Title: Programming Language I

Course Code: CSE 110

Assignment no: 3 (Loops)

Class Tasks	6
Evaluation Tasks	2
Home Tasks	8
Total	16

Class Task 1:

Write the python program, which prints the following sequences of values in loops:

a) 24, 18, 12, 6, 0, -6
b) 18, 27, 36, 45, 54, 63

==========

Hint (1): You may use a while loop for solving these problems.

Hint (2): We are already familiar with the print() function. But, when we use it to print any value, it automatically adds an additional newline after each print statement.

For example:
```
print(1)
print(2)
```

Output:
```
1
2
```

==========

To solve this problem, in Python3, we can add an extra argument (end = " ") in the print function which tells the program to skip printing the additional newline.

For example:
```
print(1, end ="")
print(2)
```

Output: (prints the next output right to the previous one)
```
12
```

==========

In Task-1(a), the loop counter may be initialized at 24 and then the loop should terminate when the loop counter reaches -6. The difference between the first two values is 24 - 18 = 6. So, the loop counter value is getting decremented by 6 in every iteration for the given sequence here.

For your understanding, the code for Task 1(a) is done for you.

```
# a) 24, 18, 12, 6, 0, -6

# initializing loop counter
counter = 24

# loop structure
while counter >= -6:

    #inside loop body
    if counter == -6:
        print(counter, end = "")
    else:
        print(counter, end = ", ")

    counter = counter - 6 #updating loop counter
    #inside loop body

#outside loop body
```

Class Task 2:

Write a Python code that will calculate the **value of y if the expression** of y is as follows (n is the input):

$y=1^2-2^2+3^2-4^2+5^2.....+n^2$

==========

Sample Input 1:
5

Sample Output 1:
15

Explanation: y = 1 - 4 + 9 - 16 + 25 = 15

==========

Sample Input 2:
10

Sample Output 2:
-55

Explanation: y = 1 - 4 + 9 - 16 + 25 - 36 + 49 - 64 + 81 - 100 = -55

==========

Sample Input 3:
20

Sample Output 3:
-210

Class Task 3:

Write a Python program which takes a number and prints the digits from the unit place, then the tenth, then hundredth, etc. (Right to Left)

[Consider the input number to be an INTEGER. You are not allowed to use String indexing for solving this task]

Example: If the user gives 32768, then print 8, 6, 7, 2, 3

==========

Hint (1): The input() function, converts the input data to String data type by default. Therefore, please type cast it to an integer before proceeding further.

Hint (2): First to get the digit from the right side, we can take the remainder of the number using modulus (%) operator i.e. mod 10 to get the rightmost digit and print it. For dropping the last digit, we can perform floor division by 10 on the number and then continue the same to print the other digits as shown below.

32768 % 10 = 8

32768 // 10 = 3276

Then,

3276 % 10 = 6

3276 // 10 = 327

and so on

327 % 10 = 7

327 // 10 = 32

32 % 10 = 2

32 // 10 = 3

3 % 10 = 3

3 // 10 = 0

Done! When the number becomes 0 you can end your loop.

Class Task 4:

Write a Python program that takes a number as input from the user and prints the divisors of that number as well as how many divisors the number has.

==========

SSample Input 1:
6

Sample Output 1:
1

2

3

6
Total 4 divisors.

==========

Sample Input 2:
121

Sample Output 2:
1

11

121
Total 3 divisors.

Class Task 5:

Write a python program that prints a rectangle of size M (height/line numbers) and N (length/column numbers) using incrementing numbers where M and N will be given as input. Please look at the examples below for clarification.

Hint: You may need to use nested loops and print the loop counter variable in one of the loops.

==========

Sample Input 1:
4
6

Sample Output 1:
123456
123456
123456
123456

Explanation: The user has given 4 rows/lines and 6 columns as input. Therefore, we have 4 lines in our output here and in each line, the column number is printed from 1 through to 6 for the 6 columns.

==========

Sample Input 2:
3
2

Sample Output 2:
12
12
12

Explanation: Our user input this time is 3 rows/lines and 2 columns. So, our output has 3 lines and, in each line, the column number 1 and 2 are printed next to each other as the user only wants 2 columns here.

