Assignment 7

Q1

```
# Program to Determine the bearing capacity of soil with water table
# Input values
BulkDensity = float(input("Enter the value of Bulk Density of soil (kN/m^3): "))
SatDensity = float(input("Enter the value of Saturated Density of soil (kN/m^3): "))
WaterDensity = float(input("Enter the unit Weight of Water (kN/m^3): "))
Df = float(input("Enter the value of depth of footing Df (m): "))
Dw = float(input("Enter the value of water table above footing level Dw (m): "))
# Handle potential empty input for Dw1
while True:
dw1_input = input("Enter the value of Water table below the level of footing Dw1 (m): ")
try:
Dw1 = float(dw1_input)
break # Exit the loop if conversion is successful
except ValueError:
print("Invalid input. Please enter a valid number for Dw1.")
B = float(input("Enter the value of width of footing B (m): "))
Nq = float(input("Enter the value of Nq: "))
N = float(input("Enter the value of N (bearing capacity factor): "))
# Submerged density
SubDensity = SatDensity - WaterDensity
print("Submerged Weight of soil is:", SubDensity)
# ---------------- CASE A ----------------
print("\nCASE A: Water table at ground surface")
qu = (SubDensity * Df * Nq) + (0.5 * B * SubDensity * N)
print("The value of ultimate bearing capacity of soil is:", qu)
Rw = 0.5 + 0.5 * (Dw / B)
```

```
print("The value of Rw is:", Rw)
Rw1 = 0.5 + 0.5 * (Dw1 / B)
print("The value of Rw1 is:", Rw1)
qu = (BulkDensity * Df * Nq * Rw) + (0.5 * B * BulkDensity * N * Rw1)
print("The approximate value of ultimate bearing capacity is:", qu)
# ---------------- CASE B ----------------
print("\nCASE B: Water table at base of footing")
qu = (BulkDensity * Df * Nq) + (0.5 * B * SubDensity * N)
print("The value of ultimate bearing capacity is:", qu)
Rw = 0.5 + 0.5 * (Dw / B)
print("The value of Rw is:", Rw)
Rw1 = 0.5 + 0.5 * (Dw1 / B)
print("The value of Rw1 is:", Rw1)
qu = (BulkDensity * Df * Nq * Rw) + (0.5 * B * BulkDensity * N * Rw1)
print("The approximate value of ultimate bearing capacity is:", qu)
# ---------------- CASE C ----------------
print("\nCASE C: Water table below base of footing")
while True:
x_input = input("Enter the value of depth of water below footing (x in m): ")
try:
x = float(x_input)
break
except ValueError:
print("Invalid input. Please enter a valid number for x.")
qu = (BulkDensity * Df * Nq) + (0.5 * B * ((BulkDensity * x) + (SubDensity * (B - x))) * N)
print("The value of ultimate bearing capacity is:", qu)
Rw = 0.5 + 0.5 * (Dw / B)
print("The value of Rw is:", Rw)
Rw1 = 0.5 + 0.5 * (Dw1 / B)
```

```
print("The value of Rw1 is:", Rw1)
qu = (BulkDensity * Df * Nq * Rw) + (0.5 * B * BulkDensity * N * Rw1)
print("The approximate value of ultimate bearing capacity is:", qu)
```

