ASSIGNMENT 1

Submitted by - **Nanda Krishnan V**

1. What is an Algorithm? What are the features of a good Algorithm.
    a. An algorithm is a step-by-step procedure or set of rules designed to solve a specific problem or perform a particular task.
    b. **The Features of algorithm includes** :
        i. **Correctness** - The algorithm should produce the correct output for all valid inputs.
        ii. **Effectiveness** -
            1. **Time Efficiency**: The algorithm should complete its task in the least possible time, using the fewest number of steps or operations.
            2. **Space Efficiency**: It should use the least amount of memory or storage.
        iii. **Finiteness** - The algorithm should have a finite number of steps and should terminate after a finite amount of time.
        iv. **Unambiguous** - An algorithm is unambiguous when every step is clearly defined and can be executed in exactly one way, ensuring consistent and predictable results.

2. Write an algorithm to find the factorial of a number.
    a. Step 1 - START
    b. Step 2 - DECLARE n, fact= 1 , i = 0
    c. Step3 - CHECK the condition that i <= n goto STEP 4 ELSE goto STEP 5
    d. Step4 - fact= fact* i
    e. Step5 - Display fact
    f. Step6 - STOP

3. Write names of 5 algorithms commonly used.
    a. Dijkstra's Algorithm
    b. Travelling Salesman Problem
    c. Knapsack Problem
    d. Quick Sort
    e. Breadth First Search (BFS)

4. Write an algorithm to find the reverse of a number.
    a. Step 1 - START
    b. Step 2 - Declare rem = 0 , rev = 0 , n
    c. Step 3 - WHILE the condition n != 0 , then goto step 4 else goto step 7
    d. Step 4 - rem = n % 10
    e. Step 5 - rev = rev * 10 + rem
    f. Step 6 - n = n / 10 , then goto step 3
    g. Step 7 - Display rem
    h. Step 8 - STOP

5. Write the algorithm to find the roots of a quadratic equation. also implement it in any programming language.
    a. Step 1 - START
    b. Step 2 - Declare a,b,c,x,m,n,p,e,f
    c. Step 3 - Computer the formula , b**2 - (4*a*c) and store in x
    d. Step 4 - Check the condition that x > 0 , then goto step 5 else goto step 14
    e. Step 5 - Compute the formula (-b+x)/(2*a) and store in m

f. Step 6 - Compute the formula (-b-x)/(2*a) and store in n
g. Step 7 - Display m,n , goto step
h. Step 8 - Check the condition the x == 0 , then goto step 8 else goto step 14
i. Step 9 - Compute -b/(2*a) and store in p
j. Step 10 - Display P and goto step 14
k. Step 11 - Check the condition that x<0 then goto step 12 else goto step 14
l. Step 12 - Compute (-b+xi)/(2*a) and store in e
m. Step 13 - Compute (-b-xi)/(2*a) and store in f
n. Step 14 - STOP

CODE -

```
import math

def find_roots(a, b, c):
    x = b**2 - (4 * a * c)

    if x > 0:
        m = (-b + math.sqrt(x)) / (2 * a)
        n = (-b - math.sqrt(x)) / (2 * a)
        print(f"The roots are real and different: m = {m}, n = {n}")

    elif x == 0:
        p = -b / (2 * a)
        print(f"The root is real and repeated: p = {p}")

    else:
        realPart = -b / (2 * a)
        imaginaryPart = math.sqrt(-x) / (2 * a)
        e = f"{realPart} + {imaginaryPart}i"
        f = f"{realPart} - {imaginaryPart}i"
        print(f"The roots are complex: e = {e}, f = {f}")


a = 1
b = -7
c = 10
find_roots(a, b, c)
```