FIRST SEMESTER SESSION 2022/2023 (A221)

STIA2024(B) DATA STRUCTURES AND ALGORITHM ANALYSIS

UUM COLLEGE OF ARTS AND SCIENCES

==

PROJECT REPORT

==

Submitted to:

Dr. Sharhida Zawani Binti Saad

Prepared by:

	NAME	MATRIC.NO
1	YAP JIA QING	278688
2	VINCENT BEH HUA EIK	279018
3	POON WAI KIT	279021
4	YAP YUN LOON	279231

Presentation Link: https://youtu.be/5o4zMmquiiU

Table of Content

1.0 Introduction

Welcome to our Java-based vacation program! This program is designed to assist you in planning and saving your perfect vacation or holiday. It allows you to capture and process useful information related to travel and tourism, including information about locations, activities, facilities, budgets and the number of days you want for your vacation. The program is user-friendly and easy to navigate, with a variety of options for inputting and manipulating data. You can add new information, update existing information, delete information that is no longer relevant, and search for specific information. The program also features a display function, which allows you to view the results of your add, update, delete, and search operations in a clear and easy-to-read format. The program utilizes a Graphical User Interface (GUI) to make it easy for you to view the results of your operations. Whether you're planning a family vacation, a romantic getaway, or a solo adventure, this program is the perfect tool to help you organize and plan your trip.

2.0 Problem Statement

Many people find it difficult to plan and organize their vacation or holiday due to the vast amount of information available and the many options to consider. There is a need for a user-friendly program that can assist in capturing and processing relevant information related to travel, tourism, and sightseeing, and provide the ability to add, update, delete, search and display this information in a clear and easy-to-read format. The program should also be designed to be easy to navigate and use, with a graphical user interface that makes it easy for the user to view the results of their operations. The goal of this program is to make the process of planning a vacation or holiday simpler and more efficient for the user.

3.0 List of Objectives

1. To develop a Java-based program that can assist users in capturing and processing relevant information related to travel, tourism, and sightseeing.

2. To provide the ability to add, update, delete, search and display this information in a clear and easy-to-read format.

3. To design the program to be user-friendly and easy to navigate, with a graphical user interface that makes it easy for the user to view the results of their operations.

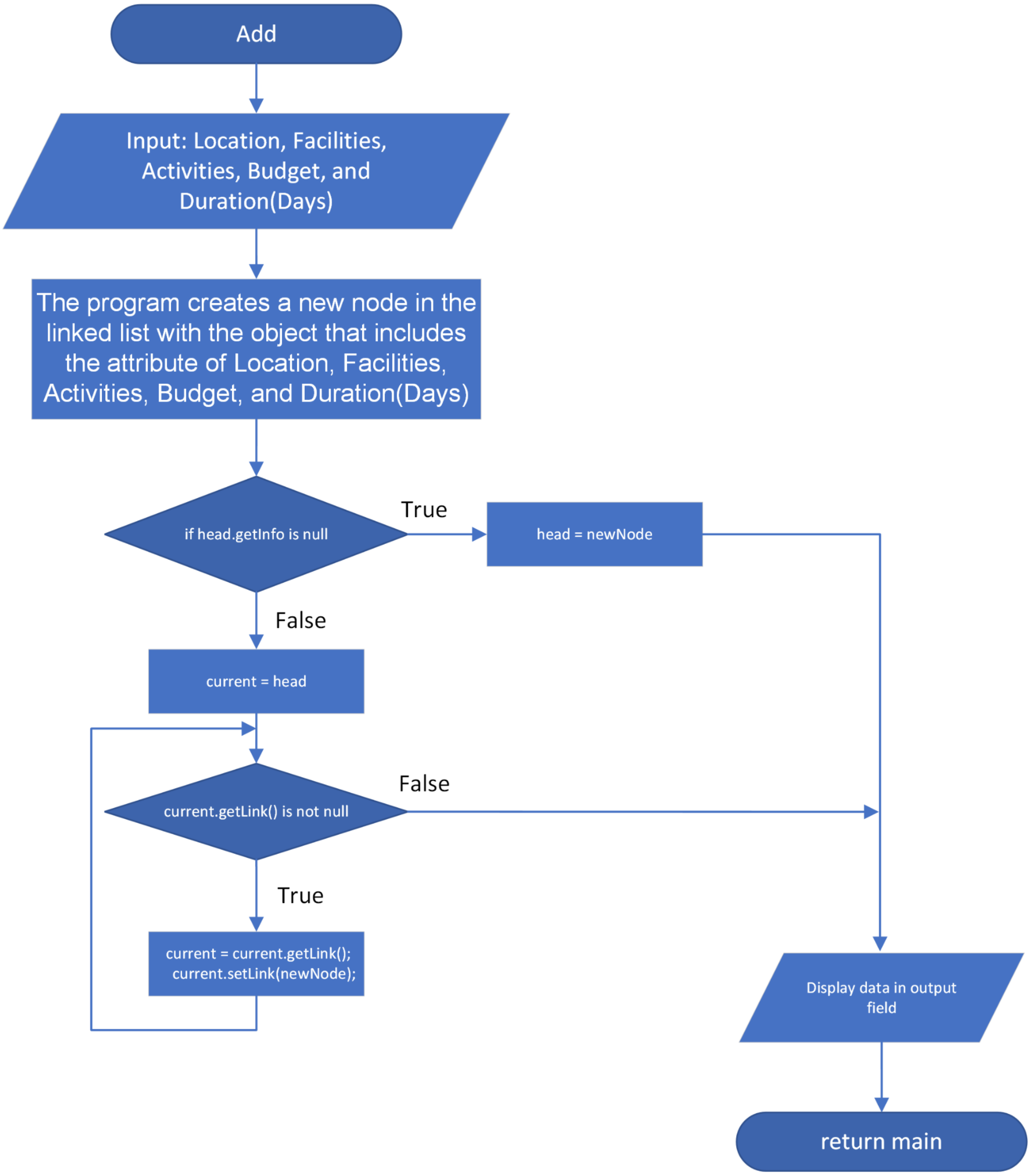
4. To make the process of planning a vacation or holiday simpler and more efficient for the user.

4.0 List of Requirements

1. The program must be able to create and manage a linked list of objects which will contain locations, activities, facilities, budgets and the number of days you want for your vacation.
2. The program must allow the user to add new data to the linked list.
3. The program must allow the user to update existing data in the linked list.
4. The program must allow the user to delete data from the linked list.
5. The program must allow the user to search for said vacation objects using their location in the linked list.
6. The program must allow the user to display the data in the linked list in a clear and easy-to-read format.
7. The program must utilize a graphical user interface (GUI) to make it easy for the user to view the results of their operations.

