Hands-on Exercise-III

(Loops)

PART-A

1. Write a Python program to input a string message and display it 10 times in the following manner. Use a *while* loop. Let the string message be “Hello”.

```
Enter a message
Hello

1st Hello
2nd Hello
3rd Hello
4th Hello
5th Hello
6th Hello
7th Hello
8th Hello
9th Hello
10th Hello
```

2. Write a Python program that gets an integer from the user. Count from 0 to that number. Use a *for* loop to do it.

```
Count to: 20
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

3. Write a Python program that gets three integers from the user. Count from the first number to the second number in increments of the third number. Use a *for* loop to do it.

```
Count from: 4
Count to: 13
Count by: 3
 4 7 10 13
```

4. Write a Python program that uses a *for* loop. With the loop, make the variable x go from -2 to 2, counting by 0.5. (This means that x can't be an int.)

```
-2.0
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
2.0
```

5. Write a Python program that, using one for loop and one if statement, prints the integers from 1,000 to 2,000 with five integers per line. Hint: Use the % operation.

6. Write a Python program to print the following output using loop.

```
1
121
1213121
121312141213121
1213121412131215121312141213121
```

7. If we list all the natural numbers below 10 that are multiples of 3 or 5, we get 3, 5, 6 and 9. The sum of these multiples is 23. Write a Python program to find the sum of all the multiples of 3 or 5 below 1000.

8. Write a Python program to print the multiplication table of a number entered by the user.

```
Enter a no. for which you want to find the multiplication table: 8
8x1=8
8x2=16
8x3=24
8x4=32
8x5=40
8x6=48
8x7=56
8x8=64
8x9=72
8x10=80
```

9. Write a Python program to find the difference between the sum of the squares of the first one hundred natural numbers and the square of the sum.

The sum of the squares of the first ten natural numbers is,
$1^2 + 2^2 + ... + 10^2 = 385$

The square of the sum of the first ten natural numbers is,
$(1 + 2 + ... + 10)^2 = 55^2 = 3025$

Hence the difference between the sum of the squares of the first ten natural numbers and the square of the sum is 3025 – 385 = 2640.

10. Write a Python program called FunctionGrowth that prints a table of the values log N, N, N log N, N^2, N^3, and 2^N for N = 16, 32, 64, ..., 2048. Use tabs (\t characters) to line up columns.

11. An integer n is divisible by 9 if the sum of its digits is divisible by 9. Write a Python program to display each digit, starting with the rightmost digit.

Your program should also determine whether or not the number is divisible by 9. Test it on the following numbers:

```
n = 154368
n = 621594
n = 123456
```

Hint: Use the % operator to get each digit; then use // to remove that digit. So 154368 % 10 gives 8 and 154368 // 10 gives 15436.

```
The next digit extracted should be 6, then 3 and so on.
```

12. Write a Python program to print largest power of two less than or equal to N.

13. Write a Python program to print the below given pattern using while loop as well as for loop in two different programs.

```
* * * * *
* * * * *
* * * * *
* * * * *
```

14. Write the Python programs to print the following four patterns using for loop using four different programs.

```
(a)            (b)            (c)            (d)

*              1              1              1
* *            1 2            2 2            2 3
* * *          1 2 3          3 3 3          4 5 6
* * * *        1 2 3 4        4 4 4 4        7 8 9 10
* * * * *      1 2 3 4 5      5 5 5 5 5      11 12 13 14 15
```

15. Write a Python program to print the following pattern using nested loops.

```
* * * * * * * * * * 1
* *   *   *   *   * 2
*   *     *     *   3
* *   *       *     4
*       *         * 5
* * *     *         6
*           *       7
* *   *       *     8
*   *           *   9
* *     *         * 10
```

PART-B

1. Write a Python program that takes the value of N through keyboard and prints a table of the power of 2 that are less than or equal to 2^N.

