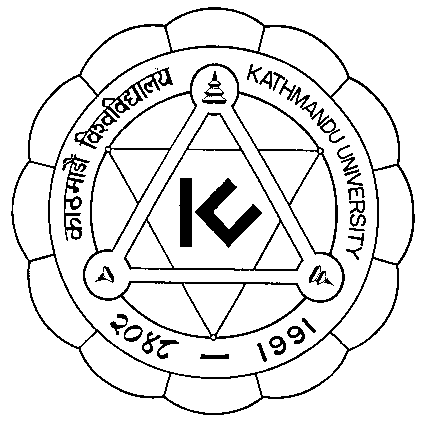
**KATHMANDU UNIVERSITY**

**Department of Computer Science & Engineering**

**Dhulikhel, Kavre.**



**A Mini-project report On**

**Binary tree traversal simulation**

**COMP 342**

**Submitted by:**

**Ejan Shakya**

**Roll no. 45**

**CE 3rd year/2nd sem**

**Submitted to**

**Mr. Dhiraj Shrestha**

**Department of Computer Science and Engineering**

**Submission Date: 27/07/2018**

Contents

[Introduction 3](#_Toc520449277)

[Objectives 4](#_Toc520449278)

[Features 4](#_Toc520449279)

[Design and Methods 4](#_Toc520449280)

[TOOLS USED 5](#_Toc520449281)

[LANGUAGE USED 5](#_Toc520449282)

[CONCLUSION 6](#_Toc520449283)

[Screenshot 6](#_Toc520449284)

[References 7](#_Toc520449285)

# Introduction

Binary Tree is a special data structure used for data storage purposes. A binary tree has a special condition that each node can have a maximum of two children. A binary tree has the benefits of both an ordered array and a linked list as search is as quick as in sorted array and insertion or deletion operation are as fast as in linked list.



Binary Search tree exhibits a special behavior. A node’s left child must have a value less than its parents’ value and the node’s right child must have a value greater than its parent value.



# Objectives

The main objectives of this project are:

* To simulate how traversal in binary tree works
* To depict the features of a binary tree

# Features

1. Randomly generate a binary tree every time the user hits the ‘R’ key
2. Traverse the tree using the keys W,S,A,D with exception handling
3. Calculate the depth of the current node where the cursor is pointed at

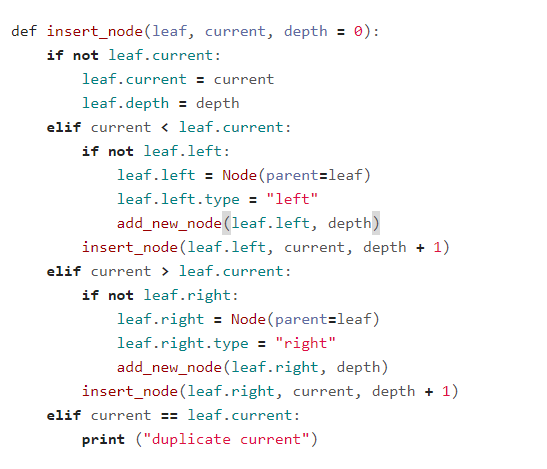
# Design and Methods

The source code has two classes: master and node class.

1. Master class has an object “self” initialized for a nodelist which I s a 2-d array of nodes sorted by depth level. It is initialized as empty and the layers are added as needed.
2. Node class depicts the features of a binary tree. For each node created in master class, node class has its attributes initialized. Attributes include:

* Parent as self.parent
* Left child as self.left
* Right child as self.right
* Current node as self.current
* Depth of the node as self.depth

Using these classes, objects and their attributes, we build a random binary tree which has the aforementioned features. To insert the node in the tree we use the following code:



In this piece of code, we insert a node that is randomly generated in the function build\_tree() is used. It checks whether the node is a left child or right child.

* If current node < leaf node, then left child is inserted
* If current node > leaf node, then right child is inserted
* If current node = leaf node, then duplicate is printed.

## TOOLS USED

Atom IDE:

Atom is a free and open-source text and source code editor for macOS, Linux, and Microsoft Windows with support for plug-ins written in Node.js, and embedded Git Control, developed by GitHub.