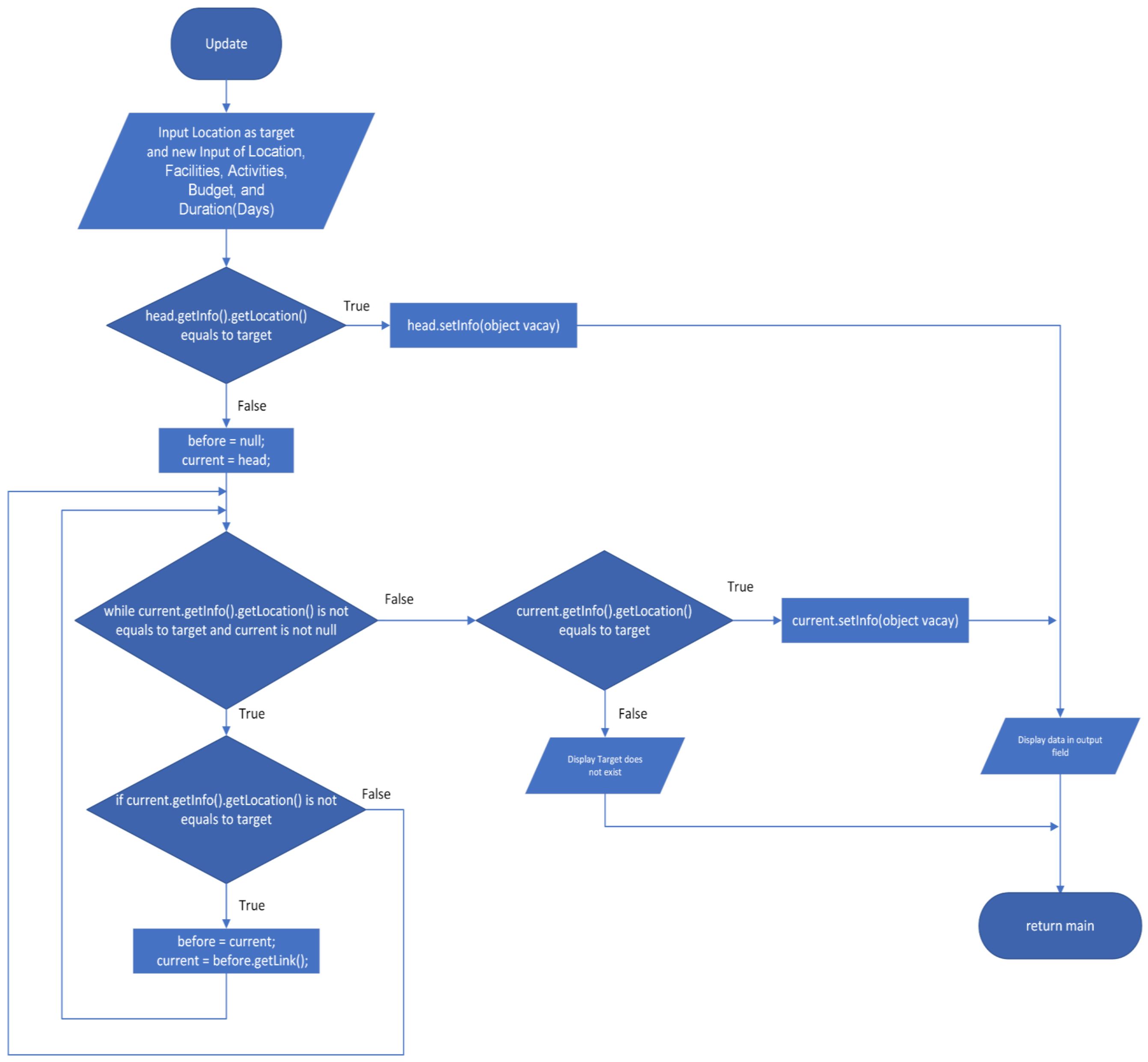
5.0 Flowchart for each Algorithm

<u>Adding information into the linked list</u>


Add
Input: Location, Facilities, Activities, Budget, and Duration(Days)
The program creates a new node in the linked list with the object that includes the attribute of Location, Facilities, Activities, Budget, and Duration(Days)
if head.getInfo is null
True
head = newNode
False
current = head
current.getLink() is not null
False
True
current = current.getLink();
current.setLink(newNode);
Display data in output field
return main

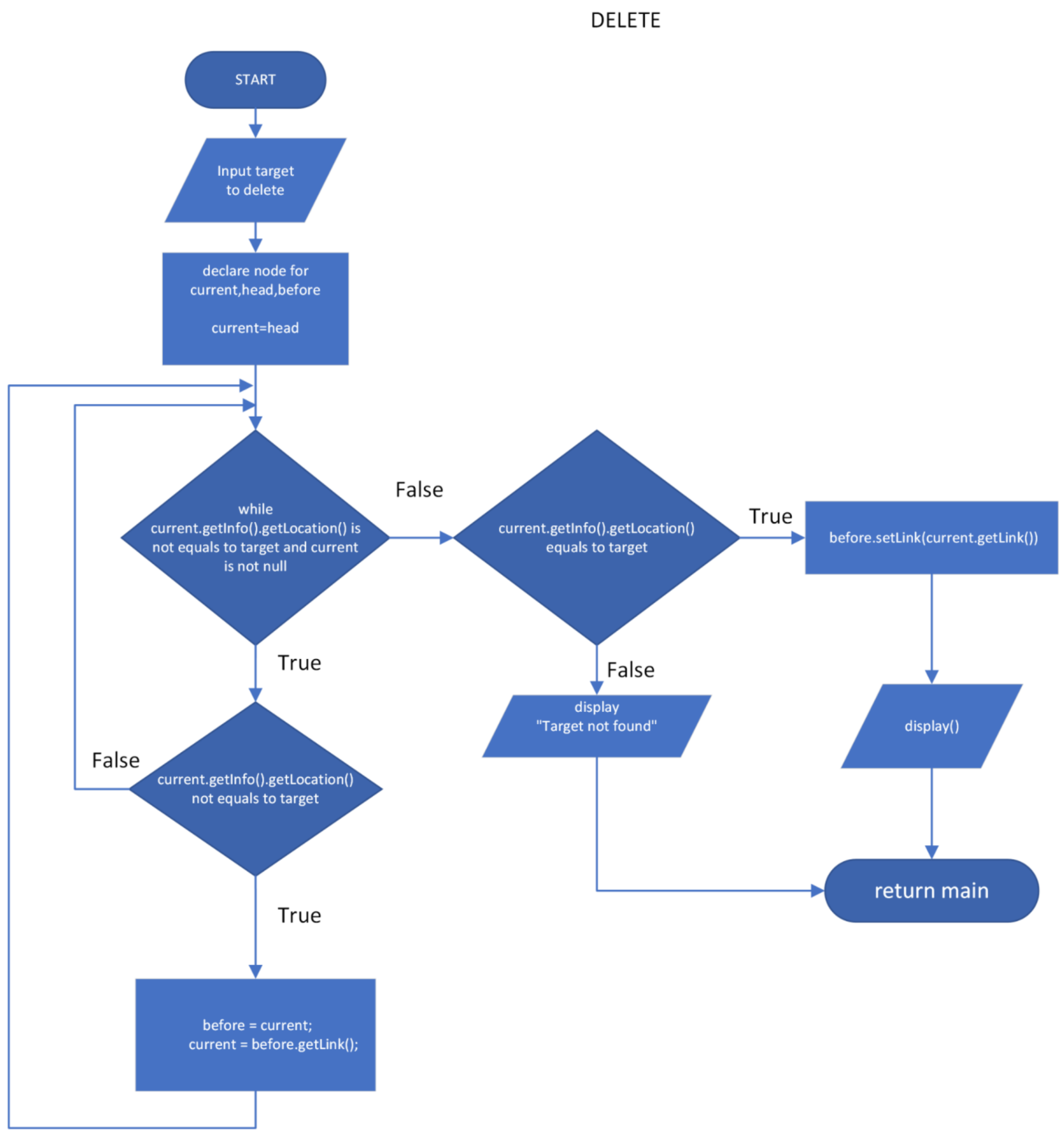
```
try{
        boolean check = true;
        boolean noduplicate = true;
        String location = locInput.getText();
        String fac = facInput.getText();
        String activity = actInput.getText();
        double budget = Double.parseDouble(budInput.getText());
        int days = Integer.parseInt(durInput.getText());


if(location.equals("")||fac.equals("")||activity.equals("")||budInput.getTe
xt() == null||durInput.getText() == null||budget<=0||days<=0){
            check = false;
        }

        if(check){

        Vacay vacay = new Vacay(location, activity, fac, budget, days);

        Node newNode = new Node(vacay);
        if(head==null){
            head=newNode;
            }

        else if (head.getInfo()== null) {
            head = newNode;
           /* if(head.getInfo().getLocation().equals(location)){
                JOptionPane.showMessageDialog(null, "This location had been
filled", "Error", JOptionPane.ERROR_MESSAGE);
            }*/
        } else {
            current = head;
            // traversing nodes
            if(head.getInfo().getLocation().equals(location)){
                    JOptionPane.showMessageDialog(null, "This location had
been filled", "Error", JOptionPane.ERROR_MESSAGE);
                    noduplicate = false;
                }
            while (current.getLink() != null) // current.link != null
            {
                current = current.getLink(); // current = current.link

                if(current.getInfo().getLocation().equals(location)){
                    JOptionPane.showMessageDialog(null, "This location had
been filled", "Error", JOptionPane.ERROR_MESSAGE);
                    noduplicate = false;
                }
            }
            if(noduplicate)
            current.setLink(newNode); // current.link= newNode
            count++;
}

        locInput.setText("");
        facInput.setText("");
        actInput.setText("");
        budInput.setText("");
        durInput.setText("");

        current = head;
        String output = "";
        String finalOutput = "";

        if (current == null) {
            displayOutput.setText("empty");
        }
        if(noduplicate){
        JOptionPane.showMessageDialog(null, "Vacation has been updated!",
"Add", JOptionPane.INFORMATION_MESSAGE);
        String title = String.format("%1$-40s%2$-30s%3$-20s%4$-15s%5$-
5s%n", "Location", "Facilities", "Activity", "Budget(RM)",
"Duration(Days)");
        String header =
"==============================================================================
==============================================\n";
        while (current != null) {

            output = String.format("%1$-40s%2$-30s%3$-20s%4$-15.2f%5$-
5s%n", current.getInfo().getLocation(), current.getInfo().getFac(),
current.getInfo().getActivity(), current.getInfo().getBudget(),
current.getInfo().getDays());
            finalOutput += output;
            current = current.getLink();
        }

        displayOutput.setText(title + header + finalOutput);
        }
        }
        else{
            JOptionPane.showMessageDialog(null, "Please fill in the
respective fields correctly", "Error", JOptionPane.ERROR_MESSAGE);
        }
        }catch(NumberFormatException e){
            JOptionPane.showMessageDialog(null, "Please only fill numbers
in the Budget/Duration field!", "Error", JOptionPane.ERROR_MESSAGE);
        }
```

