COMP2611: Data Structures
Project Assignment

DUE DATE :
Sunday 26 November, 2023 at Midnight

Objective:

Develop a graphical user interface (GUI) using wxWidgets in Linux, which implements the operations of the following Abstract Data Types (ADTs):

1. Binary Search Tree (BST)
2. AVL Tree
3. Red-Black Tree
4. Splay Tree
5. Set
6. Min Heap Tree.
7. Max Heap Tree.

The project is an extension of the work, which was already started in **Assignment #1,** but will now involve the hierarchical and Set Abstract Data Types within a GUI setting and a binary file.

The **Random-Access File** (RAF), "**Clients.dat**", contains records of clients who booked their travels with the Pelican Travel Agency. The records contain fields of:

Data	**Data Type**
Client number	integer
Name	char array [20]
Surname	char array [20]
Destination	char array [20]
Payment	char array [15]
Booking	char array [15]

e.g.

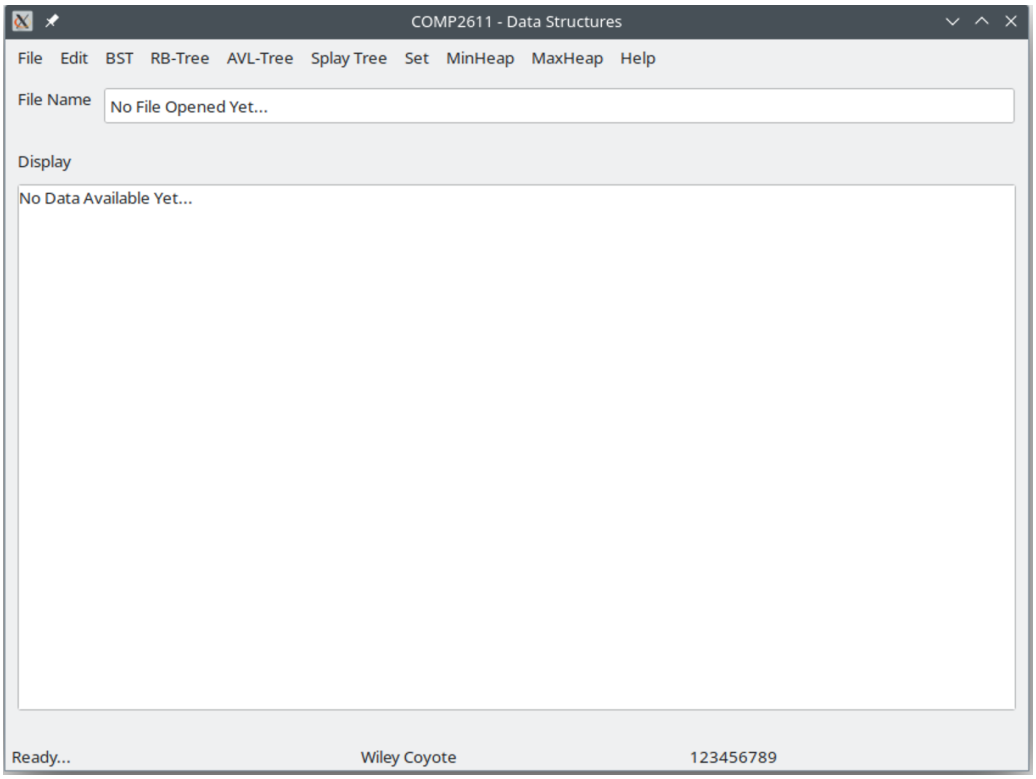
Client No.	Name	Surname	Destination	Payment	Booking
51760	Herbert	Burke	Canada	Visa	Internet
50813	Tyrone	Arthur	St Lucia	Amex	Gifted
49673	Shawn	Bynoe	Jamaica	Diner	Telephone
49673	Shawn	Bynoe	Trinidad	MasterCard	Internet

Read the data from the RAF "**Clients.dat**" to populate the ADTs of your project as follows:

1.	**BST**	Payment is MasterCard
2.	**AVL Tree**	Payment is Diner
3.	**RB-Tree**	Payment is Visa
4.	**Splay-Tree**	Payment is Amex
5.	**Set**	Set A – Internet Booking. Set B – Walk-in Booking. Intersection set – Records where Visa was used for Payment.

6.	**MinHeap**	All records. Heapness is based on Client number.
7.	**MinHeap**	All records. Heapness is based on Client number.

