LAB # 06

NESTED STATEMENTS, BREAK AND CONTINUE STATEMENTS

OBJECTIVE

Working on nested statements and control loop iteration using break and continue.

THEORY

Nested Statements:

A Nested statement is a statement that is the target of another statement. **Nested if:**

A Nested ***if*** is an ***if*** statement that is the target of another ***if*** statement. Nested ***if*** statements means an ***if*** statement inside another ***if*** statement.

Syntax:

```
if (condition1):
  # Executes when condition1 is true
if (condition2):
    # Executes when condition2 is true
  # if Block is end here
# if Block is end here
```

Example:

```
#using nested if
x=int(input("enter number="))
y=int(input("enter 2nd number="))
if x > 2:     if y > 2:
        z = x + y
print("z is", z)  else:
print("x is", x)
```

Output:

```
>>> %Run task1.py
enter number=3
enter 2nd number=8
       z is 11 >>>
```

Nested loops:

Nested loops consist of an outer loop and one or more inner loops. Each time the outer loop repeats, the inner loops are reinitialize and start again.

Example:

```
height=int(input("Enter height:
")) for row in range(1, height):
for column in range(1,height):
        print(row, end=" ")
print()
```

Output:

```
>>> %Run task2.py
Enter height: 7
1 1 1 1 1 1
2 2 2 2 2 2
3 3 3 3 3 3
4 4 4 4 4 4
5 5 5 5 5 5
6 6 6 6 6 6
```

Keywords break and continue:

The break and continue keywords provide additional controls to a loop.

The Break Statement:

The keyword ***break*** in a loop to immediately terminate a loop. Listing example presents a program to demonstrate the effect of using ***break*** in a loop.

Syntax: break

Example:

```
# Use of break statement inside
loop for word in "string":     if
word == "i":
        break
print(word) print("The
end")
```

Output:

```
>>> %Run task3.py'
s t r The end >>>
```

The continue Statement:

The ***continue*** statement breaks out of the current iteration in the loop.

Syntax: continue

Example:

```
# Program to show
the use of
continue statement
inside loops for
val in "string":
if val == "i":
continue
    print(word)
print("The end"
```

Output:

```
>>> %Run task4.py'
s t r n g
The end
>>>
```

EXERCISE

A. Point out the errors, if any, in the following Python programs.

1. Code

```
prompt = "\nPlease enter the name of a city you have visited:"
prompt+="\n(Enter 'quit' when you are finished.)"  while
True:     city = str(input(prompt))     if city == quit:
break;     else:         print("I'd love to go to " ,
city.title() , "!")
```

Output

2. **Code**

```
if x>2:
if y>2:
z=x+y
      print(“z is”, y)
else
      print(“x is”, x)
```

Output

2. Code

```
balance = int(input("enter your
balance1:")) while true:  if balance
<=9000:     continue;
balance = balance+999.99 print("Balance
is", balance)
```

Output

B. What will be the output of the following programs:

1. Code

```
i = 10 if (i
== 10):
    #  First if statement      if (i <
15):          print ("i is smaller
than 15")
    # Nested - if statement
    # Will only be executed if statement above
    # it is true      if (i < 12):
print ("i is smaller than 12 too")
else:          print ("i is greater than
15")
```

Output

2. Code

```
i = 1 j = 2 k = 3 if i
> j:    if i > k:
print('A')  else:
print('B')
```

```
i = 1 j = 2 k = 3
if i > j:     if
i> k:
print('A')
else:
        print('B')
```

Output

3. Code

```
# nested for loops  for
i in range(0, 5):  for
j in range(i):
        print(i, end=' ')
print()
```

Output

C. Write Python programs for the following:

1. Write a program to add first seven terms twice of the following series:

$$\frac{1}{1!}+\frac{2}{2!}+\frac{3}{3!}+\cdots$$

2. Write a program to print all prime numbers from 900 to 1000.
   [Hint: Use nested loops, break and continue]

3. Write a program to display multiplication table(1-5) using nested looping
   Sampled output:[hint: '{ } ' .format(value)]
   02 X 01 = 02