Updating information into the linked list


Update
Input Location as target and new Input of Location, Facilities, Activities, Budget, and Duration(Days)
head.getInfo().getLocation() equals to target
True
head.setInfo(object vacay)
False
before = null; current = head;
while current.getInfo().getLocation() is not equals to target and current is not null
False
current.getInfo().getLocation() equals to target
True
current.setInfo(object vacay)
True
False
Display Target does not exist
Display data in output field
if current.getInfo().getLocation() is not equals to target
False
True
before = current; current = before.getLink();
return main

```
boolean noduplicate = true;

        try{
        target = targetInput.getText();
        System.out.println(target);

        String location = locInput.getText();
        String fac = facInput.getText();
        String activity = actInput.getText();
        double budget = Double.parseDouble(budInput.getText());
        int days = Integer.parseInt(durInput.getText());

        Vacay vacay = new Vacay(location, activity, fac, budget, days);


        if (head.getInfo().getLocation().equals(target)) {
            head.setInfo(vacay);

        } else {
            before = null;
            current = head;

            // current.info != target
            while ((!current.getInfo().getLocation().equals(target)) &&
(current != null)) {
                if (!current.getInfo().getLocation().equals(target)) {
                    before = current;
                    current = before.getLink();
                    if(current.getInfo().getLocation().equals(target)){

                    noduplicate = false;

                }
                }// current = current.link
            }
            if(current.getInfo().getLocation().equals(location)){
                    noduplicate =true;
                    }
            if(noduplicate){
            if (current.getInfo().getLocation().equals(target)) {  //
current.info == target
                current.setInfo(vacay);
            }
            }
            else{
                JOptionPane.showMessageDialog(null, "This location had been
filled", "Error", JOptionPane.ERROR_MESSAGE);
            }
        }

        current = head;
        String output = "";
        String finalOutput = "";

        if (current == null) {
            displayOutput.setText("empty");
        }
        if(noduplicate){
        JOptionPane.showMessageDialog(null, "Update has been performed!",
"Update", JOptionPane.INFORMATION_MESSAGE);
        String title = String.format("%1$-40s%2$-30s%3$-20s%4$-15s%5$-
5s%n", "Location", "Facilities", "Activity", "Budget(RM)",
"Duration(Days)");
        String header =
"==========================================================================
============================================\n";
        while (current != null) {

            output = String.format("%1$-40s%2$-30s%3$-20s%4$-15.2f%5$-
5s%n", current.getInfo().getLocation(), current.getInfo().getFac(),
current.getInfo().getActivity(), current.getInfo().getBudget(),
current.getInfo().getDays());
            finalOutput += output;
            current = current.getLink();
        }

        displayOutput.setText(title + header + finalOutput);

        }
        }catch(Exception e){
            JOptionPane.showMessageDialog(null, "Please fill the location
that is included in the list only", "Error", JOptionPane.ERROR_MESSAGE);
        }
        locInput.setText("");
        facInput.setText("");
        actInput.setText("");
        budInput.setText("");
        durInput.setText("");
        targetInput.setText("");
```

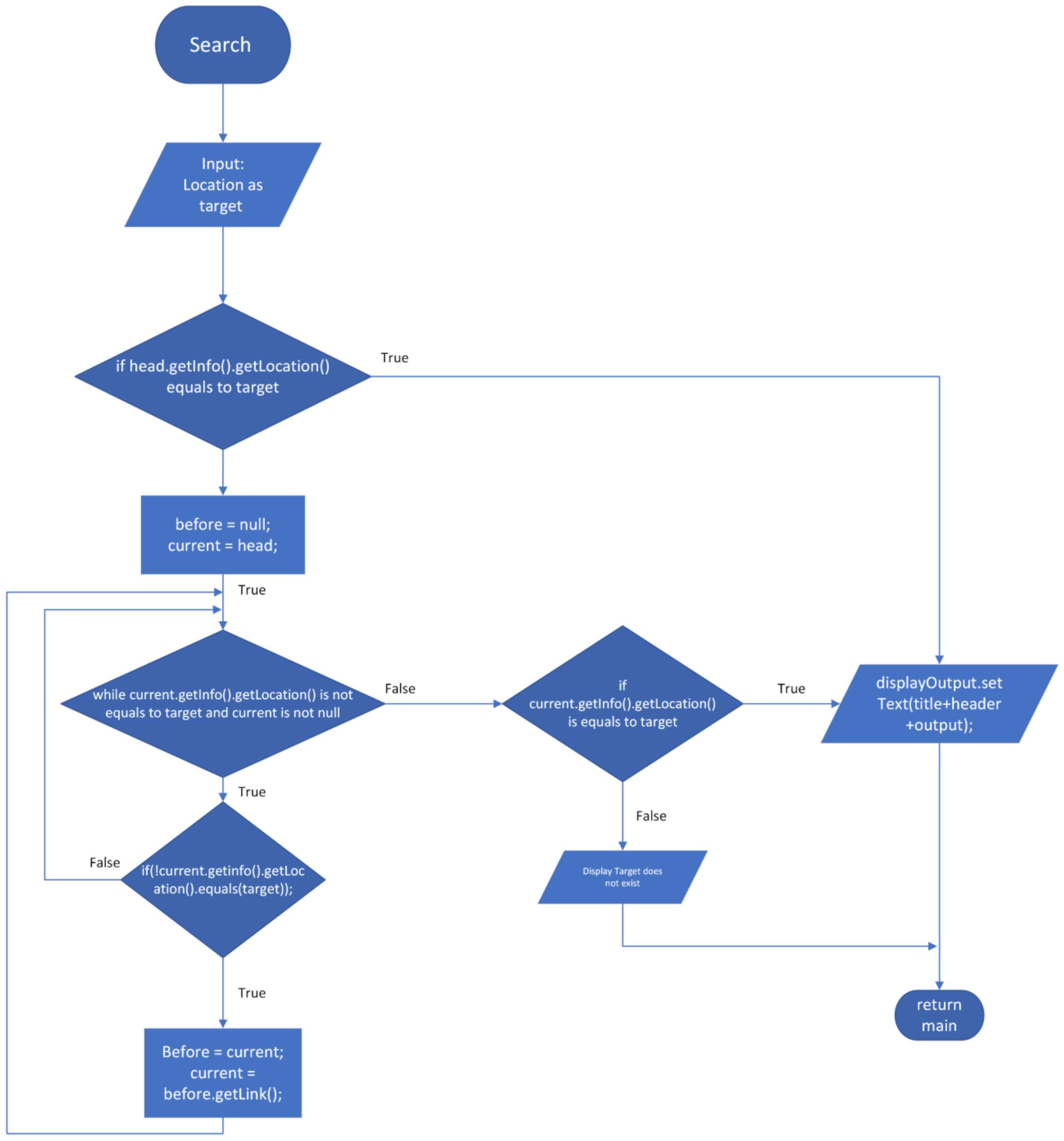
Deleting information from the linked list


