2d Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 4 at Location 2**

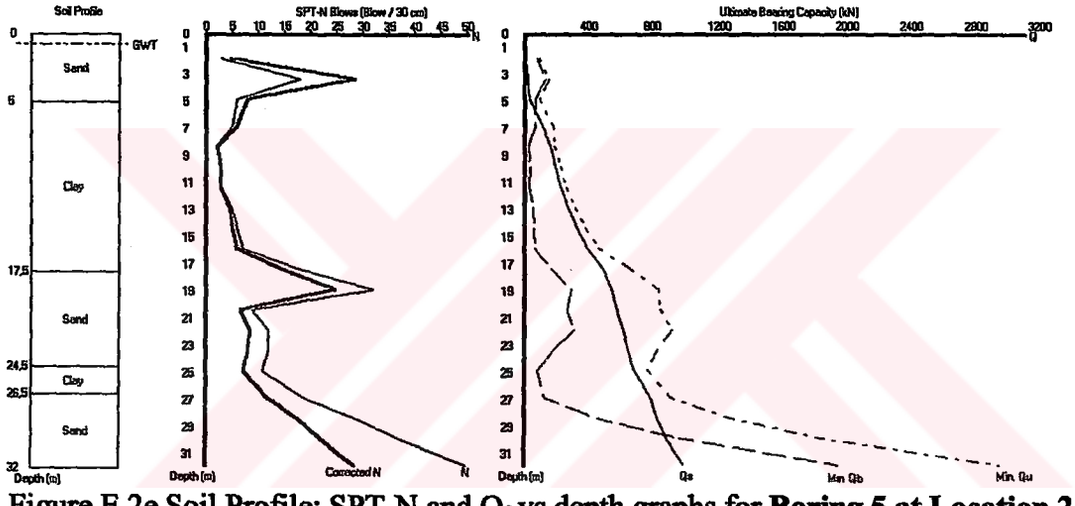


Figure E.2e Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 5 at Location 2**

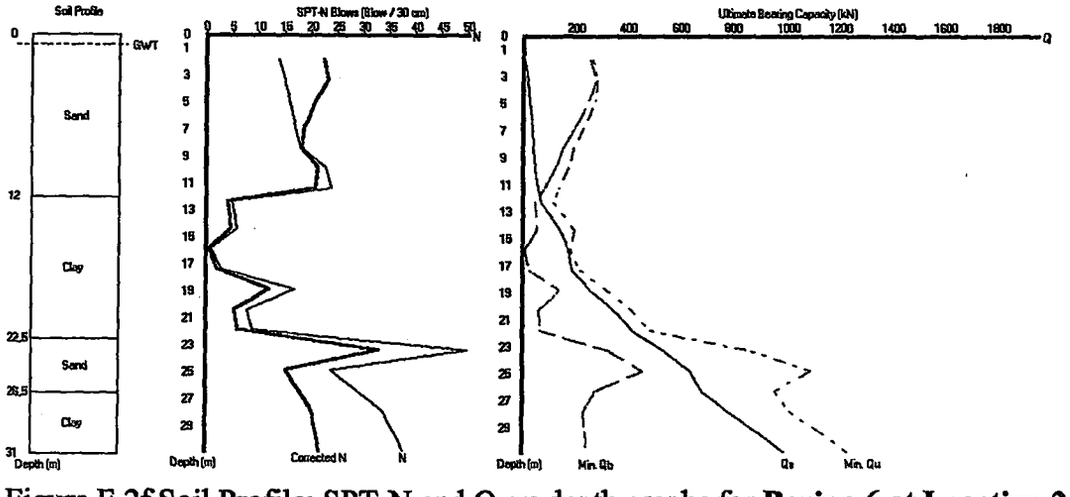


Figure E.2f Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 6 at Location 2**

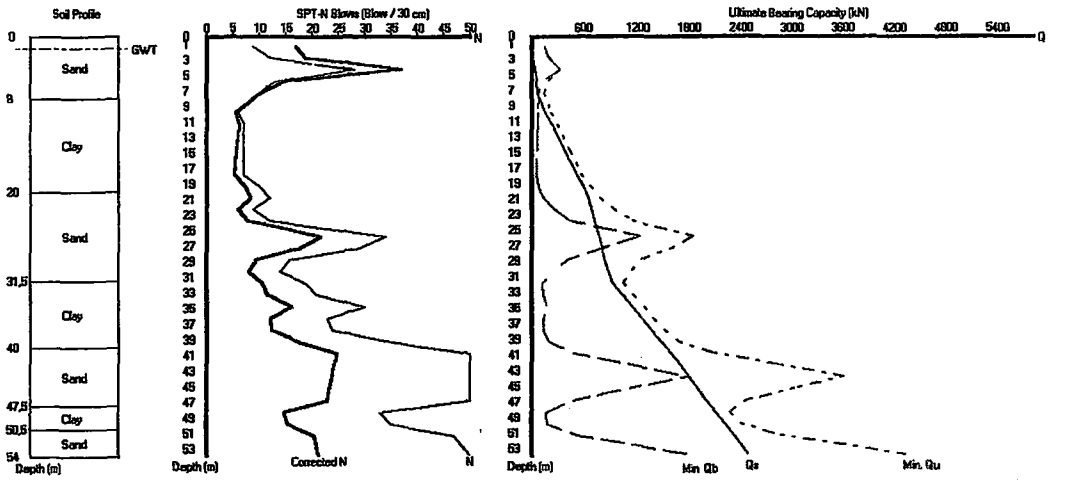


Figure E.2g Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 7 at Location 2**

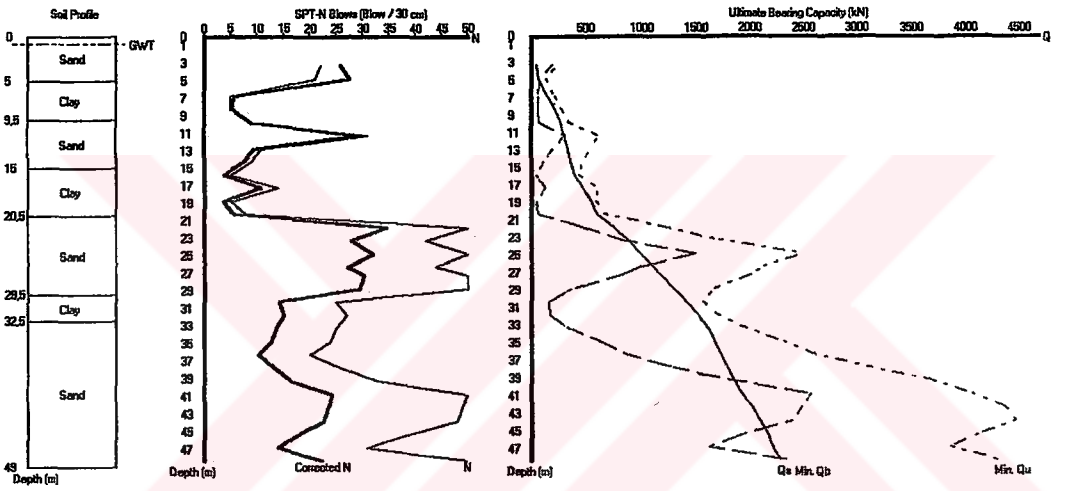


Figure E.2h Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 8 at Location 2**

