```
#Assignment4

def multiplication_table (x):
  for p in range (1,  26):
     print(f' {x}×{p}={x*p}')
num= int(input('enter number'))
multiplication_table(num)
```

```
enter number8
 8×1=8
 8×2=16
 8×3=24
 8×4=32
 8×5=40
 8×6=48
 8×7=56
 8×8=64
 8×9=72
 8×10=80
 8×11=88
 8×12=96
 8×13=104
 8×14=112
 8×15=120
 8×16=128
 8×17=136
 8×18=144
 8×19=152
 8×20=160
 8×21=168
 8×22=176
 8×23=184
 8×24=192
 8×25=200
```

Assignment 3

```
76
#Guessing Game
import random
secret_number = random.randint(1, 100)
guess = 0
print("I'm thinking of a number between 1 and 100, can you guess ")
while guess != secret_number:
  try:
     guess = int(input("Enter your guess guess: "))
     if guess < secret_number:
       prints("Too low try again")
     elif guess > secret_number:
       print("Too high,  try again ")
     else:
       print("Congratulations you got it right")
  except ValueError:
       print("Invalid input. Please enter a whole number ")
```

```
I'm thinking of a number between 1 and 100, can you guess
Too high,  try again
Too high,  try again
Too high,  try again
Enter your guess guess: 67
Too high,  try again
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
/tmp/ipython-input-2343278621.py in <cell line: 0>()
      6 while guess != secret_number:
      7   try:
----> 8     guess = int(input("Enter your guess guess: "))
      9     if guess < secret_number:
     10       prints("Too low try again")

 1 frames
/usr/local/lib/python3.12/dist-packages/ipykernel/kernelbase.py in
_input_request(self, prompt, ident, parent, password)
   1217             except KeyboardInterrupt:
   1218                 # re-raise KeyboardInterrupt, to truncate traceback
-> 1219                 raise KeyboardInterrupt("Interrupted by user") from None
   1220             except Exception:
   1221                 self.log.warning("Invalid Message:", exc_info=True)

KeyboardInterrupt: Interrupted by user
```

Assignment 5

```
# Petrophysics OOP Example
```

```
class PetrophysicsFormula:
    """Base class for all petrophysics formulas"""
    def calculate(self):
        raise NotImplementedError("This method must be overridden in subclasses")


# ------------------- FORMULAS -------------------

class Porosity(PetrophysicsFormula):
    def __init__(self, Vp, Vb):
        self.Vp = Vp
        self.Vb = Vb

    def calculate(self):
        try:
            return self.Vp / self.Vb
        except ZeroDivisionError:
            print("Error: Bulk volume (Vb) cannot be zero.")
            return None


class WaterSaturation(PetrophysicsFormula):
    def __init__(self, phi, Rw, Rt, a=1.0, m=2.0, n=2.0):
        self.phi = phi
        self.Rw = Rw
        self.Rt = Rt
        self.a = a
        self.m = m
        self.n = n

    def calculate(self):
        try:
            return ((self.a * self.Rw) / ((self.phi ** self.m) * self.Rt)) ** (1 / self.n)
        except ZeroDivisionError:
            print("Error: Division by zero in Archie’s equation.")
            return None


class HydrocarbonSaturation(PetrophysicsFormula):
    def __init__(self, Sw):
        self.Sw = Sw

    def calculate(self):
        return 1 - self.Sw


class BulkDensity(PetrophysicsFormula):
    def __init__(self, Wd, Vb):
        self.Wd = Wd
        self.Vb = Vb

    def calculate(self):
        try:
            return self.Wd / self.Vb
        except ZeroDivisionError:
            print("Error: Bulk volume (Vb) cannot be zero.")
            return None


class FormationFactor(PetrophysicsFormula):
```

```
    def __init__(self, R0=None, Rw=None, phi=None, a=1.0, m=2.0):
        self.R0 = R0
        self.Rw = Rw
        self.phi = phi
        self.a = a
        self.m = m

    def calculate(self):
        try:
            if self.R0 is not None and self.Rw is not None:
                return self.R0 / self.Rw
            elif self.phi is not None:
                return self.a / (self.phi ** self.m)
            else:
                raise ValueError("Insufficient data for Formation Factor")
        except ZeroDivisionError:
            print("Error: Division by zero in Formation Factor calculation.")
            return None


class Permeability(PetrophysicsFormula):
    def __init__(self, phi, Sgv):
        self.phi = phi
        self.Sgv = Sgv

    def calculate(self):
        try:
            return (self.phi ** 3) / (self.Sgv ** 2 * (1 - self.phi) ** 2)
        except ZeroDivisionError:
            print("Error: Invalid Sgv or phi=1 in Kozeny-Carman equation.")
            return None


# ------------------- POLYMORPHISM EXAMPLE -------------------
def compute_formula(formula: PetrophysicsFormula):
    """Demonstrates polymorphism: different formulas share the same interface"""
    return formula.calculate()


# ------------------- TESTING -------------------
if __name__ == "__main__":
    # Create objects
    phi_calc = Porosity(28, 100)
    phi = compute_formula(phi_calc)
    print("Porosity (□):", phi)

    Sw_calc = WaterSaturation(phi, Rw=0.08, Rt=20)
    Sw = compute_formula(Sw_calc)
    print("Water Saturation (Sw):", Sw)

    Sh_calc = HydrocarbonSaturation(Sw)
    print("Hydrocarbon Saturation (Sh):", compute_formula(Sh_calc))

    rho_b_calc = BulkDensity(245, 100)
    print("Bulk Density (ρb):", compute_formula(rho_b_calc))

    F_calc = FormationFactor(Rw=0.08, phi=phi)
    print("Formation Factor (F):", compute_formula(F_calc))
```

```
    k_calc = Permeability(phi, Sgv=0.5)
    print("Permeability (k):", compute_formula(k_calc))
```

```
Porosity (□): 0.28
Water Saturation (Sw): 0.2258769757263128
Hydrocarbon Saturation (Sh): 0.7741230242736872
Bulk Density (ρb): 2.45
Formation Factor (F): 12.755102040816325
Permeability (k): 0.16938271604938276
```

