03feb_assignment

February 13, 2023

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.

In Python, the 'def' keyword is used to create a function.

[1]:
```
def odd_numbers():
    odd_list = []
    for i in range(1,26):
        if i % 2 != 0:
            odd_list.append(i)
    return odd_list
```

[2]:
```
odd_numbers()
```

[2]:
```
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25]
```

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

Answer: *args is used to send a non-keyworded variable length argument list to the function. it allows the function to accept an arbitrary number of non-keyworded arguments. The arguments passed to the function using* args are stored in a tuple.

Answer: **kwargs is used to send a keyworded, variable length argument list to the function. it allows the function to accept an arbitrary number of keyworded arguments. The arguments passed to the function using** kwargs are stored in a dictionary.

[4]:
```
def test1(*args):
    return(args)
```

[5]:
```
test1(1,2,3,4)
```

[5]:
```
(1, 2, 3, 4)
```

[7]:
```
def test2(**kwargs):
    return(kwargs)
```

[11]:
```
test2(a=4, b=6, c=[4,5,8], d=45.63)
```

[11]:
```
{'a': 4, 'b': 6, 'c': [4, 5, 8], 'd': 45.63}
```

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: An iterator in Python is an object that can be iterated (looped) upon. It has a method called "next()" that is used to access the next element in the iteration.

Answer: In Python, an iterator can be created from an iterable object, such as a list or a string, using the iter() method. The next() method is used to retrieve the next item in the iteration.

[12]:
```
test3=[2,4,6,8,10,12,14,16,18,20]
s = iter(test3)
```

[13]:
```
type(s)
```

[13]:
```
list_iterator
```

[14]:
```
for i in range(5):
    print(next(s))
```

```
2
4
6
8
10
```

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

Answer: A generator function in Python is a special kind of function that returns a generator iterator. It allows you to generate a sequence of values over time, instead of generating them all at once and storing them in memory. This makes generator functions an efficient and memory-friendly alternative to traditional functions that return lists or other data structures.

Answer: The yield keyword is used in generator functions to produce a value and pause the function's execution. Each time the function is called, it resumes execution from the point where it was paused, until it encounters the yield keyword again. This allows you to iterate over the values produced by the generator function one by one, instead of having to compute all of them at once.

[24]:
```
def test4():
    a,b=0,1
    for i in range(10):
        yield a
        a,b = b, b+a
```

[25]:
```
test4()
```

[25]:
```
<generator object test4 at 0x7f09a0dc8200>
```

[23]:
```
list(test4())
```

[23]:
```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

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

[6]:
```
def prime_number():
    primes =[]
    for i in range(2,1000):
        for prime in primes:
            if i % prime ==0:
                break
        else:
                primes.append(i)
                yield i
```

[7]:
```
gen = prime_number()
for num in range(20):
    print(next(gen))
```

```
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
71
```

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

[11]:
```
def fibbo():
    a,b=0,1
    while True:
        yield a
        a,b = b, a+b
```

[12]:
```
fib= fibbo()
```

[14]:
```
for i in range(10):
    print(next(fib))
```

```
0
1
1
2
3
5
8
13
21
34
```

Q7. Write a List Comprehension to iterate through the given string: 'pwskills'.

[16]:
```
string= "pwskills"
i=[]
for letter in string:
    i.append(letter)
print(i)
```

```
['p', 'w', 's', 'k', 'i', 'l', 'l', 's']
```

[17]:
```
string1= "pwskilla"
result= [char for char in string1]
print(result)
```

```
['p', 'w', 's', 'k', 'i', 'l', 'l', 'a']
```

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

[5]:
```
num=(int(input("Enter your number")))
def palindrome(num):
    i = num
    rev = 0
    while(i>0):
        rev= (rev*10) + (i % 10)
        i = i // 10
    return rev == num
if palindrome(num):
    print(f"{num} is a palindrome.")
else:
    print(f"{num} is not a palindrome.")
```

```
Enter your number 753159951357
```

```
753159951357 is a palindrome.
```

Q9. Write a code to print odd numbers from 1 to 100 using list comprehension. Note: Use a list comprehension to create a list from 1 to 100 and use another List comprehension to filter out odd numbers.

[6]:
```
odd_numbers = [x for x in range(1, 101) if x % 2 != 0]
print(odd_numbers)
```

```
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41,
43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65, 67, 69, 71, 73, 75, 77, 79, 81,
83, 85, 87, 89, 91, 93, 95, 97, 99]
```

[ ]:
```

```