	K. J. Somaiya College of Engineering, Mumbai-77 **(A Constituent College of Somaiya Vidyavihar University)** **Department of Science and Humanities**	

Name: Shreyans Tatiya
Batch: C5_3 Roll No.: 16010123325
Experiment / assignment / tutorial No.
Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Title : User defined functions in Python

AIM: To implement User-defined functions in Python

Expected OUTCOME of Experiment:

CO2: Use different Decision making statements and Functions in Python.

Resource Needed: Python IDE

Theory:

1. Python Functions

A function is a block of code which only runs when it is called. You can pass data, known as parameters, into a function. A function can return data as a result.

Creating a Function:

Python a function is defined using the def keyword:
Example:

```
def my_function():
    print("Hello from a function")
```

Arguments:

Information can be passed into functions as arguments. Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma.

Parameters or Arguments:

The terms parameter and argument can be used for the same thing: information that is passed into a function. From a function's perspective: A parameter is the variable listed inside the parentheses in the function definition. An argument is the value that is sent to the function when it is called.

Number of Arguments:

By default, a function must be called with the correct number of arguments i.e. if your function expects 2 arguments; you have to call the function with 2 arguments, not more, and not less.

Keyword Arguments

You can also send arguments with the key = value syntax.
This way the order of the arguments does not matter.

Arbitrary Keyword Arguments, **kwargs

If you do not know how many keyword arguments will be passed into your function, add two asterisk: ** before the parameter name in the function definition.
This way the function will receive a dictionary of arguments, and can access the items accordingly

Default Parameter Value

The following example shows how to use a default parameter value.
If we call the function without argument, it uses the default value:

Passing a List as an Argument

You can send any data types of argument to a function (string, number, list, dictionary etc.), and it will be treated as the same data type inside the function.

Return Values

To let a function return a value, use the return statement:

The pass Statement

Function definitions cannot be empty, but if you for some reason have a function definition with no content, put in the pass statement to avoid getting an error.

2. Recursion Function

Python also accepts function recursion, which means a defined function can call itself. Recursion is a common mathematical and programming concept. It means that a function calls itself. This has the benefit of meaning that you can loop through data to reach a result. The developer should be very careful with recursion as it can be quite easy to slip into writing a function which never terminates, or one that uses excess amounts of memory or processor power. However, when written correctly recursion can be a very efficient and mathematically-elegant approach to programming.
To a new programmer it can take some time to work out how exactly this works, best way to find out is by testing and modifying it.

3. Lambda function

- A lambda function is a small anonymous function.
- A lambda function can take any number of arguments, but can only have one expression. Syntax of Lambda Function is given below

lambda arguments *:* expression
Lambda functions can take any number of arguments:

Problem Definition:

1. In below table input variable, python code and output column is given. You have to complete blank cell in every row.

Python Code	Output

def my_function(fname,lname):
  print(fname+ " " + lname)
 	
my_function("Amit", "Kumar")
	Amit Kumar

def my_function(fname, lname):
  print(fname + " " + lname)
 	
my_function("Emil")
	TypeError: my function() missing 1 required Positional argument: 'lname'

def my_function(*kids):
  print("The youngest child is " + kids[2])
 	
my_function("Emil", "Tobias", "Linus")
	The youngest child is Linus

def my_function(college3, college2, college1):
  print("The Best college is " + college3)
 	
my_function(**?**)
	SyntaxError: invalid syntax

def my_function(**country= "Norway"**):
  print("I am from " + country)
 	
my_function("Sweden")
my_function("India")
my_function()
my_function("Brazil")
	I am from Sweden
	I am from India
	I am from Norway
	I am from Brazil

def tri_recursion(k):
    if(k > 0):
  result = k + tri_recursion(k - 1)
  print(result)
 	
   else:
  result = 0
 	
   return result
	1
	3
	6
	10
	15
	21

print("Recursion Example Results")
tri_recursion(6)	

