PYTHON – ASSIGNMENT 3

1) How does break, continue and pass work?

The **break** statement is used inside the loop to exit out of the loop. In Python, when a break statement is encountered inside a loop, the loop is immediately terminated, and the program control transfer to the next statement following the loop.In simple words, A break keyword terminates the loop containing it. If the break statement is used inside a nested loop (loop inside another loop), it will terminate the innermost loop.

The **continue** statement skip the current iteration and move to the next iteration. In Python, when the continue statement is encountered inside the loop, it skips all the statements below it and immediately jumps to the next iteration.In simple words, the continue statement is used inside loops. Whenever the continue statement is encountered inside a loop, control directly jumps to the start of the loop for the next iteration, skipping the rest of the code present inside the loop's body for the current iteration. In some situations, it is helpful to skip executing some statement inside a loop's body if a particular condition occurs and directly move to the next iteration.

The **pass** is the keyword In Python, which won't do anything. Sometimes there is a situation in programming where we need to define a syntactically empty block. We can define that block with the pass keyword.A pass statement is a Python null statement. When the interpreter finds a pass statement in the program, it returns no operation. Nothing happens when the pass statement is executed.It is useful in a situation where we are implementing new methods or also in exception handling. It plays a role like a placeholder.

2) What is the difference between list and tuples in Python?

List and Tuple in Python are the class of data structure. The list is dynamic, whereas the tuple has static characteristics. **List** is just like the arrays, declared in other languages. Lists need not be homogeneous always which makes it the most powerful tool in Python. In Python, the list is a type of container in Data

Structures, which is used to store multiple data at the same time. Lists are a useful tool for preserving a sequence of data and further iterating over it.
Tuple is also a sequence data type that can contain elements of different data types, but these are immutable in nature. In other words, a tuple is a collection of Python objects separated by commas. The tuple is faster than the list because of static in nature.

Difference Between List and Tuple in Python:

LIST	TUPLE
Lists are mutable	Tuples are immutable
Implication of iterations is Time-consuming	The implication of iterations is comparatively Faster
The list is better for performing operations, such as insertion and deletion.	Tuple data type is appropriate for accessing the elements
Lists consume more memory	Tuple consume less memory as compared to the list
Lists have several built-in methods	Tuple does not have many built-in methods.
The unexpected changes and errors are more likely to occur	In tuple, it is hard to take place.

3) What are functions in Python?

Python Functions is a block of related statements designed to perform a computational, logical, or evaluative task. The idea is to put some commonly or repeatedly done tasks together and make a function so that instead of writing the same code again and again for different inputs, we can do the function calls to

reuse code contained in it over and over again. Functions can be both built-in or user-defined. It helps the program to be concise, non-repetitive, and organized.

4) What is a lambda function?

Lambda Function, also referred to as 'Anonymous function' is same as a regular python function but can be defined without a name. While normal functions are defined using the def keyword, anonymous functions are defined using the lambda keyword. However, they are restricted to single line of expression. They can take in mutliple parameters as in regular functions.

5) How can you generate random numbers in Python?

Python can generate random numbers by using the random module. In the below examples we will first see how to generate a single random number and then extend it to generate a list of random numbers.

Generating a Single Random Number

The random() method in random module generates a float number between 0and 1

```
import random
n = random.random()
print(n)
```

Running the above code gives us the following result –

```
0.2112200
```

Generating Number in a Range

The randint() method generates a integer between a given range of numbers.

```
import random
n = random.randint(0,22)
print(n)
```

Running the above code gives us the following result −

```
2
```

Generating a List of numbers Using For Loop

We can use the above randint() method along with a for loop to generate a list of numbers. We first create an empty list and then append the random numbers generated to the empty list one by one.

```
import random
randomlist = []
for i in range(0,5):
n = random.randint(1,30)
randomlist.append(n)
print(randomlist)
```

Running the above code gives us the following result –

```
[10, 5, 21, 1, 17]
```

Using random.sample()

We can also use the sample() method available in random module to directly generate a list of random numbers.Here we specify a range and give how many random numbers we need to generate.

```
import random
#Generate 5 random numbers between 10 and 30
randomlist = random.sample(range(10, 30), 5)
print(randomlist)
```

Running the above code gives us the following result –

```
[16, 19, 13, 18, 15]
```

6) What is the difference between range & xrange?

Both range() and xrange() are built-in functions in Python that are used to generate integers or whole numbers in a given range . The range() and xrange() comparison is relevant only if you are using both Python 2 and Python 3. It is because the

range() function in python 3 is just a re-implementation of the xrange() of python 2. It actually works the same way as the xrange does.

Here we will compare both based on consumption of memory, return type, speed and operations-

Parameters	**Range()**	**Xrange()**
Consumption of Memory	Since range() returns a list of elements, it takes more memory	In comparison to range(), it takes less memory.
Return type	It returns a list of integers.	It returns a generator object.
Speed	Its execution speed is slower.	Its execution speed is faster.
Operations	Since it returns a list, all kinds of arithmetic operations can be performed.	Such operations cannot be performed on xrange().

7) How do you write comments in Python?

To write a comment in Python, simply put the hash mark # before your desired comment:

```
# This is a comment
```

Python ignores everything after the hash mark and up to the end of the line. You can insert them anywhere in your code, even inline with other code:

```
print("This will run.")  # This won't run
```

When you run the above code, you will only see the output This will run. Everything else is ignored.

Comments should be short, sweet, and to the point. While PEP 8 advises keeping code at 79 characters or fewer per line, it suggests a max of 72 characters for inline comments and docstrings. If our comment is approaching or exceeding that length, then we'll want to spread it out over multiple lines.