**CMP4272- DATA STRUCTURES AND ALGORITHMS**

**LAB SESSION-6: Recursion**

**Objectives:**

* Identify computational problems which can be solved recursively.
* Use structural recursion to solve simple computational problems on trees and linked lists.
* Understand how recursive functions are computed using a call stack.

1. **Background:**

Recursion occurs when a thing is defined in terms of itself or of its type. Recursive systems must exhibit the following behaviour:

* A simple *base case* (or cases) resulting in a terminating scenario that does not use recursion to produce an answer
* A *recursive step;* a set of rules that reduces all other cases toward the base case

Without a base case a recursive function will continue forever.

1. **Complete the following exercises before you attempt the mandatory ones provided under Lab submission Exercises in the next section.**

**Exercise-1**:

i) Write a Python function that calculates the Factorial of a given number recursively.

*Answer:*

ii) Show call stack for Factorial(6).

*Answer:*

**Exercise-2:** Consider the following Python code:

The Node class for a tree ADT is defined in list-of-children representation as:

class Node:

    def \_\_init\_\_(self, value, children=None):

        # a leaf node has an empty list of children

        if children is None:

            children = []

        self.value = value

        self.children = children

1. **Write a function that calculates sum of all node values of a tree recursively.**

*Answer:*

1. A tree structure is constructed comprising of various nodes given as:

root = Node(1, [

    Node(2, [

        Node(3, [

            Node(4),

            Node(5)

        ]),

        Node(6)

    ]),

    Node(7, [

        Node(8, [

            Node(9)

        ])

    ]),

    Node(10)

])

**Using the recursive function defined in i), calculate and print the sum of all nodes for the given tree referred to by root.**

*Answer:*

1. **Lab Submission Exercises:**

* **Submitting the solution of the following exercises (3 and 4) is mandatory.**
* **Solutions that comprise of python code, must be well documented. (Include necessary comments)**

**Exercise-3**: Consider the following Python code:

The Node class for a tree ADT is defined in list-of-children representation as:

class Node:

    def \_\_init\_\_(self, value, children=None):

        # a leaf node has an empty list of children

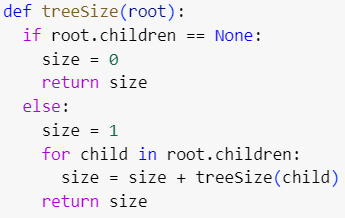
        if children is None:

            children = []

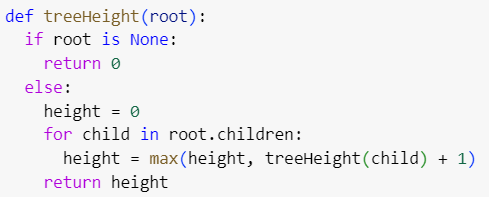
        self.value = value

        self.children = children

1. **Write a function that calculates number of nodes (size) of a tree recursively.**

****

1. **Write a function that calculates height of a tree recursively.**

**

1. A tree structure is constructed comprising of various nodes given as:

root = Node(1, [

    Node(2, [

        Node(3, [

            Node(4),

            Node(5)

        ]),

        Node(6)

    ]),

    Node(7, [

        Node(8, [

            Node(9)

        ])

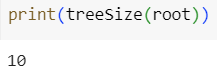
    ]),

    Node(10)

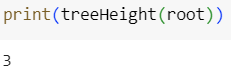
])

**Using the recursive function defined in i) and ii), to calculate and print:**

* + **Size of the tree**



* + **Height of the tree**



1. **Moodle Submission:**

You are required to submit your solution in the word document.

Naming Format: **StudetName\_studentID.docx [ or other word formats]**

Example : AliceSmith\_514099.docx

**NOTE**

* It is important to complete the weekly labs in particular labs 2, 3, 4, 5 and 6 because it contains questions that are part of the coursework. (Weightage: 25%).
* Only one of these labs will be chosen randomly for marking, so it is important that you complete and submit each of these labs.
* Ideally, you should submit each lab within one week of that lab session.
* Solutions that comprise of python code, must be well documented. (Include necessary comments)