ASSIGNMENT 2

```
import string
import math

# --------------------
# Task 1: Convert all uppercase to lowercase
def task1(s: str) -> str:
    return s.lower()

# --------------------
# Task 2: Swap uppercase ↔ lowercase
def task2(s: str) -> str:
    return s.swapcase()

# --------------------
# Task 3: Remove uppercase letters
def task3(s: str) -> str:
    return ''.join(ch for ch in s if not ch.isupper())

# --------------------
# Task 4: Count uppercase and lowercase
def task4(s: str) -> str:
    upper = sum(1 for ch in s if ch.isupper())
    lower = sum(1 for ch in s if ch.islower())
    return f"Uppercase: {upper}, Lowercase: {lower}"

# --------------------
# Task 5: Remove non-English letters
def task5(s: str) -> str:
    return ''.join(ch for ch in s if ch.isalpha())

# --------------------
# Task 6: Heron’s formula for triangle area
def task6(a: float, b: float, c: float) -> float:
    s = (a + b + c) / 2
    area = math.sqrt(s * (s - a) * (s - b) * (s - c))
    return area

# --------------------
# Task 7: Format names in a table
def task7(names: list):
    print("Formatted Names Table:\n")
    for name in names:
        print(name.ljust(15), name.center(20), name.rjust(15))

# --------------------
```

```
# Task 8: Clean string
def task8(s: str) -> dict:
    cleaned = {}
    cleaned["strip"] = s.strip()
    cleaned["no_punctuation"] = s.translate(str.maketrans('', '', string.punctuation))
    cleaned["no_spaces"] = s.replace(" ", "")
    return cleaned


# ==========================
# TESTING
# ==========================

if __name__ == "__main__":

    print("Task 1:", task1("Hello"))
    print("Task 2:", task2("HeLLo WoRLd"))
    print("Task 3:", task3("HelloWorld"))
    print("Task 4:", task4("EngiNEEr"))
    print("Task 5:", task5("Data-Driven@2025!"))
    print("Task 6: Area =", task6(3, 4, 5))

    names = ["Alice", "Bob", "Charlie"]
    task7(names)

    print("Task 8:", task8("  Hello, World!  "))
```

```
    4, Lowercase: 4


    le:

        Alice                   Alice
         Bob                      Bob
       Charlie                 Charlie
    Hello, World!', 'no_punctuation': '  Hello World  ', 'no_spaces': 'Hello,World!'}
```