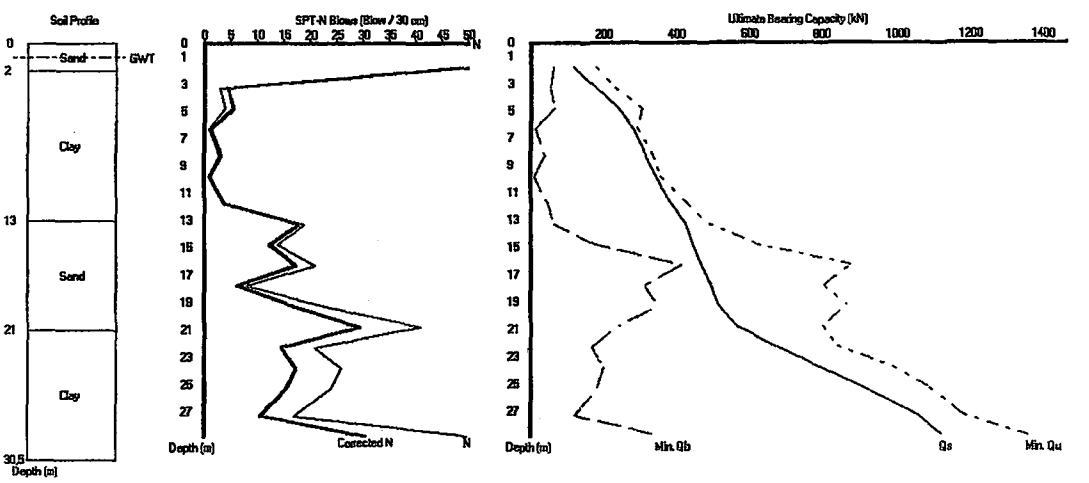


Figure E.3a Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 1 at Location 3**

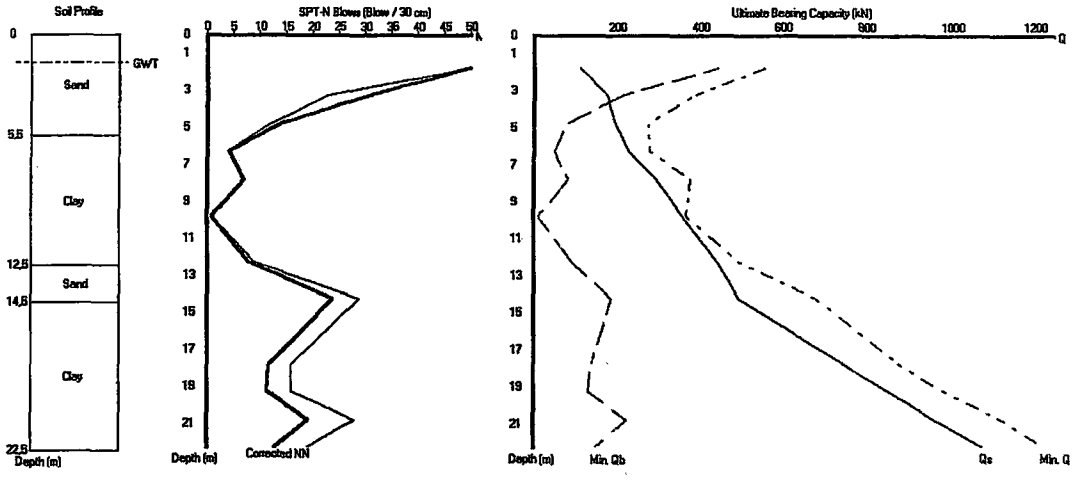


Figure E.3b Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 2 at Location 3**

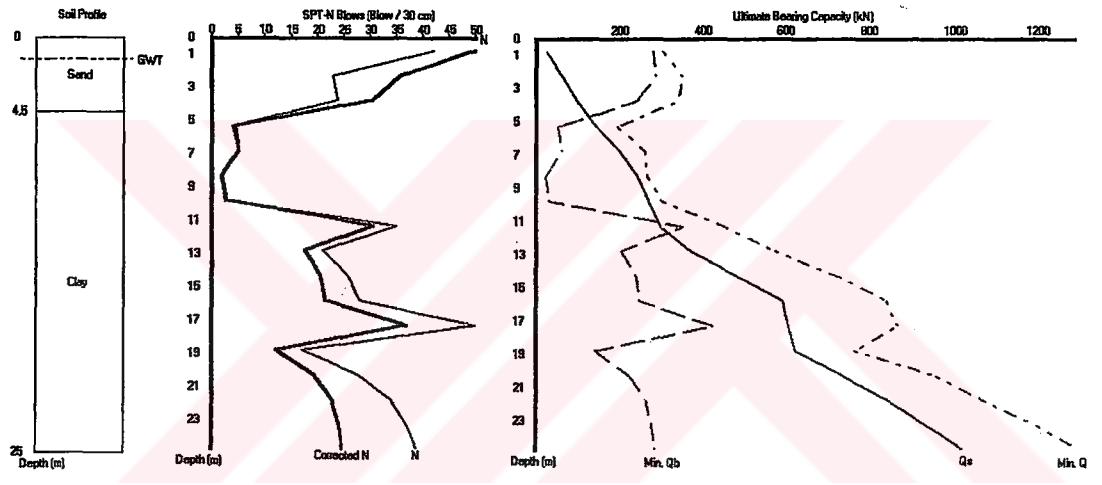


Figure E.4 Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 1 at Location 4**

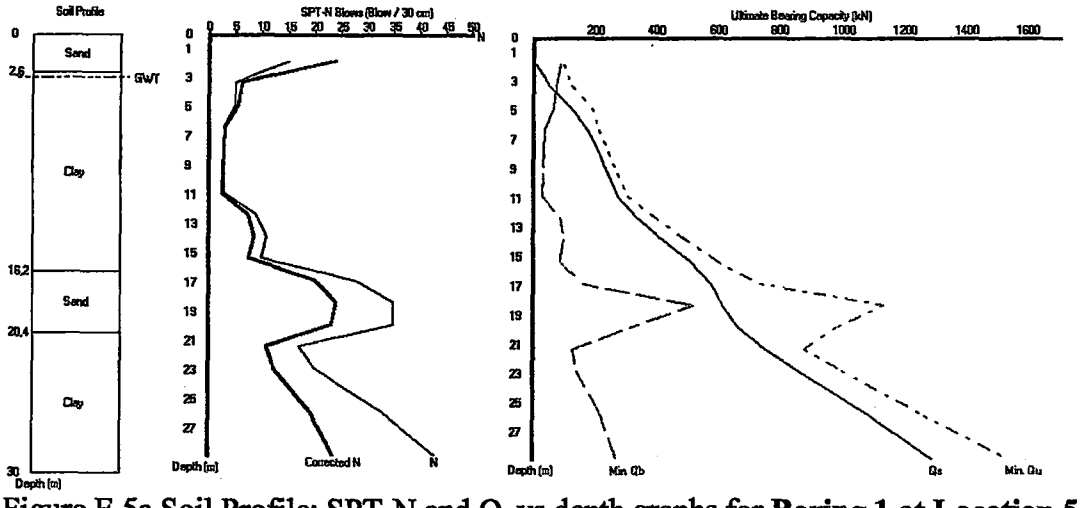


Figure E.5a Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 1 at Location 5**

