**Batch: A2 Roll No.: 16010121045**

**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: myfunction() missing 1requiredPositionalargument: ‘lname’ |
| 3. | def my\_function(\*kids): print("The youngest child is " + kids[2])my\_function("Emil", "Tobias", "Linus") | The youngest childis Linus |
| 4. | def my\_function(college3, college2, college1):  print("The Best college is " + college3)my\_function(**KJSCE,IITB,IITH**) | 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 SwedenI am from IndiaI am from NorwayI am from Brazil |
| 6. | def tri\_recursion(k): if(k > 0): result = k + tri\_recursion(k - 1) print(result) else: result = 0 return resultprint("Recursion Example Results")tri\_recursion(6) | 136101521 |
| 7. | print((lambda x: x\*2) (9)) | 18 |
| 8. | twice = lambda x: x\*2print(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:**

**2.**

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.**

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.**

 

**3.**

 

**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

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))

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

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**