output-

Enter the value of Bulk Density of soil (kN/m^3): 18

Enter the value of Saturated Density of soil (kN/m^3): 20

Enter the unit Weight of Water (kN/m^3): 10

Enter the value of depth of footing Df (m): 2

Enter the value of water table above footing level Dw (m): 0

Enter the value of Water table below the level of footing Dw1 (m): 0

Enter the value of width of footing B (m): 3

Enter the value of Nq: 33

Enter the value of N (bearing capacity factor): 34

Submerged Weight of soil is: 10.0

CASE A: Water table at ground surface

The value of ultimate bearing capacity of soil is: 1170.0

The value of Rw is: 0.5

The value of Rw1 is: 0.5

The approximate value of ultimate bearing capacity is: 1053.0

CASE B: Water table at base of footing

The value of ultimate bearing capacity is: 1698.0

The value of Rw is: 0.5

The value of Rw1 is: 0.5

The approximate value of ultimate bearing capacity is: 1053.0

CASE C: Water table below base of footing

Enter the value of depth of water below footing (x in m): 1

The value of ultimate bearing capacity is: 3126.0

The value of Rw is: 0.5

The value of Rw1 is: 0.5

The approximate value of ultimate bearing capacity is: 1053.0

Q2

```
#  To find the ultimate load carrying capacity of pile
UCS = float(input("Enter the value of UCS of soil:"))
Cu = UCS / 2
B = float(input("Enter the value of dimension of pile:"))
L = float(input("Enter the length of pile:"))
Alpha = float(input("Enter the value of adhesion factor:"))
Nc = float(input("The value of Nc: "))
Ab = B * B
print("The Base area of footing is:", Ab)
As = 4 * B * L
print("The value of cohesion of soil is:", Cu)
Qpu = Cu * Nc * Ab
print("Qpu:", Qpu)
Qf = Alpha * Cu * As
print("Qf:", Qf)
Qu = Qpu + Qf
print("The value of load carrying capacity of pile is (Qu):", Qu)
```

output-

Enter the value of UCS of soil:75

Enter the value of dimension of pile:0.45

Enter the length of pile:15

Enter the value of adhesion factor:0.8

The value of Nc: 9

The Base area of footing is: 0.2025

The value of cohesion of soil is: 37.5

Qpu: 68.34375

Qf: 810.0

The value of load carrying capacity of pile is (Qu): 878.34375

Q3

```
# Program 3: To Determine the bearing capacity of soil with water table (multiple cases)
BulkDensity = float(input("Enter the value of Bulk Density of soil:"))
SatDensity = float(input("Enter the value of Saturated Density of soil:"))
WaterDensity = float(input("Enter the unit Weight of Water:"))
Df = float(input("Enter the value of depth of footing:"))
B = float(input("Enter the value of width of footing:"))
Nq = float(input("Enter the value of Nq:"))
N_Gamma = float(input("Enter the value of N gamma (N):"))
SubDensity = SatDensity - WaterDensity
print("Submerged Weight of soil is:", SubDensity)
M = int(input("Number of data values of Water table above footing level: "))
N = int(input("Number of data values of Water table below footing level: "))
Dw = []
Dw1 = []
for i in range(1, M+1):
Depth_Dw = float(input("Enter the value of water table above footing level measured w.r.t. ground (Dw): "))
Dw.append(Depth_Dw)
Rw = 0.5 + 0.5 * (Depth_Dw / B)
print("The value of Rw is:", Rw)
for j in range(1, N+1):
Depth_Dw1 = float(input("Enter the value of water table below footing level measured w.r.t. ground (Dw1): "))
Dw1.append(Depth_Dw1)
```

```
Rw1 = 0.5 + 0.5 * (Depth_Dw1 / B)

print("The value of Rw1 is:", Rw1)

qu = (BulkDensity * Df * Nq * Rw) + (0.5 * 0.8 * B * BulkDensity * N_Gamma * Rw1)

print("qu: ", qu, "kN/m^2")
```

output-

```
Enter the value of Bulk Density of soil:18

Enter the value of Saturated Density of soil:20

Enter the unit Weight of Water:10

Enter the value of depth of footing:2

Enter the value of width of footing:3

Enter the value of Nq:33

Enter the value of N gamma (N):34

Submerged Weight of soil is: 10.0

Number of data values of Water table above footing level: 3

Number of data values of Water table below footing level: 3

Enter the value of water table above footing level measured w.r.t. ground (Dw): 0

The value of Rw is: 0.5

Enter the value of water table above footing level measured w.r.t. ground (Dw): 1

The value of Rw is: 0.6666666666666666

Enter the value of water table above footing level measured w.r.t. ground (Dw): 2

The value of Rw is: 0.8333333333333333

Enter the value of water table below footing level measured w.r.t. ground (Dw1): 0

The value of Rw1 is: 0.5

Enter the value of water table below footing level measured w.r.t. ground (Dw1): 0

The value of Rw1 is: 0.5

Enter the value of water table below footing level measured w.r.t. ground (Dw1): 1

The value of Rw1 is: 0.6666666666666666

qu:  1479.6 kN/m^2
```

