CSE102 – Computer Programming Lab.

Homework11

Selection Operations

Due Date: 21/06/2023

Hand in: A student with number 20180000001 should hand in a zip file named 20180000001.zip for this lab.

Overview

The objective of this assignment is to design and implement a C program that creates and manages **Family Trees**. This task requires you to use **struct** structure in C to define a **nod**e, which will represent a person. The node will include attributes such as **Name**, **Age**, **Mother**, and **Father**. Your program should allow users to create, add nodes to, and remove family trees. It should also provide the ability to load family trees from files and print certain details about specific people, their immediate family, or other relative

Part 1. (40 points)

In this part, your task is to create the system that allows a user to add and remove family trees. Each family tree should have its name and should be stored in a text file with the format "**familyX.txt**", where X is a unique identifier.

A node in the tree should be a struct containing the following attributes:

1. ***Name***
2. ***Age***
3. ***Mother (a pointer to another struct)***
4. ***Father (a pointer to another struct)***

You should create functionality to **add** and **remove** nodes (persons) to and from the family tree. For the **root nodes**(very first Grand Parents) set Mother and Father pointers as **NULL**. If a node has children and the user tries to remove it, make sure to warn the user that the node's children will also be removed. Each time a node is added or removed, the system should update the corresponding text file.

Requirements

1. Implement an option in the menu to add a family tree. This option should ask the user for the family tree name and create a corresponding text file.
2. Implement an option in the menu to remove a family tree. This option should ask the user for the family tree name and delete the corresponding text file.

3. Each new node (person) introduced to the family trees should be added via the menu. The system should ask for the name, age, and the names of the person's mother and father. If the mother and father are not already present in the tree, the system should warn the user. If it is root node, Mother and Father pointers should be **NULL**.

Part 2. (60 points)

In this part, your task is to allow the system to load a family tree from a text file. After loading the family tree, the user should be able to perform various operations, such as adding and removing persons(nodes), printing a person's nuclear family, and finding relatives of given person.

Requirements

1. Implement an option in the menu to load a family tree. This option should ask the user to select a text file and load the corresponding family tree into the system.
2. The user should be able to add and remove persons from the loaded family tree. For remove operation, a node with an entered name and parent names should be able to be removed via the menu. If the person has children, warn the user before deleting the node because children will also be removed afterwards.
3. Implement an option to print a person's nuclear family(Father, Mother, Spouse, Children). This option should print the person's parents, siblings, spouse, and children, if any.
4. Search Relatives of Given Person: The program should print out the relatives of a given person based on the user's selection of relationship type (siblings, parents, grandparents, cousins, spouse).
5. All changes made to the family tree should be saved to the corresponding text file.

Note: In the Family Tree, every member should have a unique name to avoid conflicts.