DELETE
START
Input target to delete
declare node for current,head,before
current=head
while current.getInfo().getLocation() is not equals to target and current is not null
False
True
current.getInfo().getLocation() equals to target
True
False
before.setLink(current.getLink())
display "Target not found"
display()
return main
current.getInfo().getLocation() not equals to target
False
True
before = current;
current = before.getLink();

```
try{
        target = targetInput.getText();

        if (head.getInfo().getLocation().equals(target)) {
            if(head.getLink()!=null){
            head = head.getLink();}
            else{
               head=null;

            }
        } else {
            before = null;
            current = head;

            // current.info != target
            while ((!current.getInfo().getLocation().equals(target)) &&
(current != null)) {

                before = current;
                current = before.getLink();
                // current = current.link
            }
            if (current.getInfo().getLocation().equals(target)) {  //
current.info == target
                before.setLink(current.getLink());

            } else {
                System.out.println("Target no found");
            }
        }

        String output = "";
        String finalOutput = "";
        String title = String.format("%1$-40s%2$-30s%3$-20s%4$-15s%5$-
5s%n", "Location", "Facilities", "Activity", "Budget(RM)",
"Duration(Days)");
        String header =
"===========================================================================
============================================\n";
        current = head;

        while (current != null) {

            output = String.format("%1$-40s%2$-30s%3$-20s%4$-15s%5$-5s%n",
current.getInfo().getLocation(), current.getInfo().getFac(),
current.getInfo().getActivity(), current.getInfo().getBudget(),
current.getInfo().getDays());
            finalOutput += output;
            current = current.getLink();
        }
        JOptionPane.showMessageDialog(null, "Delete has been performed!",
"Delete", JOptionPane.INFORMATION_MESSAGE);
        displayOutput.setText(title + header + finalOutput);
        }catch(Exception e){
            JOptionPane.showMessageDialog(null, "Please fill the location
that is included in the list only", "Error", JOptionPane.ERROR_MESSAGE);
        }
```

```
        targetInput.setText("");
        if(head == null){
            displayOutput.setText("The list is empty");
        }
```

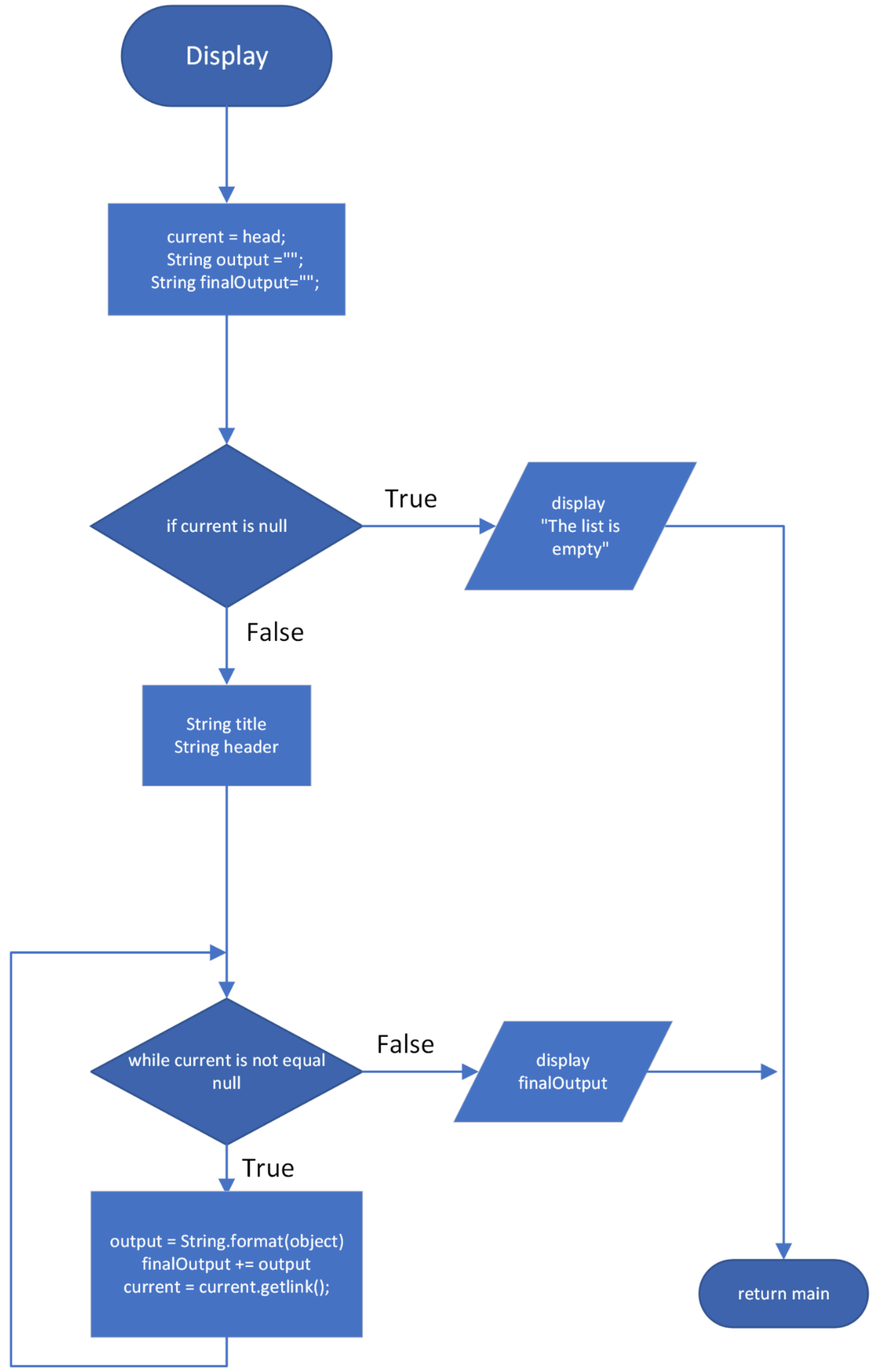
Searching information from the linked list