Your GUI should appear as the following:



The main menu bar of your GUI, with the associated sub-menu items should appear as follows:

File	**Edit**	**BST**	**AVL Tree**	**RB Tree**
• Open File • Display File • Save • Save As • Exit	• Create ADTs • Add Record • Delete Record	• Inorder • Preorder • Postorder	• Inorder • Preorder • Postorder	• Inorder • Preorder • Postorder

Splay Tree	**Min Heap**	**Max Heap**	**Set**	**Help**
• Inorder • Preorder • Postorder	• Display All • Heap Sort	• Display All • Heap Sort	• Display SetA • Display SetB • Display Intersection • Display Union	• About • Exit

In Addition:

The functionalities of the sub-menu items are self-explanatory. However, the following should be noted:

1. The data file "Clients.dat", should **NOT** be hard-coded in your program, but should be opened from the **system fileOpen dialog.**

2. When the file is opened, as well as when the **Display File** (in menu option **File**) is clicked, the file's contents should be **immediately** displayed in the main textbox. The full path of the file should also be displayed in the filename text box.

3. The key field for the record is the Client Number.

4. When the "**Create ADTs**" sub-menu item is clicked (in the Edit main menu item), the program should read the data records from the data file and populate **ALL the ADTs** which are affected by the details of the record as stipulated by the third table above.

5. When the "**Add Record**" sub-menu item is clicked (in the Edit main menu item), the program should open an input dialog where the details of the new record should be entered. Once OK is selected, **ALL the ADTs** that are affected by the details of the record as stipulated by the third table above should be updated with the new record.

6. When the "**Delete Record**" sub-menu item is clicked (in the Edit main menu item), the program should open an input dialog where the Client Number is to be entered. Once OK is selected, the record should be deleted from **ALL the ADTs** that contain the record with the Client Number.

7. In the display functions, the records should be displayed one record per line.

8. The nodes in the AVL tree must contain an attribute to describe the weight of the node (i.e., negative for **right-heavy**, positive for **left-heavy,** and zero for **balanced**), which must be displayed in brackets at the end of the line when the AVL tree is traversed.

9. The nodes of the Red-Black tree must contain an attribute to describe the node's colour (**R** or **B**), which must also be displayed when the Red-black tree is traversed.

10. The output results of all the menu functions should be displayed in the **main text box within your GUI**.

11. When **Open File** is clicked (in the File main menu item), the **system fileOpen dialog** should be opened with the option to display files of type: **Data (*.dat), Text (*.txt),** and **All (*.*).** Once the contents of a file are displayed or the result of some processing is displayed, the menu selections of **Save** and **Save As** should open the corresponding dialogs to perform the desired task. **Save As** should allow the user to specify a file name and file type into which the contents can be saved. These two functions **ONLY** save the contents of the main textbox and the functions' code has been provided from the first project.

12. Before each display operation is carried out, the display (main) textbox should be cleared.

13. The operation indicated by the sub-menu item should then be carried out on that particular ADT **ONLY**. The other ADTs **should NOT** be affected by the operations of another ADT.

14. The **Exit** menu item is to close the program

15. The **About** menu item should produce an output dialog box with suitable information about the programmer, the program, and architecture of the machine the program is running on.

16. When the cursor is placed on a sub-menu option, a description of the menu option should be displayed in the first partition of the status bar. At all other times, the string, "**Ready…**" should be displayed in that section of the status bar. The second and third sections of the status bar should contain your name and ID number respectively.

SUBMISSION:

Zip up all the files into a zip file with your ID number and "-Project02" as the file's name (e.g. **123456789-Project02.zip**) and submit it through the course's eLearning portal through the label "**Project Assignment #2**" no later than **Sunday 26 November, 2023 at midnight**. You may submit your zipped file multiple times as a **draft**, before your final submission; only the most recent copy of your submission will be retained. In fact, you are strongly encouraged to use the portal as a repository for your developing project. However, the portal will close at exactly midnight on Sunday, 26 November, 2023.

Submissions beyond the deadline will not be accepted.

Tips:

1. Start working on your project immediately and plan to finish at least a week in advance of the deadline date. Avoid, at all costs, trying to submit your assignment on the night of the deadline.
2. Build your code incrementally. Add one functionality at a time. At every stage in your code development, you should have a working project. In the unlikely event you did not complete the assignment by the due date, submit what you have completed and make sure those parts are working. **Projects that do not compile will be given an automatic zero and will not be assessed any further.**
3. Save your work often and maintain multiple backup copies. At the very least, make a backup copy for each new addition you make in your project.
4. All projects MUST be submitted via the eLearning portal.