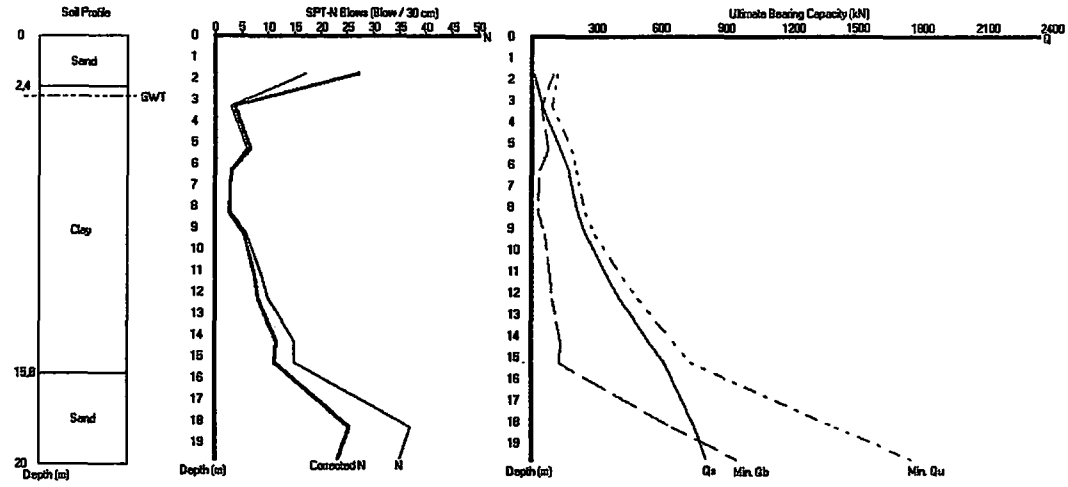


Figure E.5b Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 2 at Location 5**

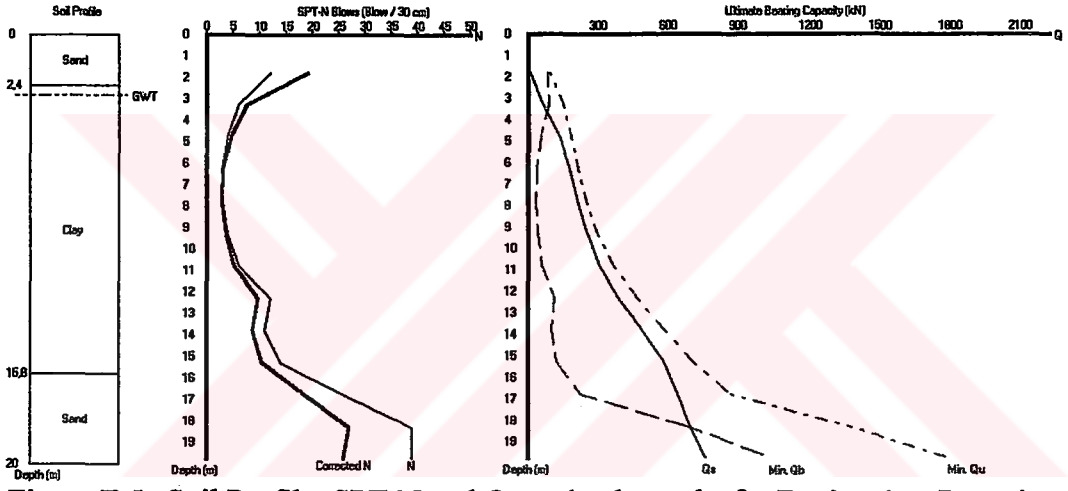


Figure E.5c Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 3 at Location 5**

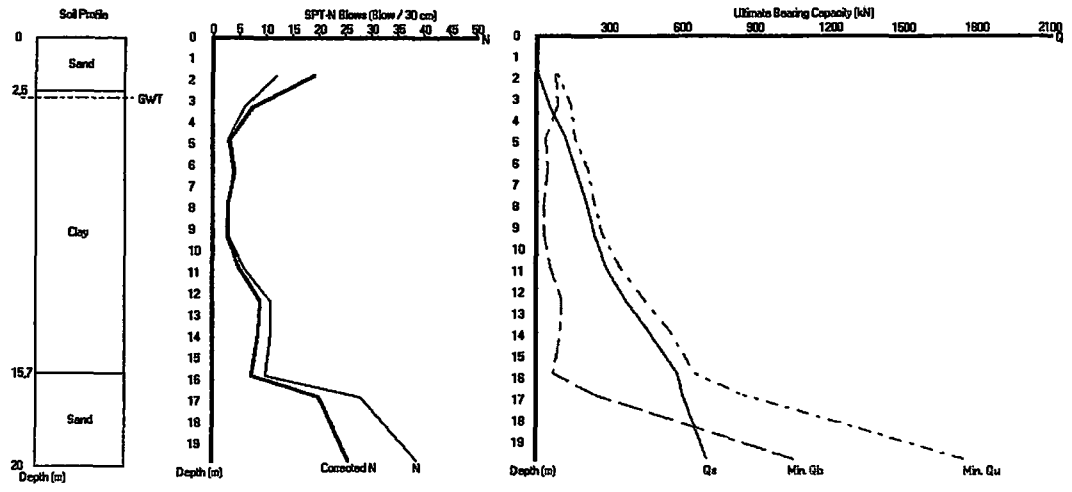


Figure E.5d Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 4 at Location 5**