print((lambda x: x*2) (9))	18

twice = lambda x: x*2
print(twice(9))
	18

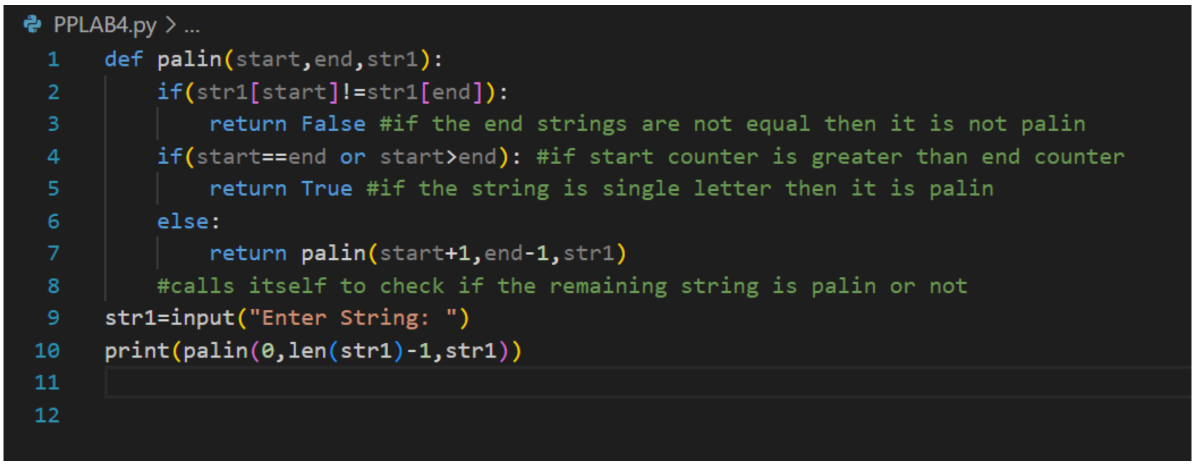
2. Write a Python program using a recursive function that takes a string as input from the user and displays whether the string is Palindrome or not.
3. Write a Python program to separate out even and odd numbers from the list entered by user by using Lambda function

Books/ Journals/ Websites referred:

1. Reema Thareja, *Python Programming: Using Problem Solving Approach*, Oxford University Press, First Edition 2017, India
2. Sheetal Taneja and Naveen Kumar, *Python Programming: A modular Approach*, Pearson India, Second Edition 2018,India

Implementation details:

2.

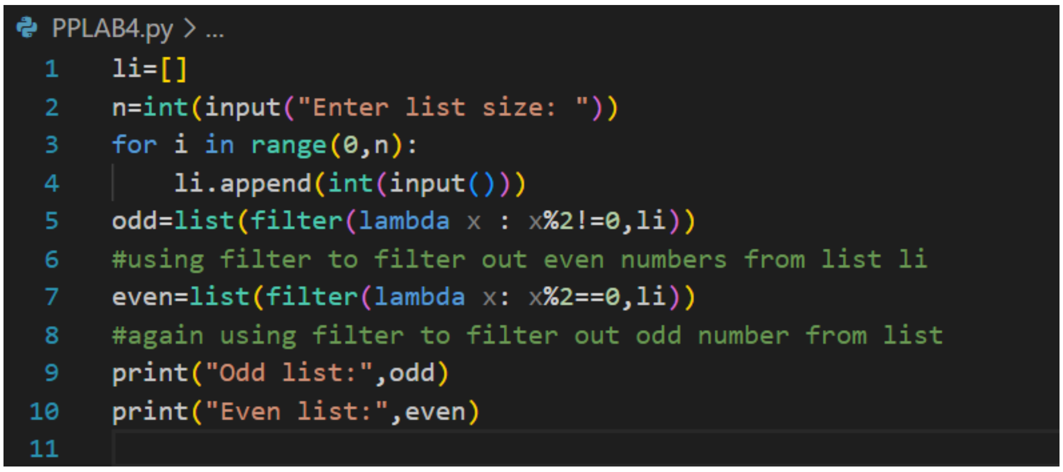


PPLAB4.py > ...

```
def palin(start,end,str1):
    if(str1[start]!=str1[end]):
        return False #if the end strings are not equal then it is not palin
    if(start==end or start>end): #if start counter is greater than end counter
        return True #if the string is single letter then it is palin
    else:
        return palin(start+1,end-1,str1)
    #calls itself to check if the remaining string is palin or not
str1=input("Enter String: ")
print(palin(0,len(str1)-1,str1))


