MODULE-2

PPT-1

The for Statement

- A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).
- This is less like the for keyword in other programming languages, and works more like an iterator method as found in other object-orientated programming languages.
- With the for loop we can execute a set of statements, once for each item in a list, tuple, set etc.

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
```

- Even strings are iterable objects, they contain a sequence of characters:

```
for x in “priyanka":
  print(x)
```

The break Statement

- With the break statement we can stop the loop before it has looped through all the items:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
  if x == "banana":
    break
```

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  if x == "banana":
    break
  print(x)
```

The continue Statement

- With the continue statement we can stop the current iteration of the loop, and continue with the next:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  if x == "banana":
    continue
  print(x)
```

The range() Function

- To loop through a set of code a specified number of times, we can use the range() function,
- The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and ends at a specified number.

```
for x in range(6):
  print(x)
```

Note that range(6) is not the values of 0 to 6, but the values 0 to 5.

The range() function defaults to 0 as a starting value, however it is possible to specify the starting value by adding a parameter: range(2, 6), which means values from 2 to 6 (but not including 6):

```
for x in range(2, 6):
  print(x)
```

- The range() function defaults to increment the sequence by 1, however it is possible to specify the increment value by adding a third parameter: range(2, 30, **3**):

Example: Increment the sequence with 3 (default is 1):

```
for x in range(2, 30, 3):
  print(x)
```

- The else keyword in a for loop specifies a block of code to be executed when the loop is finished:

```
for x in range(6):
  print(x)
else:
  print("Finally finished!")
```

Nested Loops

- A nested loop is a loop inside a loop. The "inner loop" will be executed one time for each iteration of the "outer loop":

```
adj = ["red", "big", "tasty"]
fruits = ["apple", "banana", "cherry"]
for x in adj:
  for y in fruits:
    print(x, y)
```

- for loops cannot be empty, but if you for some reason have a for loop with no content, put in the pass statement to avoid getting an error.

```
for x in [0, 1, 2]:
  pass
```

- To iterate over the indices of a sequence, you can combine range() and len() as follows:

```
a = ['Mary', 'had', 'a', 'little', 'lamb'] for i in range(len(a)):
print(i, a[i])
```

More features—

```
sum(range(4)) # 0 + 1 + 2 + 3
list(range(4)) # [0, 1, 2, 3]
```

Print all prime numbers lying between 2 and 10

```
for n in range(2, 11):
    for x in range(2, n):
        if n % x == 0:
            print(n, 'equals', x, '*', n//x)
            break
    else:
        print(n, 'is a prime number')
```

Practice question

1. Write a program to display the n terms of odd natural number and their sum.
2. Write a program to display the n terms of square natural number and their sum.
3. Write a program to check whether a number is prime number.
4. A five-digit number is entered through the keyboard. Write a program to obtain the reversed number and to determine whether the original and reversed numbers are equal or not. If equal then print number is palindrome otherwise not palindrome.
5. Write a program to print all prime numbers from 1 to 300. (Hint: Use nested loops, break and continue).
6. Write a program to convert a decimal numbers from 1 to n into binary numbers.
7. Write a program to display the first n terms of Fibonacci series. (Fibonacci series 0 1 2 3 5 8 13 .....)
8. 19.Write a program to display the multiplication table vertically from 1 to n (suppose n=10).