Search
Input: Location as target
if head.getInfo().getLocation() equals to target
True
before = null; current = head;
True
while current.getInfo().getLocation() is not equals to target and current is not null
False
if current.getInfo().getLocation() is equals to target
True
displayOutput.setText(title+header +output);
True
False
if(!current.getinfo().getLocation().equals(target));
False
Display Target does not exist
True
Before = current; current = before.getLink();
return main

```java
try{
        target = targetInput.getText();

        if (head.getInfo().getLocation().equals(target)) {
            current = head;
            String output = "";
            String finalOutput = "";

            if (current == null) {
                displayOutput.setText("empty");
            }

            String title = String.format("%1$-40s%2$-30s%3$-20s%4$-15s%5$-
5s%n", "Location", "Facilities", "Activity", "Budget(RM)",
"Duration(Days)");
            String header =
"==========================================================================
============================================\n";

            output = String.format("%1$-40s%2$-30s%3$-20s%4$-15.2f%5$-
5s%n", current.getInfo().getLocation(), current.getInfo().getFac(),
current.getInfo().getActivity(), current.getInfo().getBudget(),
current.getInfo().getDays());
            JOptionPane.showMessageDialog(null, "Search has been
performed!", "Search", JOptionPane.INFORMATION_MESSAGE);
            displayOutput.setText(title + header + output);

            locInput.setText(head.getInfo().getLocation());
            facInput.setText(head.getInfo().getFac());
            actInput.setText(head.getInfo().getActivity());
            budInput.setText(Double.toString(head.getInfo().getBudget()));
            durInput.setText(Integer.toString(head.getInfo().getDays()));

        } else {
            before = null;
            current = head;

            // current.info != target
            while ((!current.getInfo().getLocation().equals(target)) &&
(current != null)) {
                if (!current.getInfo().getLocation().equals(target)) {
                    before = current;
                    current = before.getLink();
                }// current = current.link
            }
            if (current.getInfo().getLocation().equals(target)) {  //
current.info == target

                String output = "";
                String finalOutput = "";

                if (current == null) {
                    displayOutput.setText("empty");
                }
                String title = String.format("%1$-40s%2$-30s%3$-20s%4$-
15s%5$-5s%n", "Location", "Facilities", "Activity", "Budget(RM)",
"Duration(Days)");
```

```
                String header =
"==========================================================================
============================================\n";

                output = String.format("%1$-40s%2$-30s%3$-20s%4$-15.2f%5$-
5s%n", current.getInfo().getLocation(), current.getInfo().getFac(),
current.getInfo().getActivity(), current.getInfo().getBudget(),
current.getInfo().getDays());
                displayOutput.setText(title + header + output);


            }
            locInput.setText(current.getInfo().getLocation());
            facInput.setText(current.getInfo().getFac());
            actInput.setText(current.getInfo().getActivity());

budInput.setText(Double.toString(current.getInfo().getBudget()));

durInput.setText(Integer.toString(current.getInfo().getDays()));
        }
        }catch(Exception e){
            JOptionPane.showMessageDialog(null, "Please fill the location
that is included in the list only", "Error", JOptionPane.ERROR_MESSAGE);
        }
```

Displaying information from the list



```
try {

            title = String.format("%1$-40s%2$-30s%3$-20s%4$-15s%5$-5s%n",
"Location", "Facilities", "Activity", "Budget(RM)", "Duration(Days)");
            header =
"==========================================================================
============================================\n";
            while (current != null) {

                output = String.format("%1$-40s%2$-30s%3$-20s%4$-15s%5$-
5s%n", current.getInfo().getLocation(), current.getInfo().getFac(), cur-
rent.getInfo().getActivity(), current.getInfo().getBudget(), current.get-
Info().getDays());
                finalOutput += output;
                current = current.getLink();
            }
            displayOutput.setText(title + header + finalOutput);

        } catch (NullPointerException e) {
            displayOutput.setText("The list is empty");
}
```

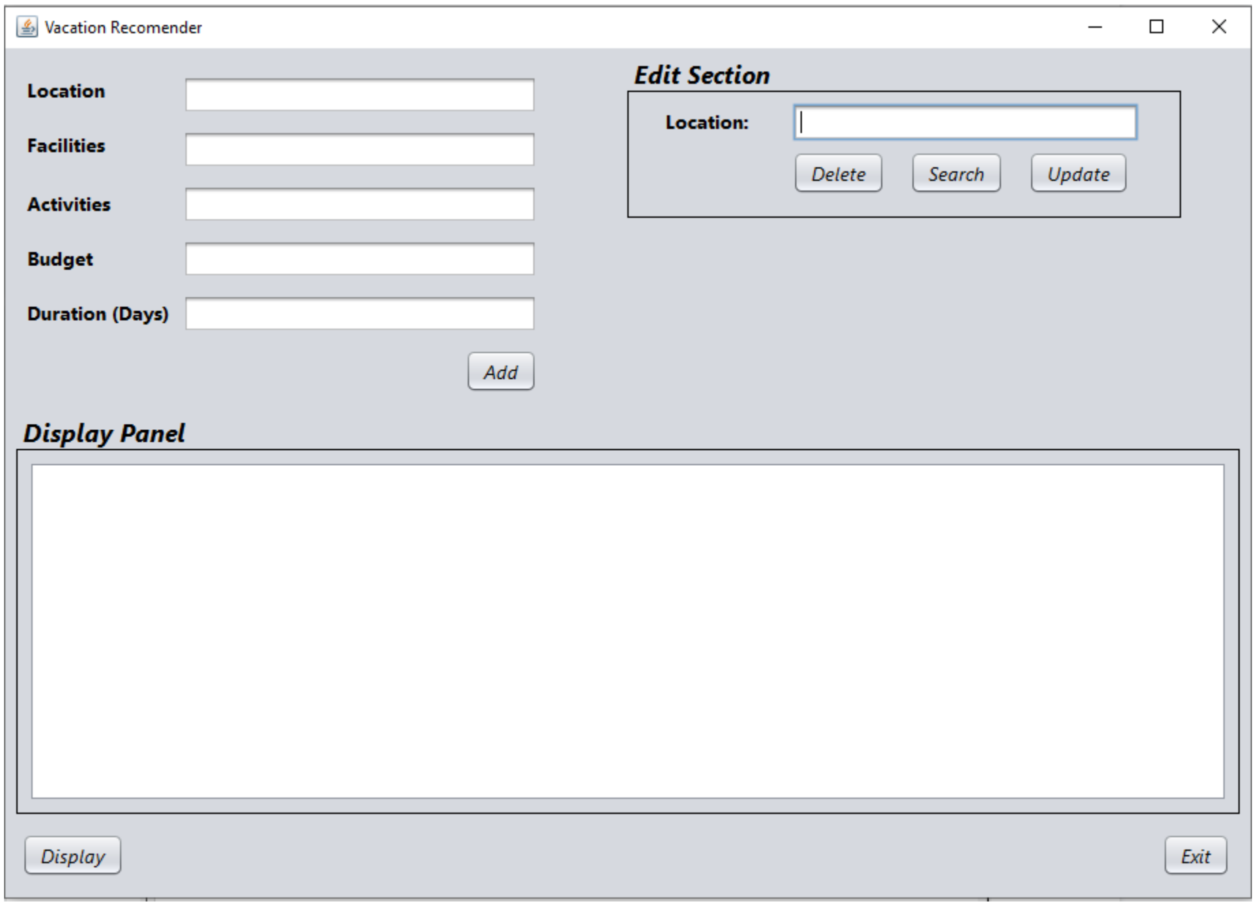
6.0 Data Structure Concept Applied

A linked list is a data structure that is used to store a collection of items, where each item is represented by a node. Each node contains two parts: data and a reference to the next node. This makes it a linear and dynamic data structure where elements are not stored at contiguous memory locations, and each element points to the next element.

In the case of this Java-based vacation program, the program uses a linked list to store information about locations, activities, facilities, budgets and the number of days you want for your vacation. Each node in the linked list represents a specific objects which contains the above set of information.

The program uses the linked list to perform various operations such as add, update, delete, search, and display. The add function creates a new node, fill the node with the information of the object and connects it to the appropriate position in the linked list. The update function searches the linked list for the specific data, update it. The delete function searches the linked list for the specific location linked to the object and deletes it. The search function searches the linked list for the specific location linked to the object and displays the result to the user. The display function displays all the data in the linked list in a clear and easy-to-read format.
The advantage of using a linked list data structure in this program is that it allows for easy manipulation of the data, and it is easy to insert and delete elements from the linked list
.
Additionally, linked lists are dynamic in nature, meaning that the size of the list can change as new elements are added or removed. This makes it a suitable data structure for a program that needs to handle a large amount of data and needs to be able to add, update, delete, search and display the data quickly and efficiently.