assignment 8

Q1

```
# To determine alkalinity of given sample

H2SO4_req = float(input("Enter the volume of H2SO4 required in ml:"))

Sample = float(input("Enter the value of sample in litres:"))

Alkalinity_Removed = H2SO4_req

print("Alkalinity Removed:", Alkalinity_Removed, "mg")

Alk_mgperlit = Alkalinity_Removed / Sample

print("Total Alkalinity:", Alk_mgperlit, "mg/lit")

OH = float(input("Enter the value of OH-Alkalinity present: "))

# Alkalinity removed till pH of 8.3

H2SO4_req = float(input("Enter the volume of H2SO4 required in ml:"))

Alkalinity_Removed = H2SO4_req

print("Alkalinity Removed:", Alkalinity_Removed, "mg")

CO3_Combined = Alkalinity_Removed / Sample

print("Carbonate Alkalinity upto pH 8.3:", CO3_Combined, "mg/lit")

CO3 = CO3_Combined - OH

print("Carbonate Alkalinity:", CO3, "mg/lit")

HCO3 = Alk_mgperlit - 2 * CO3 - OH

print("Bicarbonate Alkalinity:", HCO3, "mg/lit")
```

output-

```
Enter the volume of H2SO4 required in ml:30

Enter the value of sample in litres:0.2

Alkalinity Removed: 30.0 mg

Total Alkalinity: 150.0 mg/lit

Enter the value of OH-Alkalinity present: 5

Enter the volume of H2SO4 required in ml:11
```

Alkalinity Removed: 11.0 mg

Carbonate Alkalinity upto pH 8.3: 55.0 mg/lit

Carbonate Alkalinity: 50.0 mg/lit

Bicarbonate Alkalinity: 45.0 mg/li

Assignment 9

Q1

```
# To find BOD at 7th day 25°C
K1 = float(input("Decay Coefficient at 20°C:"))
T = float(input("Temperature of 3rd day BOD:"))
T1 = float(input("Temperature of 7th day BOD:"))
K2 = (K1 * (1.047) ** (T1 - T))
print("The value of K2 is:", K2)
# Ultimate BOD
e = 2.718
B1 = float(input("BOD at 3rd day 20°C:"))
t = float(input("Time in days for finding B1:"))
E = (1 - e ** (-0.23 * t))
print("The value of E is:", E)
L0 = B1 / E
print("The value of Ultimate BOD (L0) is:", L0)
t1 = float(input("Time in days for finding B2:"))
E1 = (1 - e ** (-K2 * t1))
print("The value of E1 is:", E1)
B2 = L0 * E1
print("The value of BOD at 7th day 25°C is:", B2)
```

output-

```
Decay Coefficient at 20°C:0.23
Temperature of 3rd day BOD:20
Temperature of 7th day BOD:25
The value of K2 is: 0.2893751572825015
BOD at 3rd day 20°C:50
```

Time in days for finding B1:3

The value of E is: 0.49838804582143437

The value of Ultimate BOD (L0) is: 100.32343355585682

Time in days for finding B2:7

The value of E1 is: 0.8680610647811111

The value of BOD at 7th day 25°C is: 87.08686655499413

Q2

```
# Determination of density of sludge removed from aeration tank

M = float(input("Enter the value of initial mass:"))

S = float(input("Enter the value of solid content in sludge (%):"))

Gs = float(input("Enter the value of Specific gravity of sludge solid:"))

Rho_W = float(input("Enter the value of density of water:"))

Ws = (S / 100) * M

m = M - Ws

print("The value of mass of water:", m)

print("The value of Solid Content in sludge:", Ws)

Vw = m / Rho_W

print("The value of Volume of water:", Vw)

Rho_S = Gs * Rho_W

print("The value of Density of solid content in sludge:", Rho_S)

Vs = Ws / Rho_S

print("The value of volume of solid content in sludge:", Vs)

Vt = Vw + Vs

print("The value of total volume of sludge:", Vt)

Rho_SL = M / Vt

print("The value of Density of sludge removed from aeration:", Rho_SL)
```