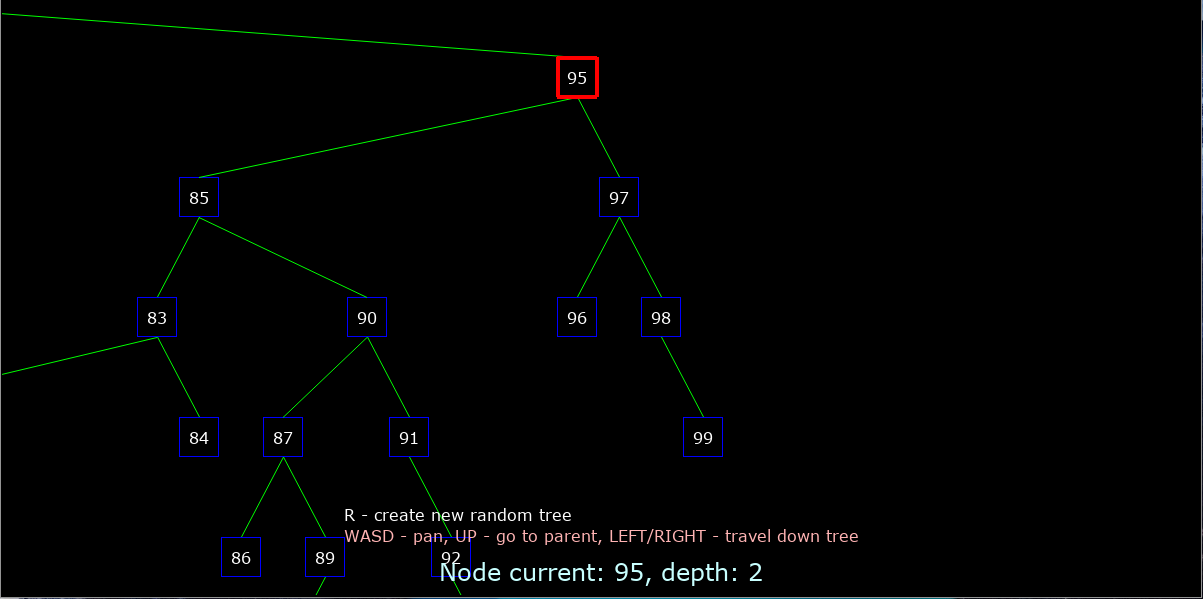
## LANGUAGE USED

Python:  
Python is an interpreted high-level programming language for general-purpose  
programming. Python features a dynamic type system and automatic memory  
management. It supports multiple programming paradigms, including object-oriented,  
imperative, functional and procedural, and has a large and comprehensive standard library.  
  
Pygame:  
Pygame is a cross-platform set of Python modules designed for writing video games. It  
includes computer graphics and sound libraries designed to be used with the Python  
programming language

# Conclusion

The main aim of this mini-project is to model and simulate various properties of a binary tree like, binary search, insertion, traversal, etc. using Python and Pygame graphics library. With this mini-project, I hope to exhibit the practical implementation of the knowledge I’ve gained in the respective field.

## Screenshot



# References

* En.wikipedia.org. (2018). Binary-tree algorithm. [online] Available at: [https://en.wikipedia.org/wiki/Sorting\_algorithm#Insertion\_sort](https://l.facebook.com/l.php?u=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FSorting_algorithm%23Insertion_sort&h=AT2Na6jMOeYiH1x6OgF4Vtwnhp-eP8GFFsBX6LG8xPIrKZyLJL6RPyt_zaVUqCpLkRLEAfDDPOkvXofzl_GKSVP7kE74EUX6jNFSSPJWCQ8fmwYzcCQsYNE9f4_HRG0x1Gt3dw) [Accessed 31 May 2018].
* En.wikipedia.org. (2018). Tkinter. [online] Available at: [https://en.wikipedia.org/wiki/Tkinter](https://l.facebook.com/l.php?u=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FTkinter&h=AT2Na6jMOeYiH1x6OgF4Vtwnhp-eP8GFFsBX6LG8xPIrKZyLJL6RPyt_zaVUqCpLkRLEAfDDPOkvXofzl_GKSVP7kE74EUX6jNFSSPJWCQ8fmwYzcCQsYNE9f4_HRG0x1Gt3dw) [Accessed 31 May 2018].
* En.wikipedia.org. (2018). Python (programming language). [online] Available at: <https://en.wikipedia.org/wiki/Python_(programming_language)> [Accessed 31 May 2018].