Class Task 6 (Tracing):

Illustrate the outputs of the following statements. Your answer will not be accepted without the workings.

```
1  x = 0
2  y = 0
3  sum = 0
4  p = 0.0
5  while (x < 10):
6      y = x // 2
7      while (y < x):
8          p = (x + 10.0) / 2
9          sum = (sum % 2) + x - y * 2 + int(p)
10         print(sum)
11         y = y + 2
12     if (x > 5):
13         x += 1
14     else:
15         x += 2
```

Output

Class Evaluation Task 1:

Write the Python programs, that print the following sequences of values in loops:

18, -27, 36, -45, 54, -63

Hints:

print(5 * (-1)) gives output -5
print("-" + str(5)) gives output -5

Class Evaluation Task 2:

Write a Python code that will read 5 numbers from the user. Your program should print the first number, the product of the first 2 numbers, the product of the first 3 numbers, and so on up to the product of 5 numbers.

==========

Sample Input 1:
-10
-2
3
2
-1

Sample Output 1:
-10
20
60
120
-120

Explanation:
When the user enters -10 at first, we print -10, then
The user enters -2, (-10 * -2) = 20, so our output is 20, then
The user enters 3, (20 * 3) = 60, hence our output is 60, then
The user enters 2, and we have (60 * 2) = 120, we print 120, and finally
The user enters -1, the final product is -120 which is printed as the last output.

Home Task 1:

Write a Python code for the following:

- Ask the user to enter a Number, **N.**
- From **1 to N (inclusive)**, display the summation of all numbers that are multiples of **either 7 or 9 but not a multiple of both**.

==========

Sample Input 1:
30

Sample Output 1:
124

Explanation: The summation of multiples of either 7 or 9 but not both from 1 up to 30 is 7 + 9 + 14 + 18 + 21 + 27 + 28 = 124

==========

Sample Input 2:
75

Sample Output 2:
385

Explanation: The summation of multiples of either 7 or 9 but not both from 1 up to 75 is 7 + 9 + 14 + 18 + 21 + 27 + 28 + 35 + 36 + 42 + 45 + 49 + 54 + 56 + 70 + 72 = 583

Home Task 2:

Write a Python program that takes a number from the user and prints its digits from left to right.

[Consider the input number to be an INTEGER. You are not allowed to use String indexing for solving this task]

Example: if the user gives 32768, then print 3, 2, 7, 6, 8

==========

Hint(1): The input() function takes the input data as String data type by default. Please convert it to an integer before starting your code for the task.

Hint(2):

Step 1: First, count how many digits are there in the input number
Step 2: Then, calculate 10 to the power (number of digits - 1).

Step 3 with explanation: Say, the input given by the user as in our example, 32768 has 5 digits, so calculating 10 to the power 4 gives us 10000. Then floor dividing 32768 by 10000, we can get 3.

Proceeding further, the remainder of 32768 by 10000 (32768 % 10000) gives us 2768. This time to get 2 we need to floor divide 2768 by 1000 which is basically our 10000/10. Again, taking the remainder of 2768 by 1000 gives us 768 which we then divide by 100 (i.e. 1000/10) and keep on doing this until there are no more digits left (zero).

To summarize and clarify:
Loop 1: First, we count digits, say 5 in this case for 32768
Loop 2: Then, we calculate 10 to the power 4 (5-1), that is 10000.
Loop 3: Then we keep repeating the three steps of floor dividing, modulus and dividing by 10 as demonstrated below.

32768 // 10000 = 3
32768 % 10000 = 2768
10000 // 10 = 1000

2768 // 1000 = 2
2768 % 1000 = 768
1000 // 10 = 100

768 // 100 = 7
768 % 100 = 68
100 // 10 = 10

68 // 10 = 6
68 % 10 = 8
10 // 10 = 1

8 // 1 = 8
8 % 1 = 0
1 // 10 = 0

Done. Loop ends as the number has become 0.