```
Enter a number
5

0     1
1     2
2     4
3     8
4     16
5     32
```

2. Given a set of n student's examination marks (in the range 0 to 100). Write a Python program to count the number of students that passed the examination. A pass is awarded for all marks of 40 and above.

3. Write a Python program that displays all the numbers from 100 to 1,000, ten per line, that are divisible by 5 and 6. Numbers are separated by exactly one space.

4. Write a Python program that reads an unspecified number of integers, determines how many positive and negative values have been read, and computes the total and average of the input values (not counting zeros). Your program ends with the input 0. Display the average as a floating-point number. Here is a sample run:

```
Enter an integer, the input ends if it is 0: 1 2 -1 3 0
The number of positives is 3
The number of negatives is 1
The total is 5.0
The average is 1.25

Enter an integer, the input ends if it is 0: 0
No numbers are entered except 0
```

5. Given a set of *n* numbers. Write a Python program that adds these numbers and returns the resultant sum and compute the average. Assume *n* is greater than or equal to zero.

6. Write a Python program to compute the harmonic mean. The harmonic mean is defined by

$$H = \frac{n}{\sum_{i=1}^{n} \left({}^{1}\!/_{a_i} \right)}$$

7. Write a Python program to compute the sum of the first *n* terms (*n>=1*) of the series.

S=1-3+5-7+9- ………

8. Input a number *n*, write a Python program to compute n factorial (written as *n!*) where *n*>=0.

9. For a given *x* and a given *n*, write a Python program to compute $x^n/n!$.

10. Write a Python program to evaluate the function sin(x) as defined by the infinite series expansion.

$$sin\ (x) = x - x^3/3! + x^5/5! - x^7/7! + ...$$

The acceptable error for computation is 10^{-6}.

11. Write a Python program to evaluate the function cos(x) as defined by the infinite series expansion.

$$cos\ (x) = 1 - x^2/2! + x^4/4! - x^6/6! + ...$$

The acceptable error for computation is 10^{-6}.

12. Assume that x is a positive variable of type double. Write a Python code fragment that uses the Taylor series expansion to set the value of sum to $e^x = 1 + x + x^2/2! + x^3/3! + \ldots\ldots$

13. Write a Python program to generate and print the first *n* terms of the Fibonacci sequence where *n>=1.*

The first few terms are:

0, 1, 1, 2, 3, 5, 8, 13, ...

Each term beyond the first two is derived from the sum of its two nearest predecessors i.e. a new term in the series (Except the first two) is found by the following formula.

new term=preceding term +term before the preceding term

Let us define:

c as new term

b as the preceding term

a as the term before the preceding term

So*, c=b+a*

Your program should handle for all positive values of n.

Example: If n=1, it will display as: Fibonacci Series is: 0

If n=2, it will display as: Fibonacci Series is: 0, 1

If n=3, it will display as: Fibonacci Series is: 0, 1, 1 ….

If n=10, it will display as: Fibonacci Series is: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34

14. Write a Python program to generate and print the first n terms of the Fibonacci numbers using an efficient algorithm. In this case, you need to find a pair of Fibonacci terms, in each iteration and display them and adjust the preceding term b and the term before the preceding term a. Your program should handle all positive values of n.

Example:

If n=10, it will display as: Fibonacci Series is: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34

If n=11, it will display as: Fibonacci Series is: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55

15. Suppose you save $100 each month into a savings account with the annual interest rate 5%.
So, the monthly interest rate is 0.05 / 12 = 0.00417.
After the first month, the value in the account becomes
100 * (1 + 0.00417) = 100.417
After the second month, the value in the account becomes
(100 + 100.417) * (1 + 0.00417) = 201.252
After the third month, the value in the account becomes
(100 + 201.252) * (1 + 0.00417) = 302.507
and so on.

Write a Python program that prompts the user to enter an amount (e.g., 100), the annual interest rate (e.g., 5), and the number of months (e.g., 6) and displays the amount in the savings account after the given month.

16. Write a Python program that accepts a positive integer n and reverses the order of its digits.

17. Write a Python program to compute the square root of a number using Newton's method.

18. Using Newton's method, write a Python program that takes integers N and k as command-line arguments and prints the kth root of N.

19. Write a Python program that puts the binary representation of a positive integer N into a String *s*.

20. Write a Python program that reads an integer and displays all its smallest factors in increasing order. For example, if the input integer is 120, the output should be as follows: 2, 2, 2, 3, 5.