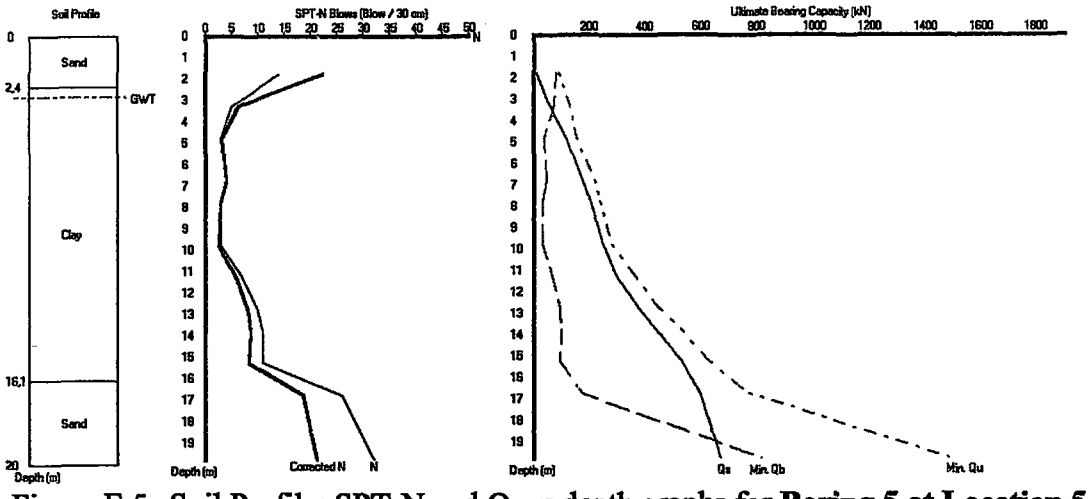


Figure E.5e Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 5 at Location 5**

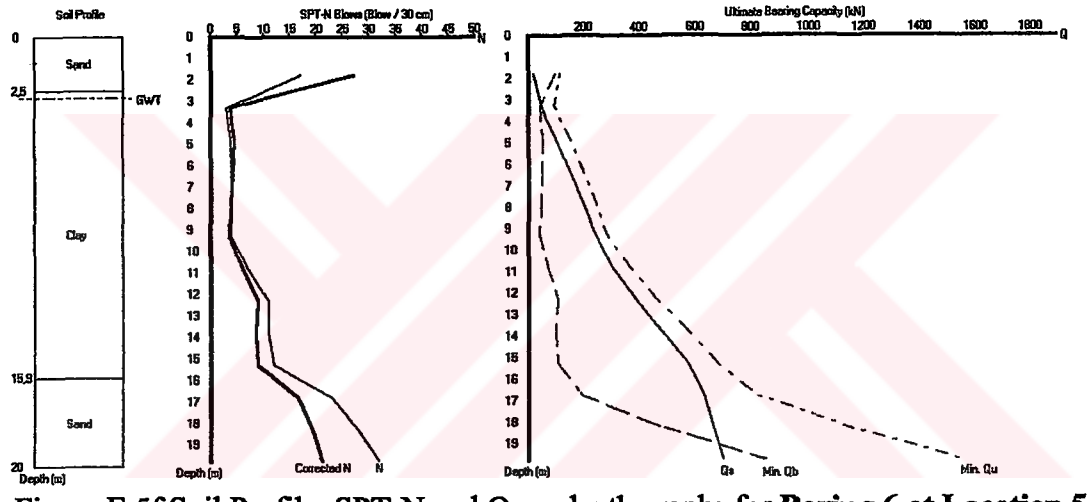


Figure E.5f Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 6 at Location 5**

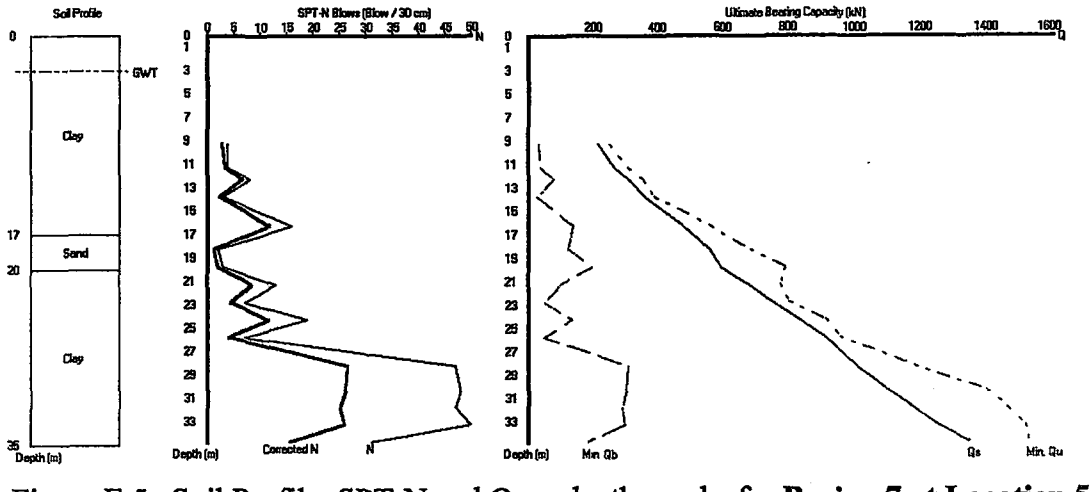


Figure E.5g Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 7 at Location 5**

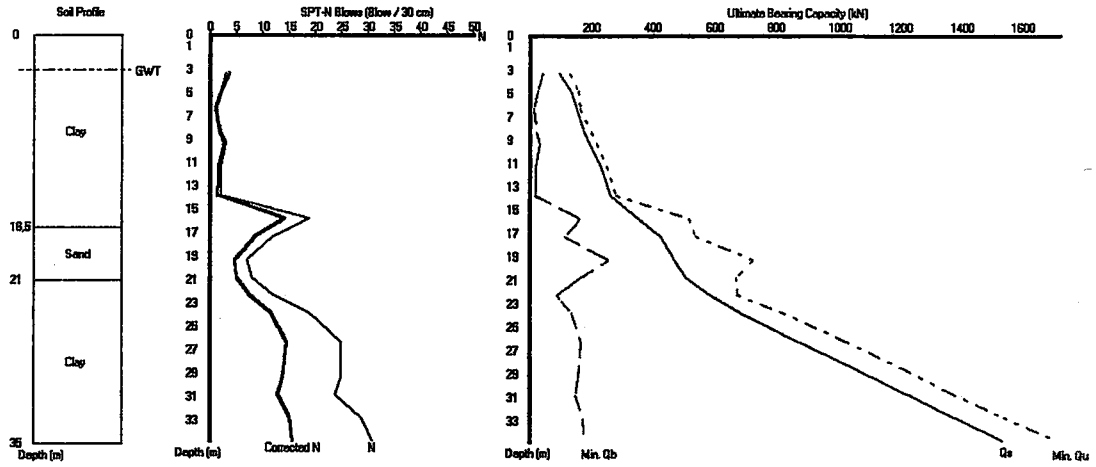


Figure E.5h Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 8 at Location 5**

