01/08/25

TASK-2 Implement conditional, control and looping statement

(a) water level monitoring system (if-elif)

Aim:- A dam monitoring system checks the water level and issues warning. write a python program that takes water level (in meters) as input and prints:

* "Low water level" if level < 50
* "Normal water level" if 50 ≤ level ≤ 80
* "High water level - Caution" if level > 80

Algorithm:-

```
1. Start the program
2. prompt the user to input the water level (in meter)
3. Convert the input to numeric value
4. check the water level:
   → If water level is less than 50, print "Low water level".
   → Else if the level is between 50 to 80, print "Normal water level".
   → Else print "High water level - caution"
5. Stop
```

Program:-

```
level = float (input ("Enter the water level (in meters):"))
if level < 50 :
    print ("Low water level :")
elif  50 <= level <= 80
    print ("Normal water level")
else :
    print ("High water level - caution")
```

Output :-

```
Enter the water level (in meters) : 100
High water level - caution
```

10

21/08/25

b) Multiplication table Generator (For Loop)

Aim: A teaching app needs to generate multiplication table. Write a python program that takes a number as input and print its multiplication table upto 10 using a for loop.

(eg: input → 3 → output : 3, 6, 9 .... 30)

Algorithm

```
1. Start the program
2. prompt the user to input a number
3. Convert the input to an integer
4. For each number from 1 to 10 (inclusive)
   → Calculate the product of input number and i.
   → print the product, seperated by spaces or line
     by line.
5. End the program
```

program :-

```
number = int (input ("Enter a number:"))
print ("Multiplication table for", number, " up to 10: ")
for i in range (1, 11):
    print (number * i, end = "  ")
```

Output

```
Enter a number: 2
Multiplication table for 2 upto 10:
  2  4  6  8  10  12  14  16  18  20
```

0/2

01/08/25

Task -2(c) - OTP verification System (while loop)

Aim: An OTP verification system gives user 3 attempts to correct OTP. write a python program that simulates asking for an OTP ("4507") upto 3 times until the correct OTP is entered or attempts are executed.

Algorithm :-

```
1) Start the program
2) Set the correct OTP to "4567"
3) Initialize a variable attempts to 0.
4) Repeat while the number of attempts is less than 3.
   → prompt the user to enter an OTP.
   → If the entered OTP matches the correct OTP.
       → print "OTP Verified Successfully!".
       → Exit the loop
   → Else :
       → print "Incorrect OTP"
       → Increase the attempts by 1.
(5) If the maximum number of attempts (3) is reached and OTP is still incorrect.
   → print "Maximum attempts exceeded. OTP verification failed"
6. End the program.
```

program :

```
Correct_otp = "4567"
 attempts = 0
  max_attempts = 3
While attempts < max_attempts :
      otp = input ("Enter OTP:")
```

```
if otp == correct_ otp :
  print ("OTP verified successfully!")
  break
else :
  print ("Incorrect OTP")
 attempts + = 1
if attempts = = max _attempts and otp ! = correct_otp:
print ("Maximum attempts exceeded. OTP verification
                                      failed")
```

VELTECH	
EX No.	2
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	—
TOTAL (20)	5
SIGN WITH DATE	8/8/25

Result :- Thus the python program to implement Conditional, Control and Looping statements was done Successfully.

8/8/25

Output:-

```
Enter otp : 2345
Incorrect otp

Enter otp : 4567
otp verified successfully!
```