Programming Assignment
CS 321
2017

March 29, 2017

Write the following programs in Python. We may use scripts to organize/evaluate/check plagiarism. Please follow the instructions very carefully to avoid any error.

- Send your mail to Shivdutt Sharma `shiv.sharma@iitgn.ac.in` cc to `bireswar@iitgn.ac.in`
- Your email subject must be "CS321 Programming Assignment" (You may copy/paste this from here).
- Names that you should use for each program file is given along with the problem. For example the first program must be named BinaryTreeCrawler.py.
- Keep all your programs in as single folder. If your name is Rohit and roll is 1320014, name the folder Rohit1320014. Compress (zip/tar.gz) the folder and attache the compressed (zipped) file with your mail.

Deadline 10 April 2017.

1. **Binary Tree Crawler** In this program the computer takes input from the user before performing any operation. The operations that need to be supported are 1. Move 2. Add 3. Print. The program shows a menu

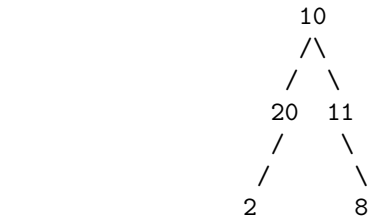
   ```
   1. Move
   2. Add
   3. Print
   4. Exit
   ```

   If the user picks 1, then the program asks for a movement pattern. For example if the user gives "LLRRL" the program starts from the current node and moves left, again to the left followed by two right movements and finally a left movement. If the movement is not possible then it reports

that and stays in the current node otherwise the current node is updated. Further movement starts from the new node. A movement pattern is a string with letters L, R, P, r where L=left, R=Right, P=Parent, r=root.

In operation 2 the program tries to add a new node to the left or right of the current node. If the user picks 2, the program asks "Left or Right?". Suppose the user picks "L" then if the left child of the current node is "Nil" it asks for a value and adds a node with the value as the left child of the current node. If the left child is not nil, it says "Cannot Add, Place not empty".

The print operation prints the binary tree. The output should match the picture of the tree. For example the text based console output may look like



(Use the Node definition without any change from the code given at the end. If you were in ES102 you may need start working on this problem as soon as possible.) (filename:BinaryTreeCrawler.py) 10.

2. Implement Huffman code. The letters and the frequencies will be given as input. (Extra points if you can use a priority queue that stores tree nodes). (Huffman.py) 10.

3. Write a program that takes two strings as inputs and prints a longest common subsequence of the two strings. (filename:LCS.py) 10.

4. The input for this algorithm will be the adjacency matrix for an undirected graph. The algorithm should check if the graph is a tree. (filename:Tree.py) 10.

```
#-----------Node Definitation (Do not change)-------------

class Node():
    def __init__(self,value):
        self.val=value
        self.left=None
        self.right=None
        self.parent=None
```