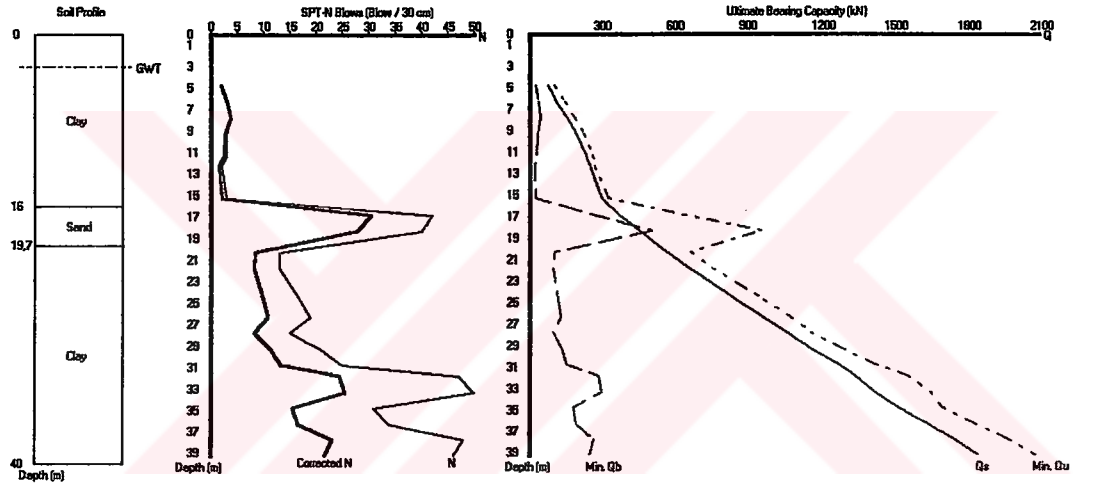


Figure E.5i Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 9 at Location 5**

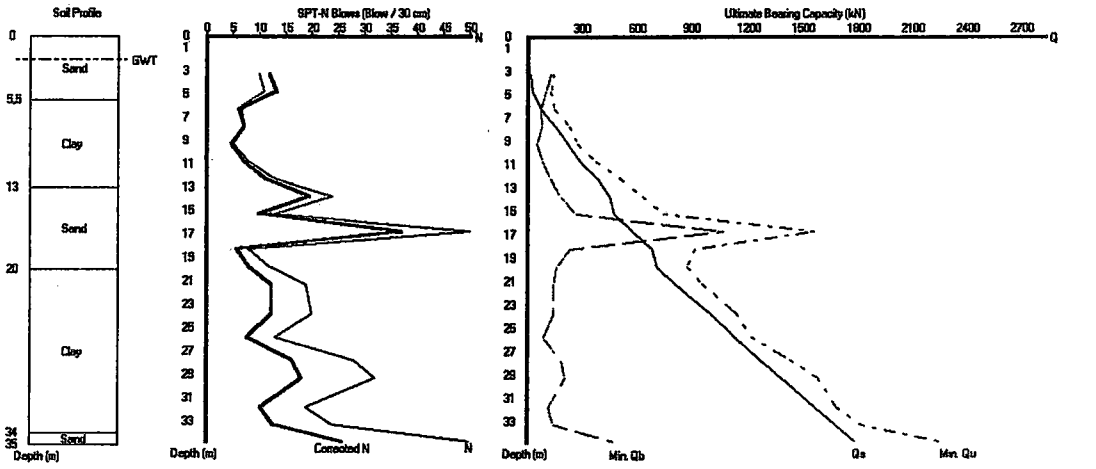


Figure E.6a Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 1 at Location 6**

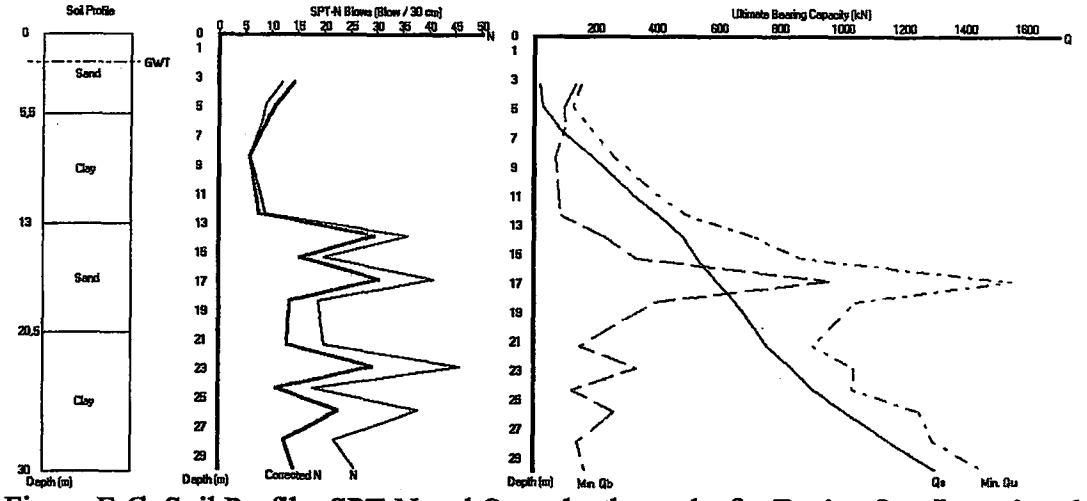


Figure E.6b Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 2 at Location 6**

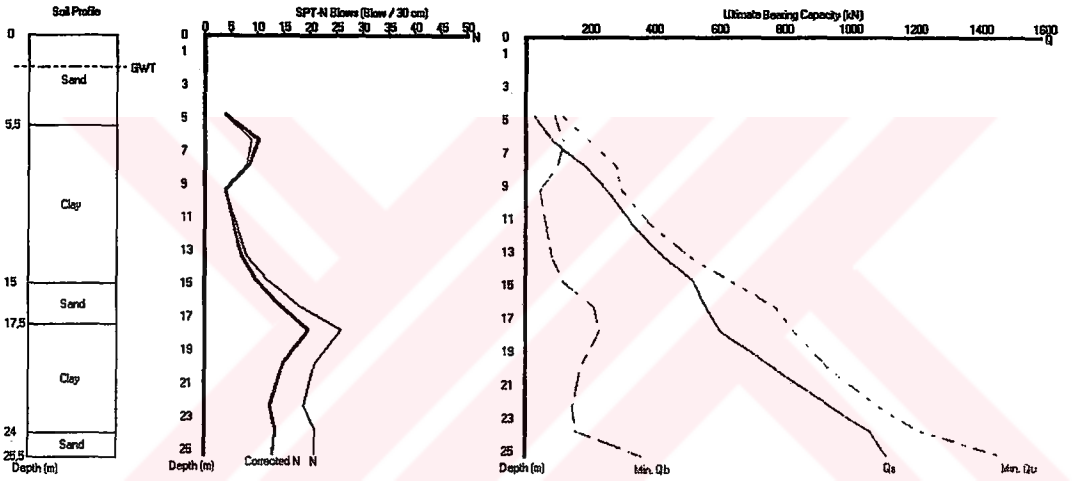


Figure E.6c Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 3 at Location 6**