output-

Enter the value of initial mass:100

Enter the value of solid content in sludge (%):2

Enter the value of Specific gravity of sludge solid:2.2

Enter the value of density of water:1000

The value of mass of water: 98.0

The value of Solid Content in sludge: 2.0

The value of Volume of water: 0.098

The value of Density of solid content in sludge: 2200.0

The value of volume of solid content in sludge: 0.0009090909090909091

The value of total volume of sludge: 0.09890909090909092

The value of Density of sludge removed from aeration: 1011.0294117647057

Assignment 10

Q1

```
# Design of Tension Member (IS 800:2007)
# ---------------- INPUT SECTION ----------------
Tu = float(input("Enter the value of ultimate tensile load Tu (kN): "))
fy = float(input("Enter the value of yield strength of steel fy (MPa): "))
fu = float(input("Enter the value of ultimate strength of steel fu (MPa): "))
fub = float(input("Enter the value of ultimate strength of bolt fub (MPa): "))
Gamma_m0 = float(input("Enter the value of partial factor of safety Gamma_m0: "))
Gamma_m1 = float(input("Enter the value of partial factor of safety Gamma_m1: "))
Gamma_mb = float(input("Enter the value of partial factor of safety Gamma_mb: "))
# ---------------- GROSS AREA ----------------
print("\n--- Gross Area Required ---")
Agreq = 1.1 * Tu * 1000 / fy
print("The gross area required is:", 1.2 * Agreq)
# Section Selection (example ISA 100x65x8)
Ag = float(input("Enter the value of gross area Ag of steel section (mm²): "))
Lcl = float(input("Enter the length of connected leg Lcl (mm): "))
Lol = float(input("Enter the length of outstand leg Lol (mm): "))
t = float(input("Enter the thickness t (mm): "))
# ---------------- BOLTED CONNECTION ----------------
print("\n--- Design of Connections ---")
d = float(input("Enter the nominal diameter of bolt d (mm): "))
do = d + 2  # diameter of bolt hole
print("The diameter of bolt hole is:", do)
# Minimum pitch distance as per IS code
pmin = 2.5 * d
print("The minimum pitch is:", pmin)
```

```
# Edge distance as per IS code
e = 1.5 * do
print("The edge distance is:", e)
nn = int(input("Number of shear planes with threads intercepting shear plane: "))
ns = int(input("Number of shear planes without threads: "))
# Net area of bolt
Anb = 0.78 * (3.1416/4) * d * d
print("Threaded area of bolt (Anb):", Anb)
Asb = 0.7854 * d * d
print("Shank area of bolt (Asb):", Asb)
# Shear capacity of bolt
Vdsb = (fub / (1.732 * Gamma_mb)) * (nn * Anb + ns * Asb) * 1e-3
print("Shear capacity of bolt Vdsb (kN):", Vdsb)
# Bearing strength factors
kb1 = e / (3 * do)
print("Kb1:", kb1)
kb2 = (pmin / (3 * do)) - 0.25
print("Kb2:", kb2)
kb3 = fub / fu
print("Kb3:", kb3)
kb4 = 1
print("Kb4:", kb4)
kb = min(kb1, kb2, kb3, kb4)
print("Kb (governing value):", kb)
# Bearing capacity of bolt
Vdpb = (2.5 * kb * d * t * fu * 1e-3) / Gamma_mb
print("Bearing capacity of bolt Vdpb (kN):", Vdpb)
# Design strength of bolt
Vd = min(Vdsb, Vdpb)
```

```
print("Design strength of bolt Vd (kN):", Vd)
# Number of bolts required
N = Tu / Vd
print("Number of bolts required:", N)
N = int(input("Enter the actual number of bolts provided: "))
# ---------------- STRENGTH CHECKS ----------------
print("\n--- Strength Checks ---")
# 1. Yielding of Gross Section
Tdg = (Ag * fy * 1e-3) / Gamma_m0
print("Tensile strength (Yielding of gross section) Tdg (kN):", Tdg)
# 2. Rupture of Critical Section
Anc = (Lcl - (t/2) - do) * t
print("Net Area of Connecting leg (Anc):", Anc)
Ago = (Lol - (t/2)) * t
print("Gross Area of Outstand leg (Ago):", Ago)
Lc = (N - 1) * pmin
print("Shear Lag distance Lc (mm):", Lc)
bs = 0.6 * Lcl + Lol
print("Shear lag width bs (mm):", bs)
Beta = (fy/fu) * (bs/Lc) if Lc > 0 else 1.0
if Beta > 1.4:
Beta = 1.4
print("Beta factor:", Beta)
Tdn = (0.9 * fu * Anc / Gamma_m1) * 1e-3 + (Beta * Ago * fy / Gamma_m0) * 1e-3
print("Tensile strength due to rupture of critical section Tdn (kN):", Tdn)
# 3. Block Shear
Avg = (pmin * (N - 1) + e) * t
print("Shear area Avg (mm²):", Avg)
```

```
Avn = ((pmin * (N - 1) + e) - (N - 1) * do + 0.5 * do) * t
print("Net shear area Avn (mm²):", Avn)
Atg = Lcl * t
print("Gross tension area Atg (mm²):", Atg)
Atn = (Lcl - 0.5 * do) * t
print("Net tension area Atn (mm²):", Atn)
Tb1 = (((Avg * fy) / (1.732 * Gamma_m0)) + (0.9 * fu * Atn) / Gamma_m1) * 1e-3
print("Block shear strength (mode 1) Tb1 (kN):", Tb1)
Tb2 = ((0.9 * Avn * fu) / (1.732 * Gamma_m1) + (Atg * fy) / Gamma_m0) * 1e-3
print("Block shear strength (mode 2) Tb2 (kN):", Tb2)
Tb = min(Tb1, Tb2)
print("Block shear strength Tb (kN):", Tb)
# Governing Strength
Td = min(Tdg, Tdn, Tb)
print("Design tensile strength of section Td (kN):", Td)
# ---------------- SAFETY CHECK ----------------
if Td > Tu:
print("\n ✅ SAFE: Section is adequate")
else:
print("\n ❌ NOT SAFE: Revise the Section")
```

