PYTHON Assignment Questions

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.
Answer: “Def” keyword is used to define a function.

Code:
```
def odd():
    n=25
    nums=[]
    for i in range (1,n+1):
        if i%2!=0:
            nums.append(i)
    print("List of Odd Numbers are:", nums)
odd()
```

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

Answer: *args and **kwargs are used when you are unsure about the arguments and want to pass an unspecified number of arguments.

**kwargs is used when arguments are passed in the form of key and value pair.
Code 1:
```
def sum(*args):

    a=0
    for i in args:
        a=a+int(i)
    return print("sum of given numbers", args, "is", a)

sum(1,2,3,6,5,)
```

Code2:
```
def mlist(**kwargs):
    for i in kwargs.keys():
        print("Monument in",i, "is",kwargs[i])

mlist(Delhi="Red Fort", Jaipur="Hawa mahal", Agra="Taj mahal")
```

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].

Answer: Iterators are the objects that are used to get the next value of the sequence.

iter() is the method used to initialise the iterator object and next() is used to iterate over iterable object

Code:
```
list = [2, 4, 6, 8, 10, 12, 14, 16,  18, 20]
elements= iter(list)

print(next(elements))
print(next(elements))
print(next(elements))
print(next(elements))
print(next(elements))
```

Q4. What is a generator function in python? Why yield keyword is used? Give an example of a generator function.
Answer: A Python generator function allows you to declare a function that behaves like an iterator.
Yield is used when we want to iterate over a sequence, but don't want to store the entire sequence in memory.

Code:
```
fruits = ("apple", "banana", "cherry")
myit = iter(fruits)

print(next(myit))
print(next(myit))
print(next(myit))
```

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

Answer:

Code:
```
def prime_generator(n):
    for i in range(1, n):
        for j in range(2, int(i ** 0.5) + 1):
            if i % j == 0:
                break
        else:
            yield i

for i, p in enumerate(prime_generator(1000)):
    if i == 37:
        break
    print(p,",", end=' ')
```

Note: Create your assignment in Jupyter notebook and upload it to GitHub & share that GitHub repository link through your dashboard.