```

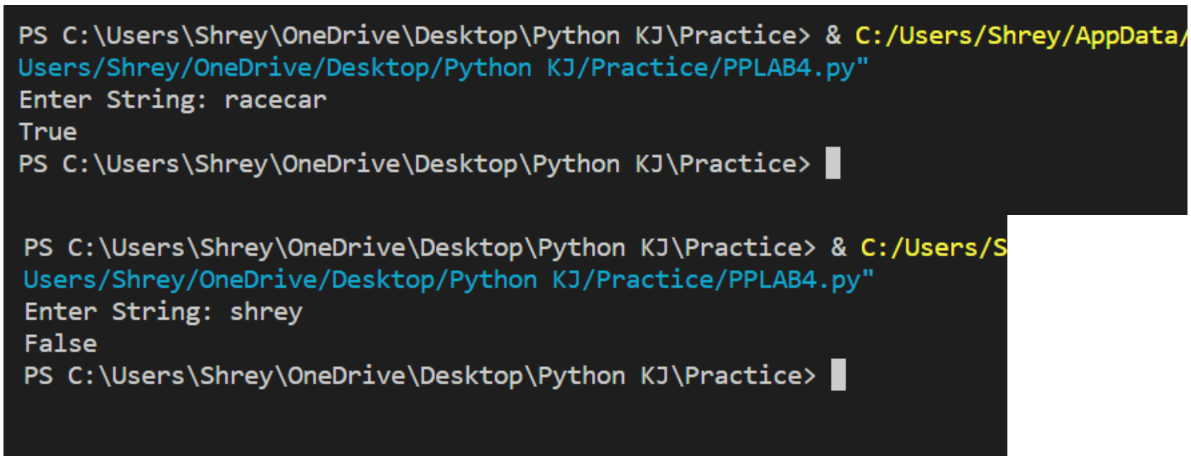
3.



```
PPLAB4.py > ...
li=[]
n=int(input("Enter list size: "))
for i in range(0,n):
    li.append(int(input()))
odd=list(filter(lambda x : x%2!=0,li))
#using filter to filter out even numbers from list li
even=list(filter(lambda x: x%2==0,li))
#again using filter to filter out odd number from list
print("Odd list:",odd)
print("Even list:",even)
```

Output(s):

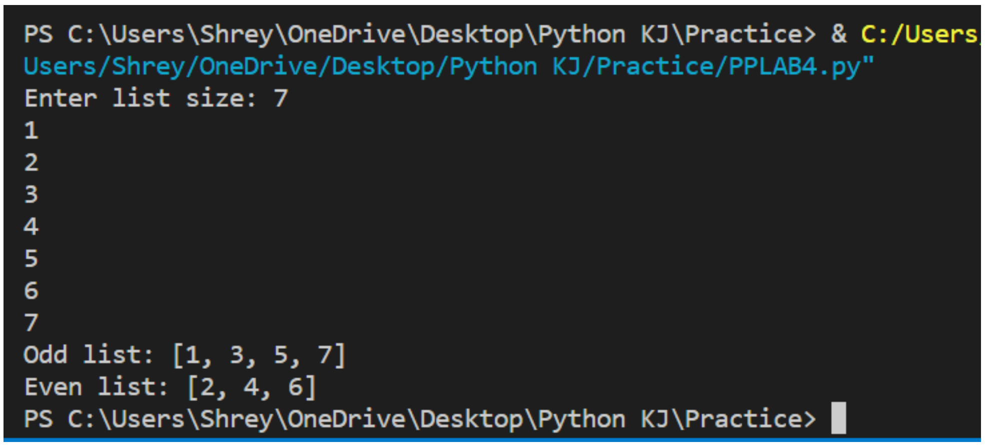
2.