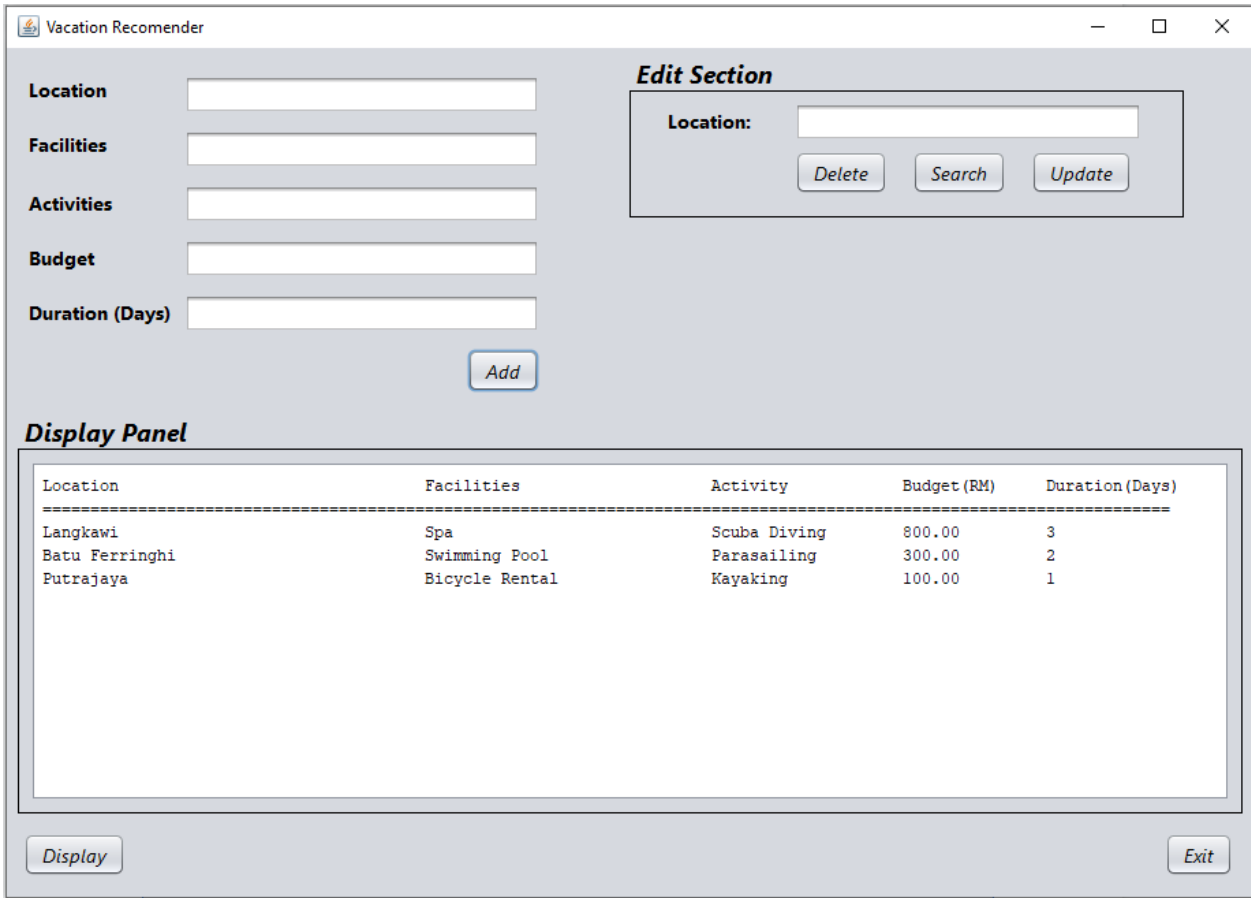
7.0 GUI Interface



The image above is our design for the GUI.

8.0 Sample Outputs

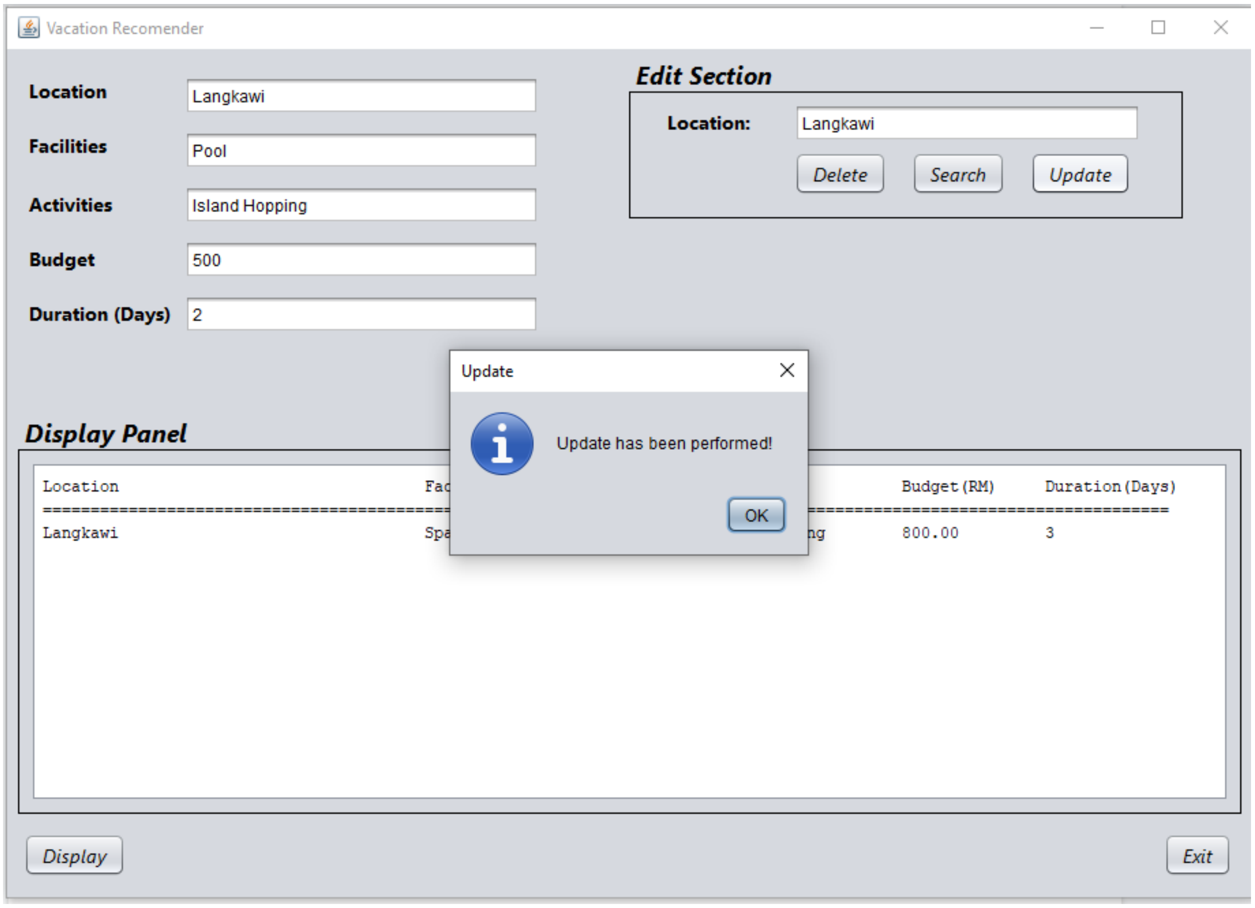
<u>Add Object</u>



```
Location                                Facilities                    Activity           Budget(RM)     Duration(Days)
=======================================================================================================================
Langkawi                                Spa                           Scuba Diving       800.00         3
Batu Ferringhi                          Swimming Pool                 Parasailing        300.00         2
Putrajaya                               Bicycle Rental                Kayaking           100.00         1
```

Three object of vacay have been added into the linked list.