output-

```
Enter the value of ultimate tensile load Tu (kN): 225
Enter the value of yield strength of steel fy (MPa): 250
Enter the value of ultimate strength of steel fu (MPa): 410
Enter the value of ultimate strength of bolt fub (MPa): 400
Enter the value of partial factor of safety Gamma_m0: 1.1
Enter the value of partial factor of safety Gamma_m1: 1.25
Enter the value of partial factor of safety Gamma_mb: 1.25
```

--- Gross Area Required ---

The gross area required is: 1188.0

Enter the value of gross area Ag of steel section (mm^2): 1257

Enter the length of connected leg Lcl (mm): 100

Enter the length of outstand leg Lol (mm): 65

Enter the thickness t (mm): 8

--- Design of Connections ---

Enter the nominal diameter of bolt d (mm): 20

The diameter of bolt hole is: 22.0

The minimum pitch is: 50.0

The edge distance is: 33.0

Number of shear planes with threads intercepting shear plane: 1

Number of shear planes without threads: 0

Threaded area of bolt (Anb): 245.0448

Shank area of bolt (Asb): 314.16

Shear capacity of bolt Vdsb (kN): 45.273866050808316

Kb1: 0.5

Kb2: 0.5075757575757576

Kb3: 0.975609756097561

Kb4: 1

Kb (governing value): 0.5

Bearing capacity of bolt Vdpb (kN): 65.6

Design strength of bolt Vd (kN): 45.273866050808316

Number of bolts required: 4.969754510195687

Enter the actual number of bolts provided: 5

--- Strength Checks ---

Tensile strength (Yielding of gross section) Tdg (kN): 285.6818181818182

Net Area of Connecting leg (Anc): 592.0

Gross Area of Outstand leg (Ago): 488.0

Shear Lag distance Lc (mm): 200.0

Shear lag width bs (mm): 1.0

Beta factor: 0.38109756097560976

Tensile strength due to rupture of critical section Tdn (kN): 217.0255840354767

Shear area Avg (mm^2): 1864.0

Net shear area Avn (mm^2): 1248.0

Gross tension area Atg (mm^2): 800.0

Net tension area Atn (mm^2): 712.0

Block shear strength (mode 1) Tb1 (kN): 454.776143439009

Block shear strength (mode 2) Tb2 (kN): 394.525803065295

Block shear strength Tb (kN): 394.525803065295

Design tensile strength of section Td (kN): 217.0255840354767

❌ NOT SAFE: Revise the Section