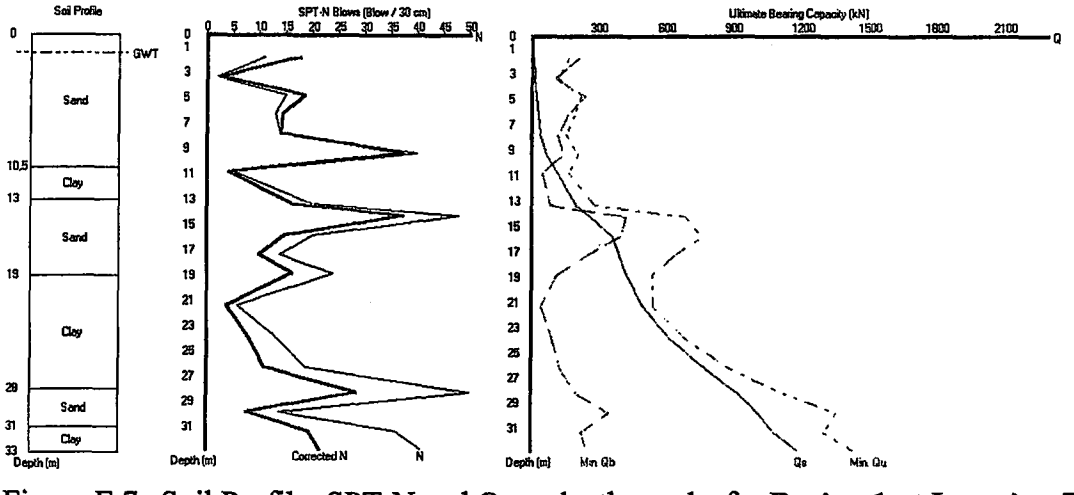


Figure E.7a Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 1 at Location 7**

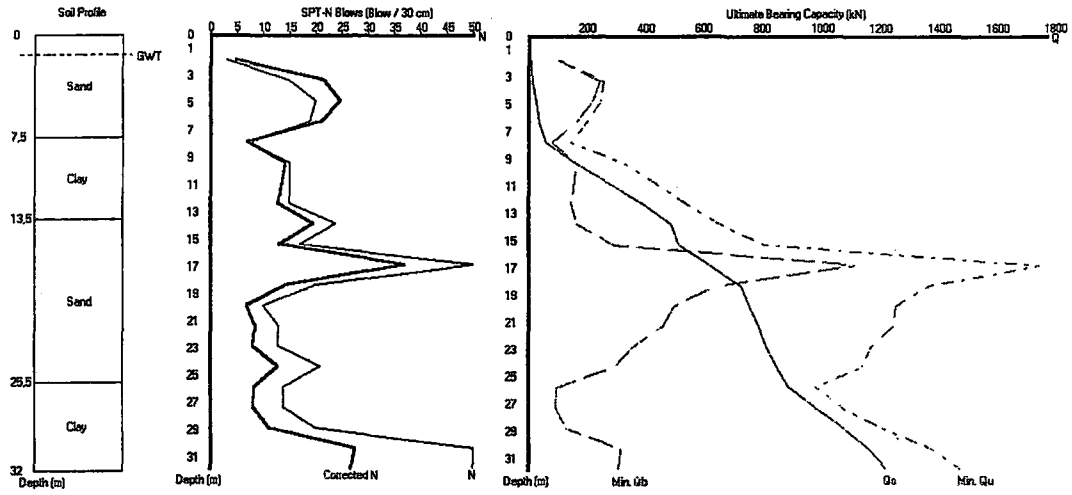


Figure E.7b Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 2 at Location 7**

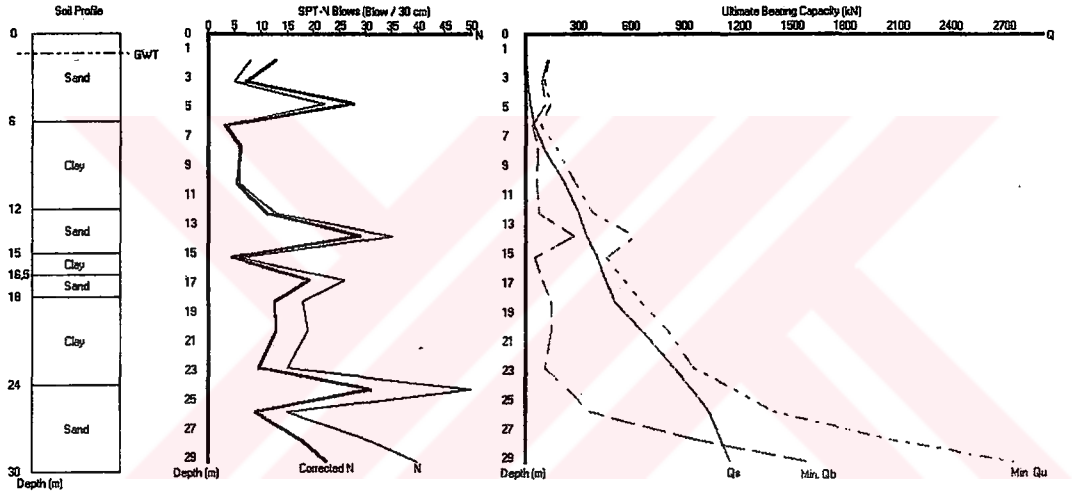


Figure E.7c Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 3 at Location 7**

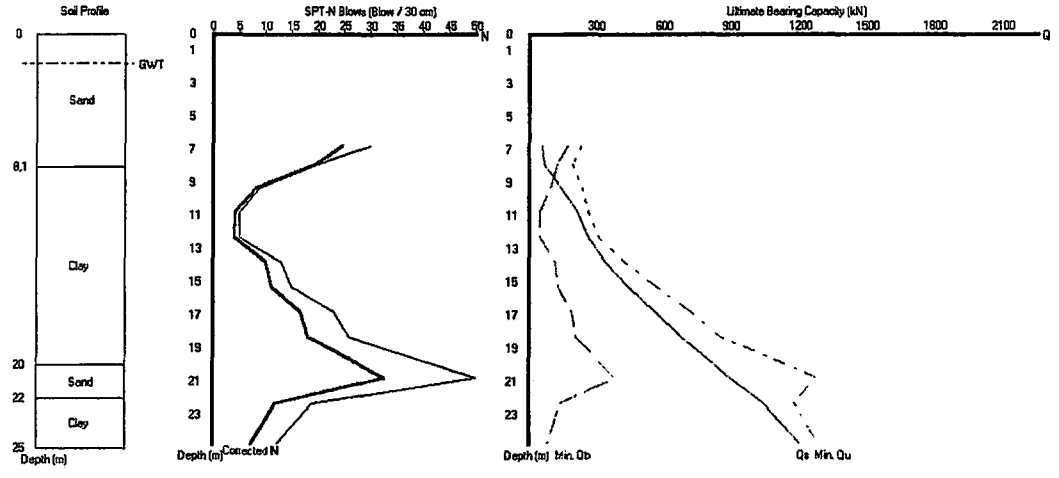


Figure E.8a Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 1 at Location 8**

