Python Assignment 1

Q1. Which keyword is used to create a function? Create a function to return a list of odd numbers in the range of 1 to 25.

Ans.

```
# Create a function to return a list of odd numbers in the range of 1 to 25.
def listOfOdd():
  """a function to return a list of odd numbers in the range of 1 to 25."""
  l1=[]
  for i in range(1,25):
    if i%2 != 0:
      l1.append(i)
  return l1
listOfOdd()
```

Q2.. Why *args and **kwargs is used in some functions? Create a function each for *args and **kwargs to demonstrate their use.

Ans. We use *args and **kwargs as an argument when we are unsure about the number of arguments to pass in the functions.

```
#a function for *args to demonstrate its use.
def demo1(*args):
  """a function for *args to demonstrate its use."""
  index=0
  for i in args:
    print("The args["+str(index)+"] having:",i)
    index+=1
demo1("Hi", "There", "This is test", 1)

#a function for kwargs to demonstrate its use.
def demo2(**kwargs):
  """a function for kwargs to demonstrate its use."""
  index=0
  for key, value in kwargs.items():
    print("The kwargs["+str(index)+"] having key '"+str(key)+ "' and value '"+str(value)+"'")
    index+=1
demo2(FirstName= "Vishvesh", LastName= "Jain", Location= "Delhi", CourseEnrolled= "Data Science Master")
```

Q3. What is an iterator in python? Name the method used to initialise the iterator object and the method used for iteration. Use these methods to print the first five elements of the given list [2, 4, 6, 8, 10, 12, 14, 16, 18, 20].

Ans. Iterators are methods that iterate collections like lists, tuples, etc. Using an iterator method, we can loop through an object and return its elements. The for loop in Python is used to iterate over a sequence of elements, such as a list, tuple, or

string. When we use the for loop with an iterator, the loop will automatically iterate over the elements of the iterator until it is exhausted.

```
#Use iterator methods to print the first five elements of the given list [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]

l=[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]

count=0

for i in l:

    if count<5:

        print(i)

        count+=1
```

Q4. **What is a generator function in python? Why yield keyword is used? Give an example of a generator function.**

Ans: A generator function in Python is a special type of function that returns a generator iterator, which can be used to iterate over a sequence of elements. A generator function is defined like a normal function, but instead of using the return statement to return a value, it uses the yield statement. The yield statement is used to produce a value, and the generator function can be resumed from where it left off the next time next() is called on the generator.

Here's an example of a generator function that generates the Fibonacci sequence:

```
#the fibonacci series program using generator yield

def fibonacci():

  """This function will generate fibonacci sequence"""

  a, b = 0, 1

  while True:
```

```
        yield a
        a, b = b, a + b

fib = fibonacci()
for i in range(10):
    print(next(fib))
```

Q5. Create a generator function for prime numbers less than 1000. Use the next() method to print the first 20 prime numbers.

```
#a generator function for prime numbers less than 1000. Use the next() method to print the first 20 prime numbers.
def primes():
    """Create a generator function for prime numbers less than 1000."""
    yield 2
    primes_list = [2]
    for i in range(3, 1000):
        is_prime = True
        for prime in primes_list:
            if i % prime == 0:
                is_prime = False
```

```
            break
    if is_prime:
        primes_list.append(i)
        yield i

prime_gen = primes()
# the next() method to print the first 20 prime numbers.
for i in range(20):
  print(next(prime_gen))
```

Q6. Write a python program to print the first 10 Fibonacci numbers using a while loop.

```
#a python program to print the first 10 Fibonacci numbers using a while loop.
def fibonacci(n):
  """This function will print fibonacci sequence"""
  a, b = 0, 1
  count = 0
  while count < n:
    print(a)
    a, b = b, a + b
```

```
    count += 1
```

```
fibonacci(10)
```

Q7. Write a List Comprehension to iterate through the given string: ‘pwskills’. Expected output: ['p', 'w', 's', 'k', 'i', 'l', 'l', 's']

Ans:

```
#a List Comprehension to iterate through the given string: ‘pwskills’.

string = 'pwskills'

output = [char for char in string]

print(output)
```

Q8. Write a python program to check whether a given number is Palindrome or not using a while loop.

```
#a python program to check whether a given number is Palindrome or not using a while loop.

n = int(input("Please give a number : "))

def reverse(num):

  if num<10:
```

```
    return num
  else:
    return int(str(num%10) + str(reverse(num//10)))
def isPalindrome(num):
  if num == reverse(num):
     return 1
  return 0
if isPalindrome(n) == 1:
  print("Given number is a palindrome")
else:
  print("Given number is a not palindrome")
```

Q9. Write a code to print odd numbers from 1 to 100 using list comprehension.

```
#a code to print odd numbers from 1 to 100 using list comprehension.
[odd for odd in range(1,100) if odd%2!=0]
```