National University of Computer and Emerging Sciences

School of Computing Summer 2023 Islamabad Campus

CS-2001: Data Structures (Summer 23)

Assignment 3

(Deadline: 26 July, 2023 11:59 PM)

Submission: Combine all your work (solution folder) in one .zip file. Use proper naming convention for your submission file. Name the .zip file as COURSE_ROLL-NUM_01.zip (e.g. DS_19i0412_01.zip). Submit .zip file on Google form within the deadline. Submissions on the email will not be accepted.

Plagiarism: Plagiarism cases will be dealt with strictly. If found plagiarized, both the involved parties will be awarded zero marks in this assignment. Copying from the Internet is the easiest way to get caught!

Deadline: The deadline to submit the assignment is 26 July, 2023 11:59 PM. Correct and timely submission of the assignment is the responsibility of every student.

Comments: Comment your code properly. Write your name and roll number (as a block comment) at the beginning of the solution to each problem.

QUESTION – 1 (Search Engine) [60 Marks]

In this technological age, search engines have become an essential element of our life. This project will need you to create your own search engine. Your software should be able to read information from a file and search it. A console-based application is required of you. You have more flexibility in the code this time. Although we require neat and commented code, there are no constraints on the code structure.

Input Format:

A sample file is provided to you (Assignment_data.txt). In this sample file every pair of lines is a data item. The first of these lines contains an ID and a URL. The second line contains a set of keywords (space separated). There will be a blank line between pairs of lines. If you see two blank lines in a row that means you have reached the end of the file.

Your Program:

At the start, your program should ask for a filename. Then it will read the data from that file and make BST out of it. This BST will be used to search later on. Then, it should present a menu to the user. It should ask for a keyword and then output the list of URLs that match it.

Task 1:

Task 1 is all about setting up the interface for your search engine. Write a simple console-based interface. It doesn't need any functionality, but it should be able to read from a file, ask the user for keywords to search, etc. It's okay to hardcode results at this time. Your code should be able to handle invalid input.

A sample run of your program is given below. Bold lines indicate program output, lines starting with a '>' indicate user entered input:

Please enter a filename: > Assignment_data.txt
File loaded successfully.
Please enter a word to search:
> life
3 result(s) found
1. http://www.gutenberg.net/dickens/otwist/4.html
2. http://www.gutenberg.net/dickens/otwist/32.html
3. http://www.gutenberg.net/dickens/otwist/40.html

Task 2:

Implement the search engine using a **Binary Search Tree.** The basic idea here is to have each node store a keyword, along with associated list of URLs. It would be helpful to implement your BST using a class, with left and right child pointers, similar to what was done in the lectures. All you have to do here is implement the insert, search, and delete operations that were explained. All these operations are required for only keywords.

QUESTION – 2 (Simple Binary Trees) [40 Marks]

Create a class for a simple binary tree. A tree which simply fills values from left to right in a depth level and once depth is filled, we move to next depth level to fill. The tree will be filled by reading the q2.txt file, you can change q2.txt file to check all possible cases. In q2.txt file, '>>>>' indicates the start of a new tree. Make your code in such a way that it runs all test cases and provides output for all 3 parts separately. You must cover all possible cases.

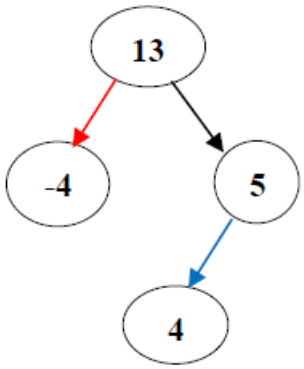
Part 1 [10 Marks] : Given a binary tree, write a function which creates a linked list of all the nodes at each depth (e.g., if you have a tree with depth D, you'll have D linked lists). The tree will be passed as an argument to the function and the function will print and return the number of linked lists created (depth of the tree).

Part 2 [10 Marks] : Tl and T2 are two binary trees, with Tl bigger than T2. Create a function to determine if T2 is a subtree of Tl. A tree T2 is a subtree of Tl if there exists a node n in Tl such that the subtree of n is identical to T2. That is, if you cut off the tree at node n, the two trees would be identical. Both trees are passed as an argument to the function. The function will return true is T2 is a subtree of T1, otherwise it will return false.

Part 3 [20 Marks] : You are given a binary tree in which each node contains an integer value (which might be positive or negative). Write a function to count the number of paths that sum to a given value. The path does not need to start or end at the root or a leaf, but it must go downwards (traveling only from parent nodes to child nodes). The tree and value are passed as an argument to the function. The path count is returned by the function.
Sum of a path is the sum of all node values in that path.

EXAMPLE Input:

Tree =
Value = 9



Output: 2 path

GOOD LUCK!!!