National Institute of Technology, Calicut

Department of Computer Science and Engineering

CS2094D – Data Structures Lab

Assignment-4 Advanced batch

Policies for Submission and Evaluation

You must submit your assignment in the moodle (Eduserver) course page, on or before the submission deadline. Also, ensure that your programs in the assignment must compile and execute without errors in Athena server. During evaluation your uploaded programs will be checked in Athena server only. Failure to execute programs in the assignment without compilation errors may lead to zero marks for that program.

Your submission will also be tested for plagiarism, by automated tools. In case your code fails to pass the test, you will be straightaway awarded zero marks for this assignment and considered by the examiner for awarding F grade in the course. Detection of ANY malpractice regarding the lab course will also lead to awarding an F grade.

Naming Conventions for Submission

Submit a single ZIP (.zip) file (do not submit in any other archived formats like .rar or .tar.gz). The name of this file must be ASSG<NUMBER>_<ROLLNO>_<FIRSTNAME>.zip (For example: ASSG4_BxxyyyyCS_LAXMAN.zip). DO NOT add any other files (like temporary files, input files, etc.) except your source code, into the zip archive. The source codes must be named as ASSG<NUMBER>_<ROLLNO>_<FIRSTNAME>_<PROGRAM-NUMBER>.<extension>
(For example: ASSG4_BxxyyyyCS_LAXMAN_1.c). If there is a part a and a part b or a particular question, then name the source files for each part separately as inASSG4_BxxyyyyCS_LAXMAN_1b.c.

If you do not conform to the above naming conventions, your submission might not be recognized by some automated tools, and hence will lead to a score of 0 for the submission. So, make sure that you follow the naming conventions.Standard of Conduct Violations of academic integrity will be severely penalized.

Each student is expected to adhere to high standards of ethical conduct, especially those related to cheating and plagiarism. Any submitted work MUST BE an individual effort. Any academic dishonesty will result in zero marks in the corresponding exam or evaluation and will be reported to the department council for record keeping and for permission to assign F grade in the course. The department policy on academic integrity can be found at:

http://cse.nitc.ac.in/sites/default/files/Academic-Integrity.pdf .

Assignment Questions

1. Write a program to perform depth-first and breadth-first searching on a directed graph.

Input/Output format:

● The first line of the input contains a positive integer n, the number of vertices in the graph, in the range 1 to 10000.

● The subsequent n lines contain the labels of the nodes adjacent to the respective nodes, sorted in ascending order from left to right. If a node has no adjacent nodes, then the line corresponding to its adjacency list will be empty.

● The rest of the input consists of multiple lines, each one containing a three-letter string followed by zero or two integers. The integers, if given, will be in the range 0 to n-1.

- ○ The string “dfs” will be followed by two integers, say start and key. Perform a depth first search for key in the graph by starting a traversal from start. Output the nodes visited by the traversal either till the node labeled key is found, or all the nodes have been traversed without finding a node with label key.
- ○ The string “bfs” will be followed by two integers, say start and key. Perform a breadth first search for key in the graph by starting a traversal from start. Output the nodes visited by the traversal either till the node labeled key is found, or all the nodes have been traversed without finding a node with label key.
- ○ The string “stp” means terminate the program.

● The output, if any, of each command should be printed on a separate line, and the nodes visited during a traversal should be space separated.

Sample Input

```
11

3
6 10
0 7
3 8

5
8
0 4 9
8
```

6

bfs 1 4

bfs 9 2

dfs 4 8

dfs 1 4

bfs 4 8

dfs 9 2

stp

Sample Output

1 3 0 7 8 4

9 8 0 4 3 7

4 8

1 3 7 8 9 4

4 3 8

9 8 4 3 7 0

2. Write a program that implements Dijkstra's algorithm for computing shortest paths in a directed graph with positive edge weights.

Input/output format:

- The first line of the input contains a positive integer n, the number of vertices in the graph, in the range 1 to 1000.
- The subsequent n lines contain the labels of the nodes adjacent to the respective nodes, sorted in ascending order from left to right. If a node has no adjacent nodes, then the line corresponding to its adjacency list will be empty.
- The subsequent n lines contain the weights of the edges corresponding to the adjacency list. The edge weights are positive real numbers in the range (0, 10000]. If a node has no adjacent nodes, then the line corresponding to its adjacent edge weights will be empty.
- The rest of the input consists of multiple lines, each one containing a four-letter string followed by zero, one or two integers. The integers, if given, will be in the range 0 to n-1.
    - The string "apsp" will be followed by a single integer, the label of the source vertex. Output the shortest path distance from the source vertex to all the n vertices in the graph, sorted in the order of their labels, in a space separated format. Print "INF" for nodes that are unreachable from the source vertex.
    - The string "sssp" will be followed by two integers, respectively, labels of the source and

destination nodes. Output the shortest path from the source node to the destination node, if such a path exists. Print “UNREACHABLE”, otherwise.

- The string “stop” means terminate the program.

- The output, if any, of each command should be printed on a separate line.

Sample Input

```
9
1 4
5
3
6

2 7 8
2
4
5 7
2 20
3
7
5

1 6 4
0
2
2 1
apsp 0
sssp 0 6
sssp 0 7
sssp 5 6
sssp 8 7
stop
```

Sample Output

```
0 2 6 13 12 5 18 10 9
18
10
```

13

1

3. Write a C Program for implementing Red Black Tree Insertion. Initially the tree is empty. Input should be read from the file ***input.txt*** and output should be written to the file ***output.txt.*** Your program should contain the following functions:

- insert(R, element) – Inserts the data specified by element into tree R.
- leftRotate(R,k) – Perform left rotation in the tree R, with respect to node k
- rightRotate(R,k) – Perform right rotation in the tree R, with respect to node k
- fixViolation(R, k) - fixes violations caused by insertion of k.
- printTree(R) – prints the tree given by R in the paranthesis format as Tree t is represented as (t(left-subtree)(right-subtree)). Empty parentheses ( ) represents a null tree. There is no space in between paranthesis or numbers.

Input File Format

a) The input consists of multiple lines, each line of the input contains the value to be inserted into the red black tree.

Output File Format:

The output (if any) of each command should be printed on a separate line. If the tree after insertion needs rotation/recoloring it should be printed on the next line. (***eg:In the sample output Lines 3 and 4 corresponding to the tree after insertion of 12 and rotation after the insertion of 12 respectively***)

Sample Input

4

7

12

15

3

5

14

18

Sample Output:

(4B)

(4B (7R))

(4B(7R (12R)))

((4R) 7B (12R))

((4R) 7B(12R (15R)))

((4B) 7B (12B (15R)))

(((3R) 4B) 7B (12B (15R)))

(((3R) 4B (5R)) 7B (12B (15R)))

(((3R) 4B (5R)) 7B (12B ((14R) 15R)))

(((3R) 4B (5R)) 7B ((12R) 14B (15R)))

(((3R) 4B (5R)) 7B ((12R) 14B (15R (18R))))

(((3R) 4B (5R)) 7B ((12B) 14R (15B (18R))))