Example Outputs:

```
*****************************************
Family Tree System Menu:
1-Add Family Tree
2-Remove Family Tree
3-Load Family Tree
4-Exit
>2
Enter the name of the txt file of the family tree: family1.txt
family1.txt has been removed successfully…
```

```
*****************************************
Family Tree System Menu:
1-Add Family Tree
2-Remove Family Tree
3-Load Family Tree
4-Exit
>1
1-Add node
2-Save Family Tree
3-Exit
>1
Enter the name of the person: John
Enter the age of the person: 35
...
```

```
Family Tree System Menu:
1-Add Family Tree
2-Remove Family Tree
3-Load Family Tree
4-Exit
>1
1-Add node
2-Save Family Tree
3-Exit
>1
Enter the name of the person: Alice
Enter the age of the person: 30
Enter the name of the mother: Jane
Enter the name of the father: John

Alice has been added to the family tree successfully…

1-Add node
2-Save Family Tree
3-Exit
```

```
*****************************************
Family Tree System Menu:
1-Add Family Tree
2-Remove Family Tree
3-Load Family Tree
4-Exit
>3
Listing Family Trees:
1-Family1.txt
2-Family2.txt
>1
Family1.txt has been selected…
Select an operation
1-Add new person
2-Remove a person
3-Print a Person's nuclear family
4-Search Relatives of Given Person
5-Return to the main menu
```

```
Family Tree System Menu:
1-Add Family Tree
2-Remove Family Tree
3-Load Family Tree
4-Exit
>3
Listing Family Trees:
1-Family1.txt
2-Family2.txt
>2
Family2.txt has been selected
Select an operation
1-Add new person
2-Remove a person
3-Print a Person's nuclear family
4-Search Relatives of Given Person
5-Return to the main menu
>2
Enter the name of the person you want to remove: Alice
Warning: Alice has children, removing Alice will also remove her children.
Do you still want to continue? Yes
Alice and her children have been removed from the family tree.
```

```
Family Tree System Menu:
1-Add Family Tree
2-Remove Family Tree
3-Load Family Tree
4-Exit
>3
Listing Family Trees:
1-Family1.txt
2-Family2.txt
>1
Family1.txt has been selected
Select an operation
1-Add new person
2-Remove a person
3-Print a Person's nuclear family
4-Search Relatives of Given Person
5-Return to the main menu
>3
Enter the name of the person: Bob
Printing nuclear family of Bob:
Father: John
Mother: Jane
Siblings: Alice
Children: Charlie, Daisy

Family1.txt has been selected
Select an operation
1-Add new person
2-Remove a person
3-Print a Person's nuclear family
4-Search Relatives of Given Person
5-Return to the main menu
>4
Enter the name of the person: Charlie
Select type of relatives:
1-Parents
2-Spouse
3-Children
4-Grandparents
5-Cousins
>1
Printing parents of Charlie:
Father: Bob
Mother: Emily
```

```
Family1.txt has been selected
Select an operation
1-Add new person
2-Remove a person
3-Print a Person's nuclear family
4-Search Relatives of Given Person
5-Return to the main menu
>4
Enter the name of the person: Daisy
Select type of relatives:
1-Parents
2-Spouse
3-Children
4-Grandparents
5-Cousins
>5
Printing cousins of Daisy:
Cousins: None
```

```
Family1.txt has been selected
Select an operation
1-Add new person
2-Remove a person
3-Print a Person's nuclear family
4-Search Relatives of Given Person
5-Return to the main menu
>4
Enter the name of the person: Emily
Select type of relatives:
1-Parents
2-Spouse
3-Children
4-Grandparents
5-Cousins
>2
Printing spouse of Emily:
Spouse: Bob
```

General Rules:

1. You will have two hours to provide a solution to the given problem set.
2. You will be able to hand in your solutions via Teams in the next two hours. The submission will be closed exactly at 10am.
3. There will be an interview session immediately after the submission deadline. Starting at 10am, you will be randomly invited to attend a meeting by a TA to demonstrate your solution and answer any questions asked by the TA.
4. You must be available until 1pm to respond to the demo invitation whenever you receive it. You will have 3 minutes after you are called via Teams. If you do not answer/appear in 3 minutes, you will miss you interview.
5. If you miss your interview or are unable to give satisfactory answers to the questions, you will receive a zero for that lab even if you have submitted your solution.
6. If you have not submitted a solution in time, you will not be invited for the interview and receive zero for that lab.
7. Due to time constraints, some students may not be invited to an interview. In that case, their solutions will be graded offline.
8. Unless you aren't declared for a specific prototype, you may use arbitrary but proper function and variable names that evoke its functionality.
9. The solution must be developed on given version of OS and must be compiled with GCC compiler, any problem which rises due to using another OS or compiler won't be tolerated.
10. Note that if any part of your program is not working as expected, then you can get zero from the related part, even it is working partially.
11. Zip your solution file before uploading it to MS Teams. The zip file must contain the C file with your solution and screenshots of the valid outputs of the program.