Update Object



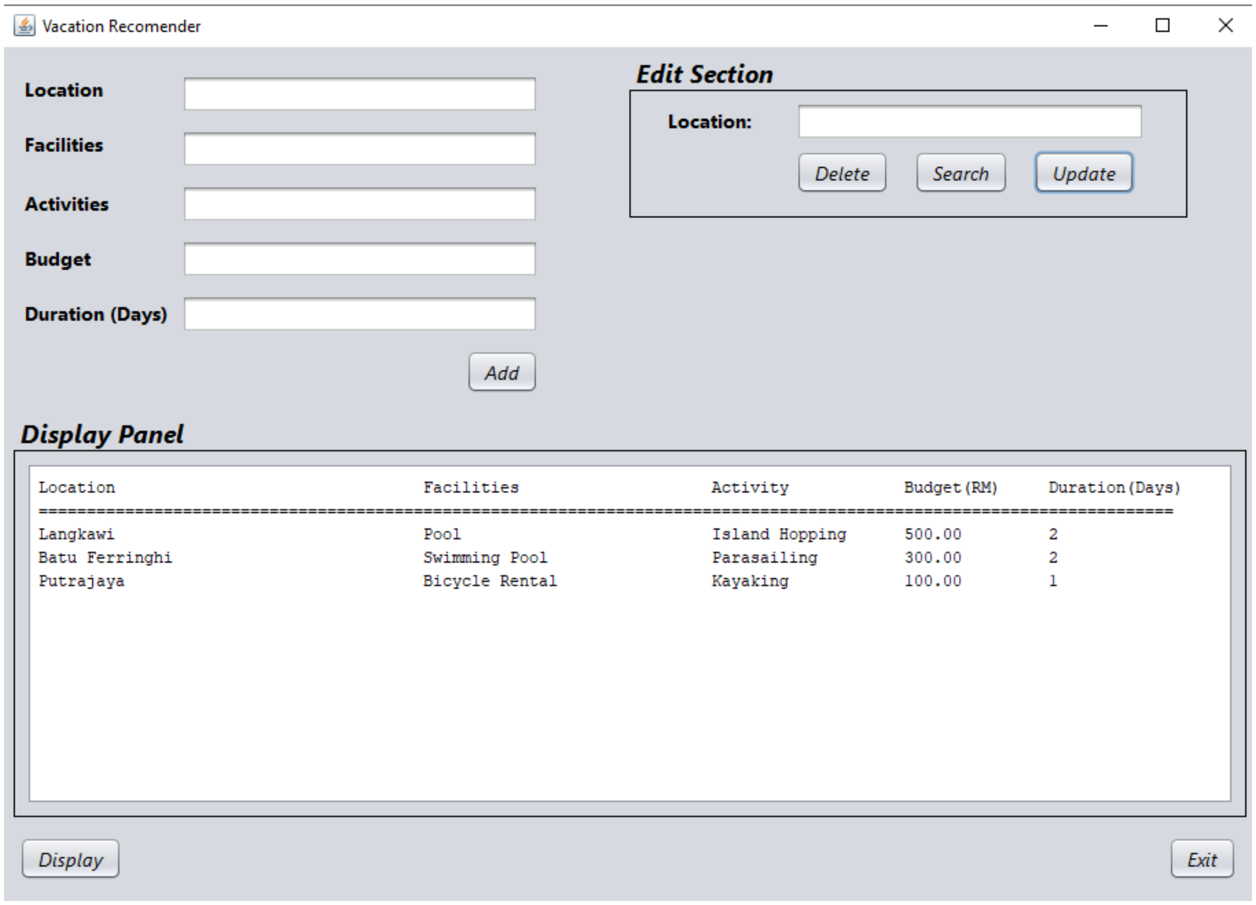
Location	Langkawi
Facilities	Pool
Activities	Island Hopping
Budget	500
Duration (Days)	2

Location: Langkawi

Update has been performed!

```
Location                                       Fac                                        Budget(RM)     Duration(Days)
=====================================================     ======================================
Langkawi                                       Spa                                     ng  800.00         3
```

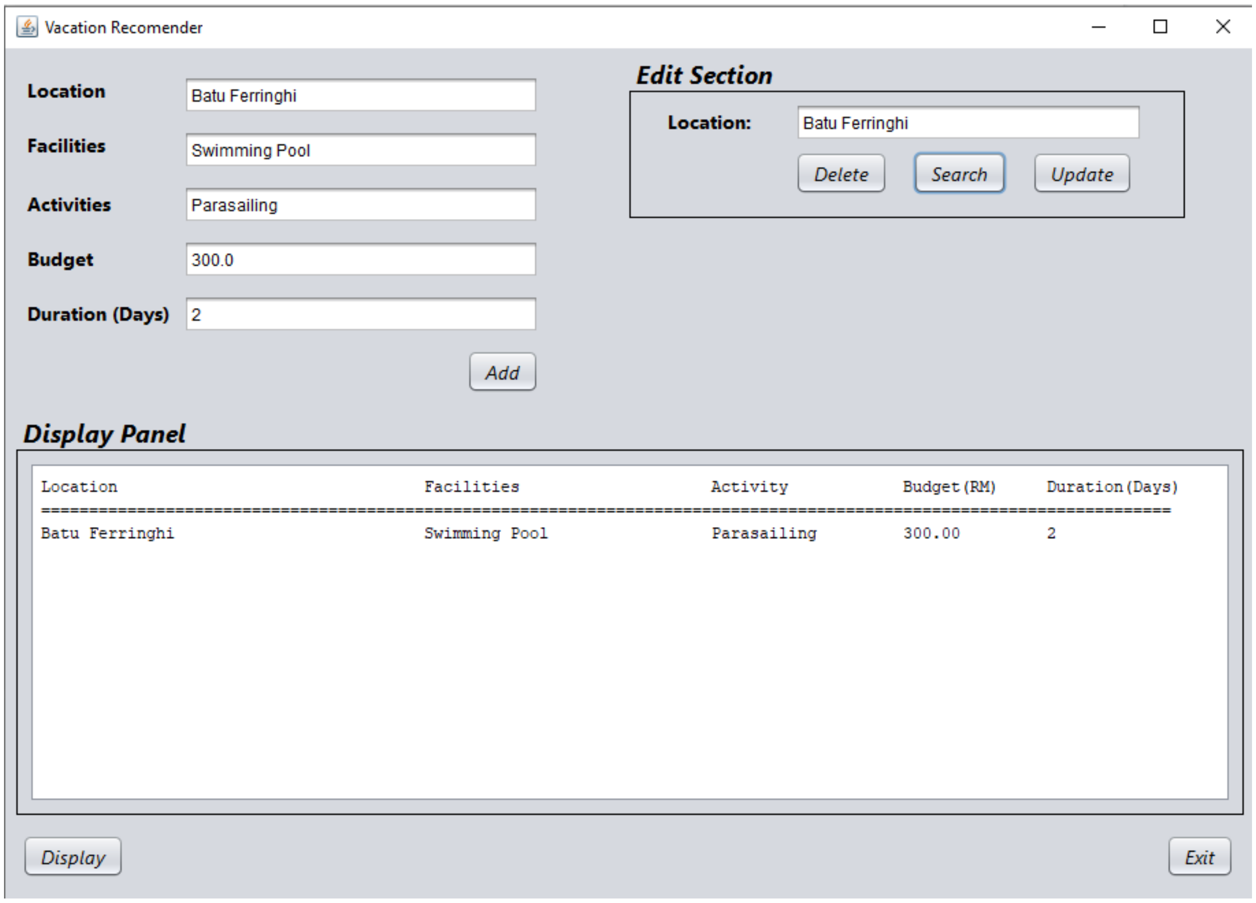
An update is performed for the vacay Langkawi.



```
Location                                        Facilities                          Activity                Budget(RM)      Duration(Days)
==================================================================================================================================
Langkawi                                        Pool                                Island Hopping          500.00          2
Batu Ferringhi                                  Swimming Pool                       Parasailing             300.00          2
Putrajaya                                       Bicycle Rental                      Kayaking                100.00          1
```

The facilities, activity, budget and duration for Langkawi has been updated.