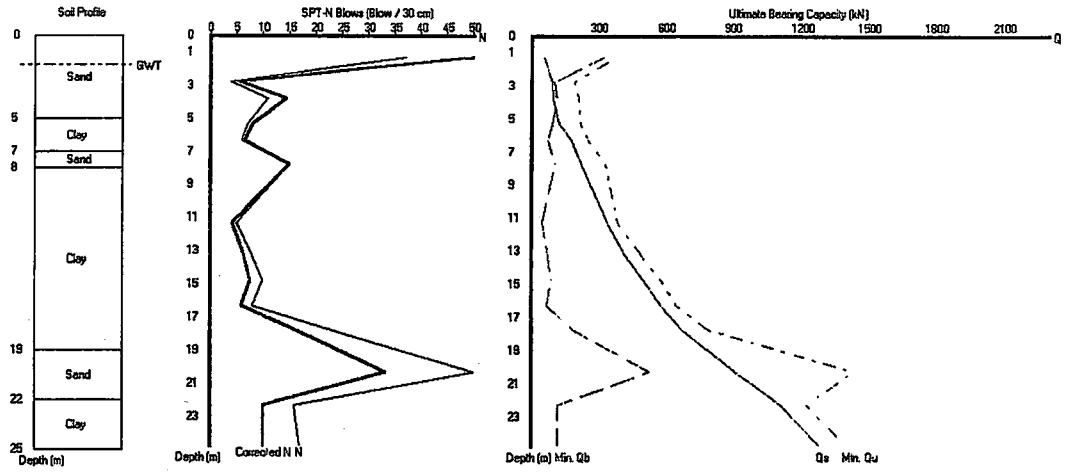


Figure E.8b Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 2 at Location 8**

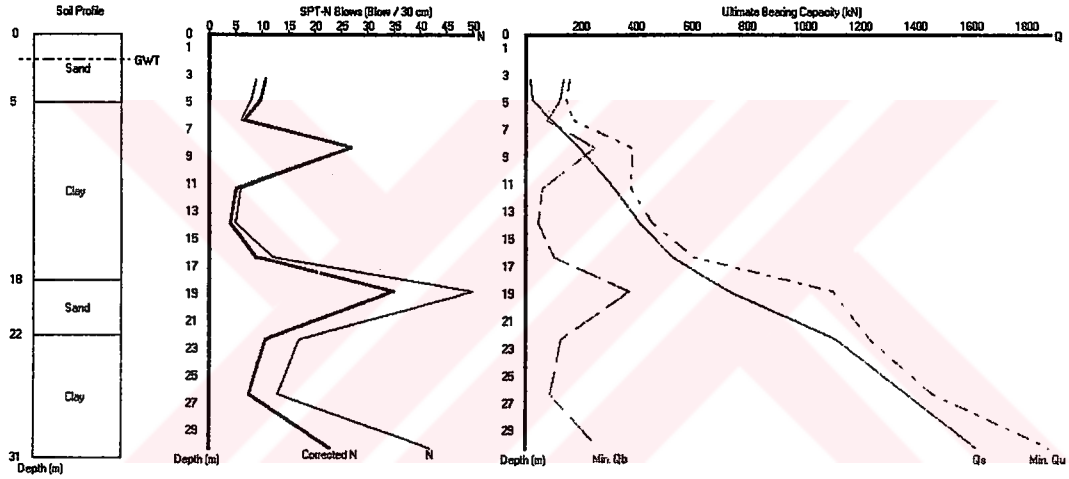


Figure E.8c Soil Profile; SPT-N and Q_f vs depth graphs for **Boring 3 at Location 8**

APPENDIX F Trend Line Curves and Their Equations

Table F. 1 ϕ and z_c/d Values Read from Figure 2.3

ϕ	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
z_c/D	5.0	5.2	5.3	5.5	5.9	6.0	6.4	6.7	7.0	7.8	10.0	12.0	14.7	18.0	20.0

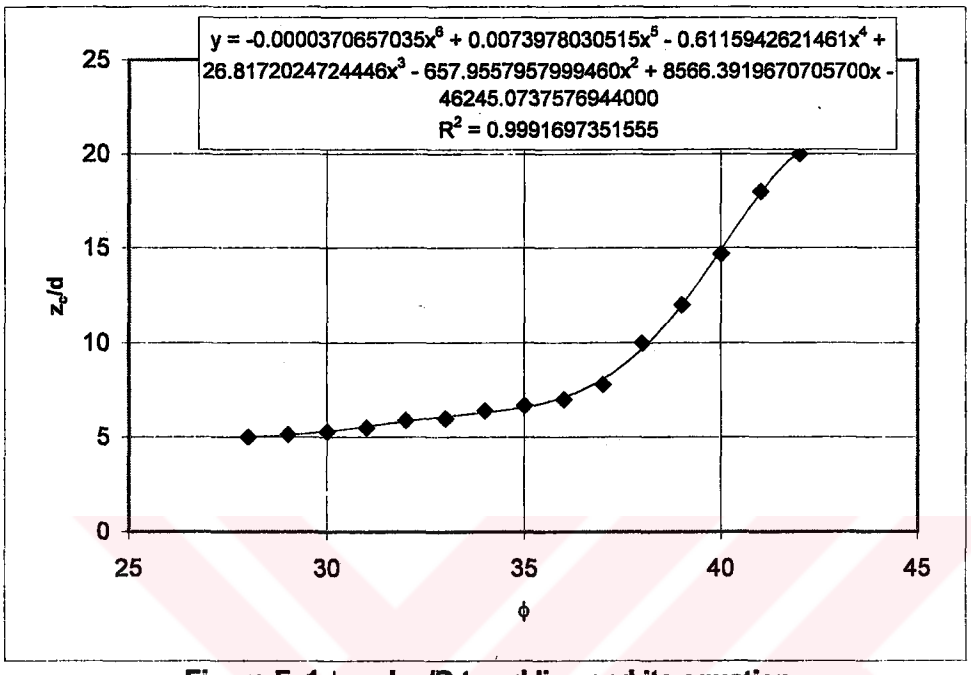


Figure F. 1 ϕ and z_c/D trend line and its equation

Table F. 2 ϕ and Ks tan ϕ Values Read from Figure 2.2(a)

ϕ	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
Ks tan ϕ	1.00	1.03	1.05	1.10	1.16	1.23	1.30	1.38	1.47	1.60	1.70	1.85	2.05	2.34	2.70

