**Batch: B2 Roll No.: 121**

**Experiment / assignment / tutorial No.**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| **TITLE:**  Recursion and Lambda Function |

**AIM:** To implement recursion function and lambda function

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Expected OUTCOME of Experiment:**

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

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Resource Needed: Python IDE**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Theory:**

## 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

## In 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.

## Recursion

## 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.

|  |  |  |
| --- | --- | --- |
| S.No | Python Code | Output |
| 1. | def my\_function(fname, lname):   print(fname + " " + lname)  my\_function("Amit", "Kumar") | Amit Kumar |
| 2. | def my\_function(fname, lname):   print(fname + " " + lname)  my\_function("Emil") | TypeError: my\_function() missing 1 required positional argument: ‘lname’ |
| 3. | def my\_function(\*kids):  print("The youngest child is " + kids[2])  my\_function("Emil", "Tobias", "Linus") | The youngest child is Linus |
| 4. | def my\_function(college3, college2, college1):   print("The Best college is " + college3)  my\_function(**college3 = “KJSCE”, college2 = “”, college1 = “”**) | The Best college is KJSCE |
| 5. | 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 |
| 6. | def tri\_recursion(k):  if(k > 0):  result = k + tri\_recursion(k - 1)  print(result)  else:  result = 0  return result  print("Recursion Example Results")  tri\_recursion(6) | 1  3  6  10  15  21 |
| 7. | print((lambda x: x\*2) (9)) | 18 |
| 8. | twice = lambda x: x\*2  print(twice(9)) | 18 |

1. 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.
2. 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:**

**Code for Question 2:**

numlist = eval(input("Ente a list of integers: "))

even = list(filter(lambda x: x%2==0, numlist))

print(even)

odd = list(filter(lambda y: y%2!=0, numlist))

print(odd)

**Code for Question 3:**

word = input("Enter a string: ")

#end=len(word)-1

def palindrome(start, end, word):

if(word[start]==word[end] and start<end):

palindrome(start+1, end-1, word)

else:

if(start>=end):

print(word, " is a Palindrome.")

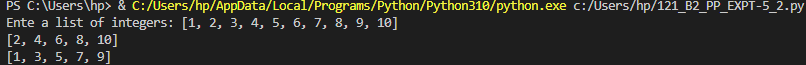
elif(start<end):

print(word, " is not a Palindrome.")

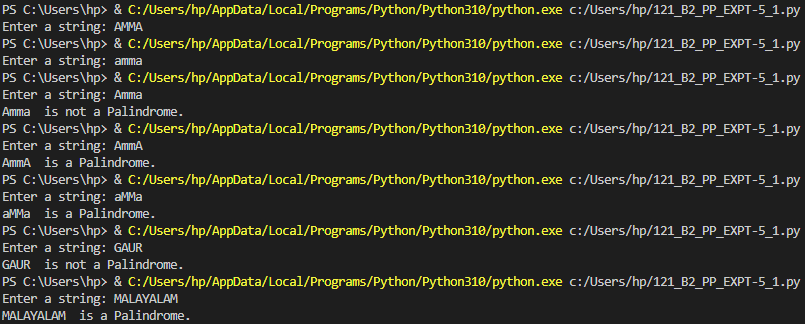
palindrome(0, len(word)-1, word)

**Output(s):**

Output for Question 2:



Output for Question 3:



**Conclusion:**

**Thus, in this experiment, the application of recursive functions and lambda function was shown. Recursive functions can make programming efficient; make the code look elegant and increase the processing speed. However, they have to be used carefully as a little inattention can lead to an infinite loop. Lambda functions are anonymous functions, i.e., they are unnamed. They are completely defined in a single line. When used decisively, they help to reduce the size of code and number of lines. Further, they can be used to perform specific tasks within a function. When combined with a filter function, lambda functions can practically be used to iterate through a list and to get list items which match a specific condition. Thus, these functions make Python Programming up-to-date and efficient.**

**Post Lab Descriptive Questions**

1. Write a python program to calculate factorial using recursion

Ans.

num = int(input("Enter a number to find its factorial: "))

def fact(num):

if num==1:

return num

else:

num = num - 1

return num\*fact(num)

print("The factorial of ", num, " is ", num\*fact(num))



1. What are the common functional programming methods that use lambdas?

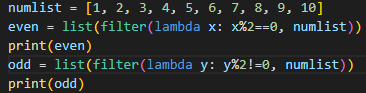
Ans. The common functional programming methods that use lambda functions are:

* + 1. To compare two quantities:

max = lambda a,b: a if a>b else b

print(max(45000, 48190))

* + 1. To get a list of numbers from a list which satisfies a given condition:





* + 1. They are used in conjunction with filter (as seen above), map and sort methods.

**Date: \_13-06-2022\_ Signature of faculty in-charge**