Home Task 3:

Write a Python program that asks the user for an integer number and tells if it is a prime number or a perfect number or neither.

Note: A number cannot be both prime and perfect.

Prime Number: If a number has only two divisors, (1 and itself), then it is a prime number. Else, then it is not a prime number.

Perfect Number: A number is said to be a perfect number if the sum of its divisors, including 1 but not the number itself is equal to that number.

Hint: You may take help from Class Task 4 for finding the divisors.

Sample Input 1:
6

Sample Output 1:
6 is a perfect number

Explanation:
6 has 4 divisors: 1, 2, 3, and 6.
If we add all divisors of 6 except 6 itself, 1 + 2 + 3 = 6. The sum of the divisors excluding the number itself sums up to the number, therefore "6 is a perfect number" is printed.

Sample Input 2:
11

Sample Output 2:
11 is a prime number

Explanation: 11 has only 2 divisors: 1 and 11.

Sample Input 3:
33

Sample Output 3:
33 is not a prime or perfect number

Explanation:
33 has 4 divisors: 1, 3, 11, and 33. So it is not a prime number.
If we add all divisors except 33 itself, 1 + 3 + 11 = 15. The sum is not equal to the number, therefore, 33 is not a perfect number.

Home Task 4:

Write a Python program that asks the user for a quantity, then takes that many numbers as input and prints the maximum, minimum and average of those numbers.

[Please note that you CANNOT use max, min built-in functions]

[Also, you DO NOT need to use lists for this task]

==========

Example: If the user enters 5 as an input for quantity and the enters the 5 numbers, 10, 4, -1, -100, and 1.
The output of your program should be: “Maximum 10”, “Minimum -100”, “Average is -17.2” as shown below.

Input:
5

10

4

-1

-100

1

Output:
Maximum 10
Minimum -100
Average is -17.2

Explanation: Average calculation: (10 + 4 + (-1) + (-100) + 1)/5 = -86/5 = -17.2

Home Task 5:

Write a python program that prints a right-angled triangle of height N using incrementing numbers where N will be given as input.

Hint: You may need to use nested loops. Try to think up to which point the inner loop should run.

==========

Sample Input 1:
4

Sample Output 1:
1
12
123
1234

Explanation: For an input of 4, we have 4 rows/lines where in each line, the respective column number is printed sequentially up to the line/row number. So, in line number 1, we have 1 only. In line 2, 12 is printed. In line 3, we have 123 and so on.

==========

Sample Input 2:
5

Sample Output 2:
1
12
123
1234
12345

Explanation: Numbers are printed sequentially up to the line number for each of the lines.

Home Task 6:

Illustrate the outputs of the following statements. Your answer will not be accepted without the workings.

```
1  x = 0
2  y = 0
3  q = 0
4  sum = 0
5  while (x < 10):
6     p = 0.0
7     p = x + y - sum + int(5 / 3) / 3.0 %  2 ** 3
8     y = x // 5
9     while (y  <  x):
10        q = (x + 10.0) / 2
11        sum =  (sum % 2) +  y  +  int(p)
12        print(sum)
13        y = y + 4
14    if (x  >  5):
15        x += 3
16    else:
17        x += 2
18        print(int(p) + q)
19 print(sum - q)
```

Output

Home Task 7:

Illustrate the outputs of the following statements. Your answer will not be accepted without the workings.

```
1   test = 1
2   j = 0
3   k = 100
4   while (k > 0):
5     while ( j < k ):
6       test += k + j -21
7       print(1 + int (test / 2))
8       j += 10
9     k -= 10
10    test = 1
```

Output

Home Task 8:

Illustrate the outputs of the following statements. Your answer will not be accepted without the workings.

```
1  p = 0
2  r = 0
3  while p < 7:
4     q = 1
5     while q <= p:
6         q = q + 1
7         if q == 4 or q == 5:
8             continue
9             print(p - q * 2)
10        elif p==6:
11            break
12            print(p % q)
```

```
13          else:
14              r = p * q - int(21 / q)
15              print(r)
16      p = p + 1
```

Output