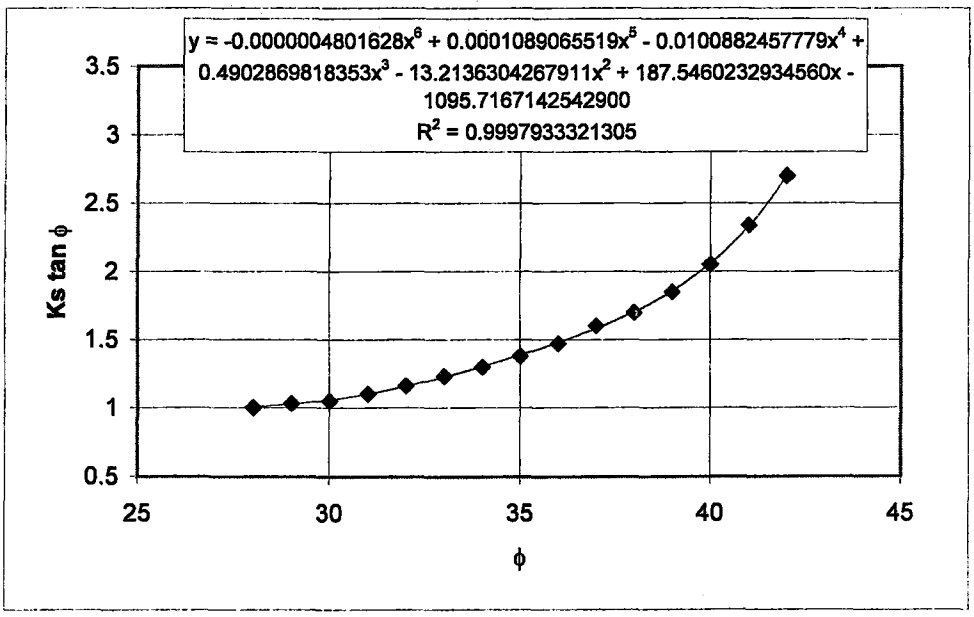


Figure F. 2 ϕ and Ks tan ϕ trend line and its equation

Table F. 3 ϕ and Ks tan ϕ Values Read from Figure 2.2(b)

ϕ	30	31	32	33	34	35	36	37	38	39	40
Ks tan ϕ	0.058	0.072	0.087	0.130	0.202	0.280	0.363	0.482	0.625	0.769	1.007

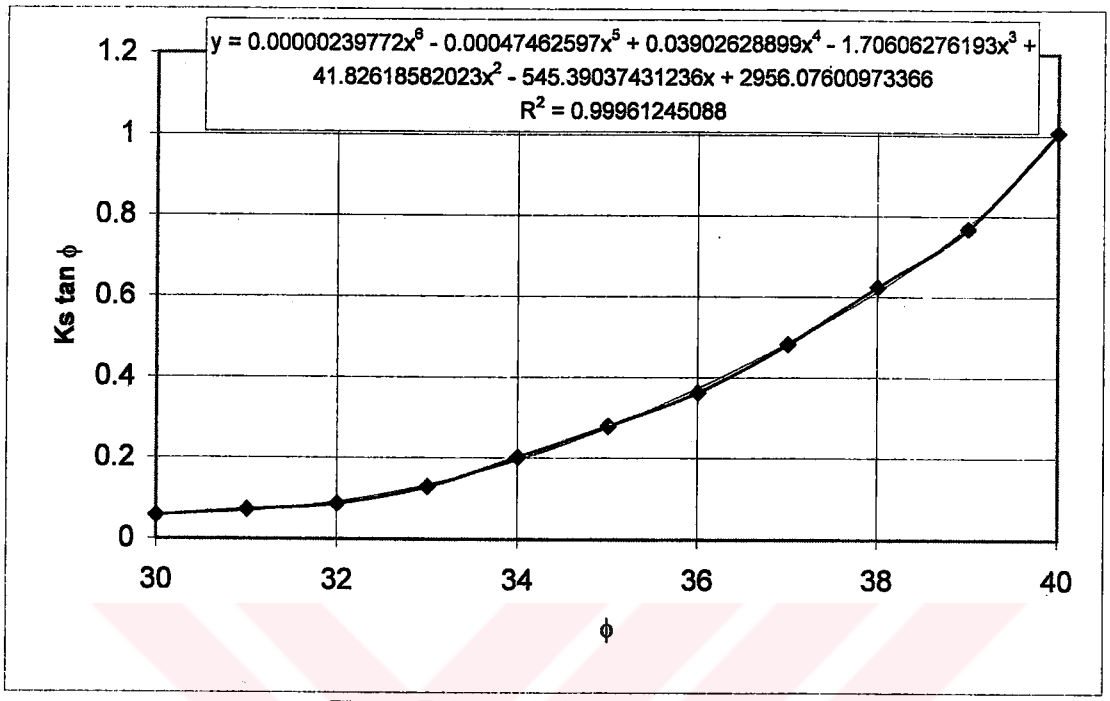


Figure F. 3 ϕ and Ks tan ϕ trend line and its equation

Table F. 4 ϕ and Ip Values Read from Figure 4.6

ϕ	26.15	21.4	18.1	15	13.8	12	10.08	10	9.9
Ip	10	20	30	40	50	60	70	80	90

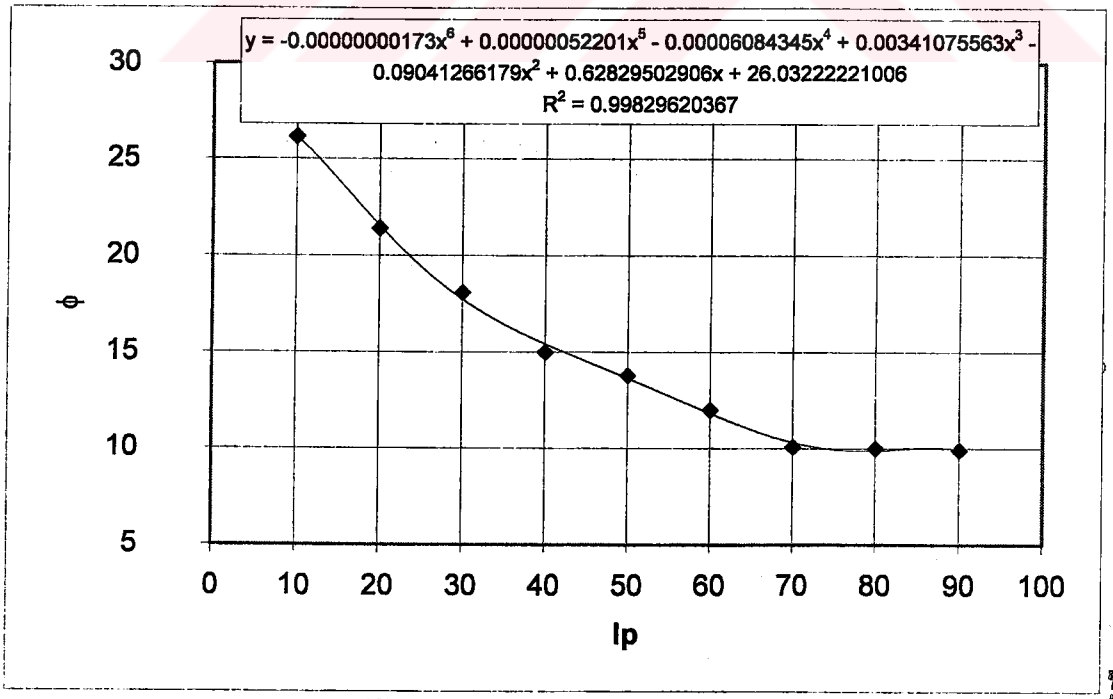


Figure F. 4 ϕ and Ip trend line and its equation