21. Write a Python program GCD that finds the greatest common divisor (gcd) of two integers using Euclid's algorithm, which is an iterative computation based on the following observation: if *x* is greater than *y*, then if *y* divides *x*, the gcd of *x* and *y* is *y*; otherwise, the gcd of *x* and *y* is the same as the gcd of *x % y* and *y*.

22. Write a Python program to check a number *n* is prime or not. The number to be inputted through keyboard.

23. Write a Python program called PrimeCounter that takes a command line argument N and finds the number of primes less than or equal to N.

24. Write a Python program that takes a command-line argument N and prints out all integers less than or equal to N that can be expressed as the sum of two cubes in two different ways. In other words, find distinct positive integers *a, b, c*, and *d* such that $a^3 + b^3 = c^3 + d^3$. Use four nested for loops.

PART-C

1. Write a Python program that reads a list of numbers and makes a count of the number of negatives and the number of non-negative members in the set.

2. Write a Python program to compute the sum of the square of n numbers. That is,

$$s = \sum_{i=1}^{n} (a_i)^2$$

3. Write a Python program to generate the first n terms of the sequence without using multiplication.

1 2 4 8 16 32 ……….

4. Write a Python program to prints out n values of the sequence

1 -1 1 -1 1 -1 ………

5. For a given *n*, write a Python program to compute *1/n!.*
6. Write a Python program to determine whether or not a number *n* is a factorial number.
7. For some integer *n*, write a Python program to find the largest factorial number present as factor in *n*.
8. Write a Python program to simulate multiplication by addition. Your program should accept as input two integers (they may be zero, positive, or negative).
9. Write a Python program to find the sum of the first n terms of the series

$$f_s=0!+1!+2!+3!+........+n! \quad (n>=0)$$

10. The first few members of the Lucas sequence which is a variation on the Fibonacci sequence are:

1 3 4 7 11 18 29 ....

Write a Python program to generate the Lucas sequence.

11. Given a=0, b=1 and c=1 are the first three numbers of some sequence. All other numbers in the sequence are generated from the sum of their three most recent predecessors. Write a Python program to generate this sequence.
12. Given two numbers d and e are suspected of being consecutive members of the Fibonacci sequence. Write a Python program that will refute or confirm this conjecture.
13. Write a Python program to generate the sequence where each member is the sum of the adjacent factorials.

$f_3=1! + 0!$

$f_4=2! + 1!$

$f_5=3! + 2!$

Note that by definition 0! = 1.

14. Write a Python program that counts the no of digits in an integer.
15. Write a Python program to compute the sum of the digits in an integer.
16. Write a Python program that reads in a set of n single digits and converts them into a single decimal integer. For example, the program should convert the set of 5 digits {2, 7, 4, 9, 3} to the integer 27493.
17. Input an integer n. Write a Python program that will find the smallest exact divisor other than one.

18. Write a Python program to find the list of all exact divisors of a given positive integer n using an efficient algorithm.

19. For an integer in the range 1 to 100, write a Python program to find the number that has the most divisors.

20. Write a Python program to compute GCD which does not use either division or mod operator.

21. Write a Python program to compute the smallest common multiple (SCM) of two non-zero positive integer's n and p.

```
(Note: The SCM is defined as the smallest integer m such that n and p
divide exactly into m.)
```

22. Write a Python program to compute the smallest common divisor other than one of two positive non-zero integers.

23. Given some integer x, write a Python program to compute the value of x^n where n is a positive integer considerably greater than 1.

24. Input a number n. Write a Python program to generate the nth member of the Fibonacci sequence using an efficient algorithm.

25. Write a Python program to find all common prime divisors of two integers.

26. Amicable numbers are pair of numbers each of whose divisors add to the other number. Example: The smallest pair of amicable numbers is (220, 284). They are amicable because the proper divisors of 220 are 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 and 110, of which the sum is 284; and the proper divisors of 284 are 1, 2, 4, 71 and 142, of which the sum is 220.

Note: 1 is included as a divisor but the numbers are not included as their own divisors.

Write a Python program that tests whether a given pair of numbers is amicable numbers or not.

27. A perfect number is one whose divisors add up to the number.

Example: The first perfect number is 6, because 1, 2, and 3 are its proper divisors, and 1+2+3=6

Write a Python program that prints all perfect numbers in between 1 and 500.

****