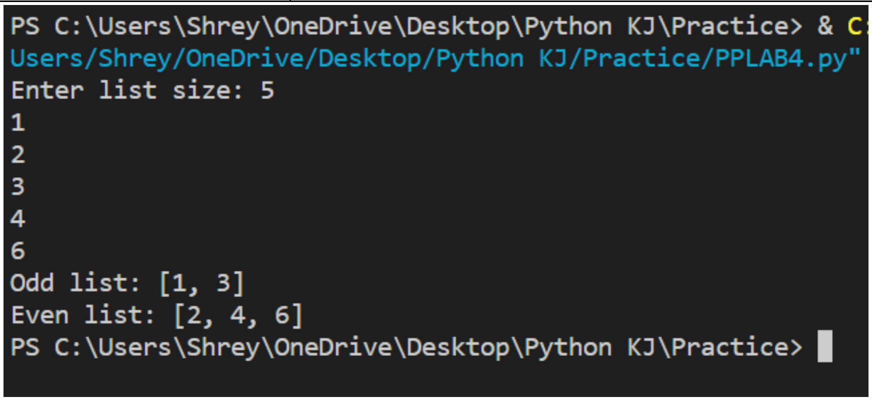
```
PS C:\Users\Shrey\OneDrive\Desktop\Python KJ\Practice> & C:/Users/Shrey/AppData/
Users/Shrey/OneDrive/Desktop/Python KJ/Practice/PPLAB4.py"
Enter String: racecar
True
PS C:\Users\Shrey\OneDrive\Desktop\Python KJ\Practice>
```

```
PS C:\Users\Shrey\OneDrive\Desktop\Python KJ\Practice> & C:/Users/S
Users/Shrey/OneDrive/Desktop/Python KJ/Practice/PPLAB4.py"
Enter String: shrey
False
PS C:\Users\Shrey\OneDrive\Desktop\Python KJ\Practice>
```

3.



```
PS C:\Users\Shrey\OneDrive\Desktop\Python KJ\Practice> & C:/Users
Users/Shrey/OneDrive/Desktop/Python KJ/Practice/PPLAB4.py"
Enter list size: 7
1
2
3
4
5
6
7
Odd list: [1, 3, 5, 7]
Even list: [2, 4, 6]
PS C:\Users\Shrey\OneDrive\Desktop\Python KJ\Practice>
```

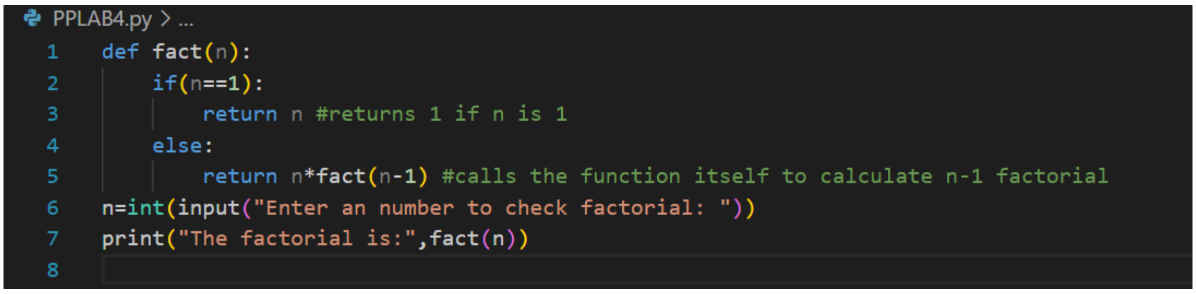


```
PS C:\Users\Shrey\OneDrive\Desktop\Python KJ\Practice> & C:
Users/Shrey/OneDrive/Desktop/Python KJ/Practice/PPLAB4.py"
Enter list size: 5
1
2
3
4
6
Odd list: [1, 3]
Even list: [2, 4, 6]
PS C:\Users\Shrey\OneDrive\Desktop\Python KJ\Practice>
```

Conclusion:
Hence, we understood the concept and use of recursions and lambda in Python and programs were executed successfully.

Post Lab Descriptive Questions

1. Write a python program to calculate factorial using recursion.
   Ans:



```python
def fact(n):
    if(n==1):
        return n #returns 1 if n is 1
    else:
        return n*fact(n-1) #calls the function itself to calculate n-1 factorial
n=int(input("Enter an number to check factorial: "))
print("The factorial is:",fact(n))
```

2. What are the common functional programming methods that use lambdas?
   Ans:
   Lambda functions are anonymous functions which take in only one particular arguments. They allow us to create and use a function in a single line which are useful when we need a short function that will be used only once. They are mostly
   used in conjunction with the map, filter, short methods and in cases where the programmer needs to perform a small task repetitively.

Date: ______________ **Signature of faculty in-charge**