Search Object



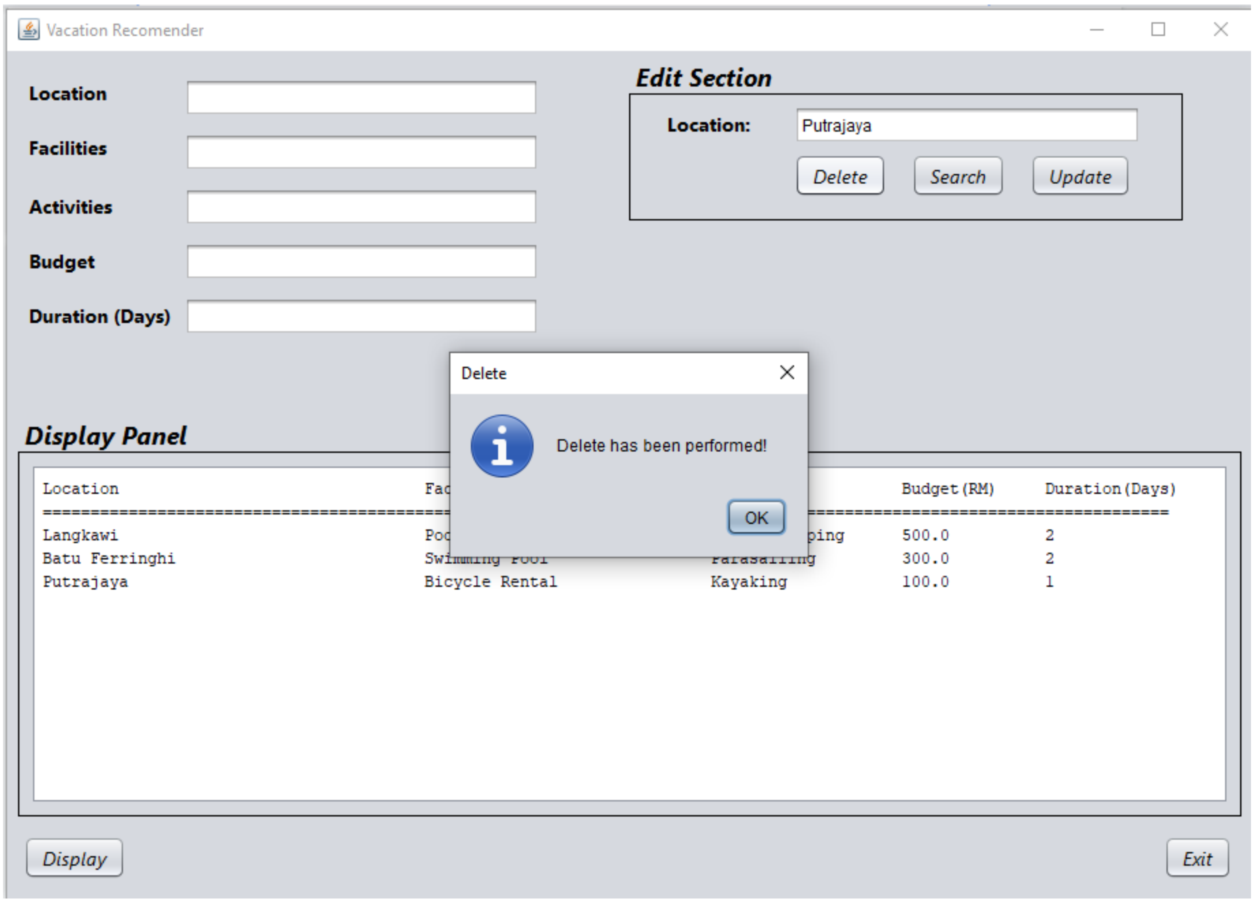
Location: Batu Ferringhi
Facilities: Swimming Pool
Activities: Parasailing
Budget: 300.0
Duration (Days): 2

Edit Section — Location: Batu Ferringhi

```
Location                                Facilities                    Activity             Budget(RM)     Duration(Days)
=======================================================================================================================
Batu Ferringhi                          Swimming Pool                 Parasailing          300.00         2
```

The search was performed on Batu Ferringhi and it is displayed in the display panel.

Delete Object



Vacation Recomender

Location

Facilities

Activities

Budget

Duration (Days)

Edit Section

Location: Putrajaya

Delete *Search* *Update*

Delete

Delete has been performed!

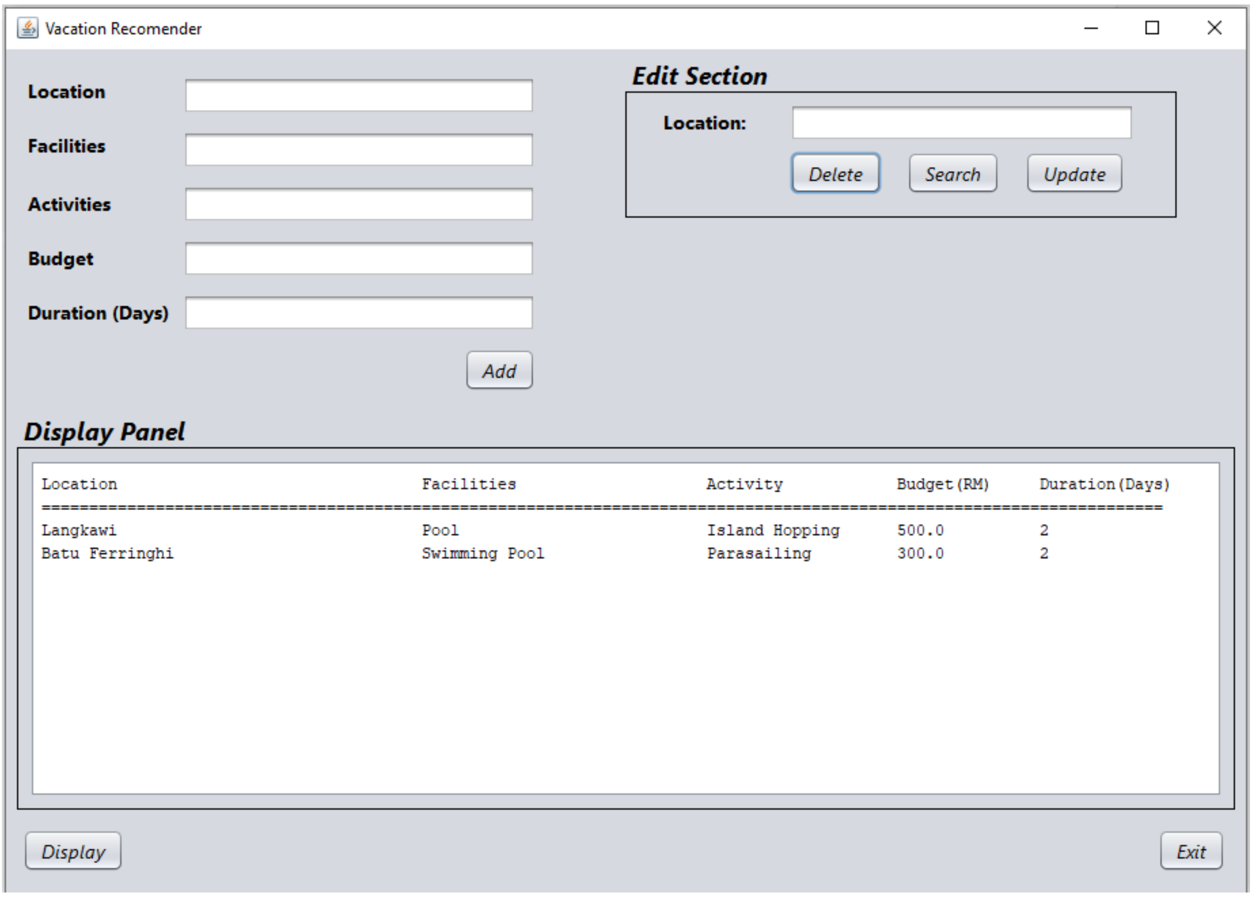
OK

Display Panel

```
Location                                Fac                            Budget(RM)     Duration(Days)
========================================================================================================
Langkawi                                Poc                        ping    500.0          2
Batu Ferringhi                          Swimming Pool            Parasailing   300.0          2
Putrajaya                               Bicycle Rental            Kayaking      100.0          1
```

Display *Exit*

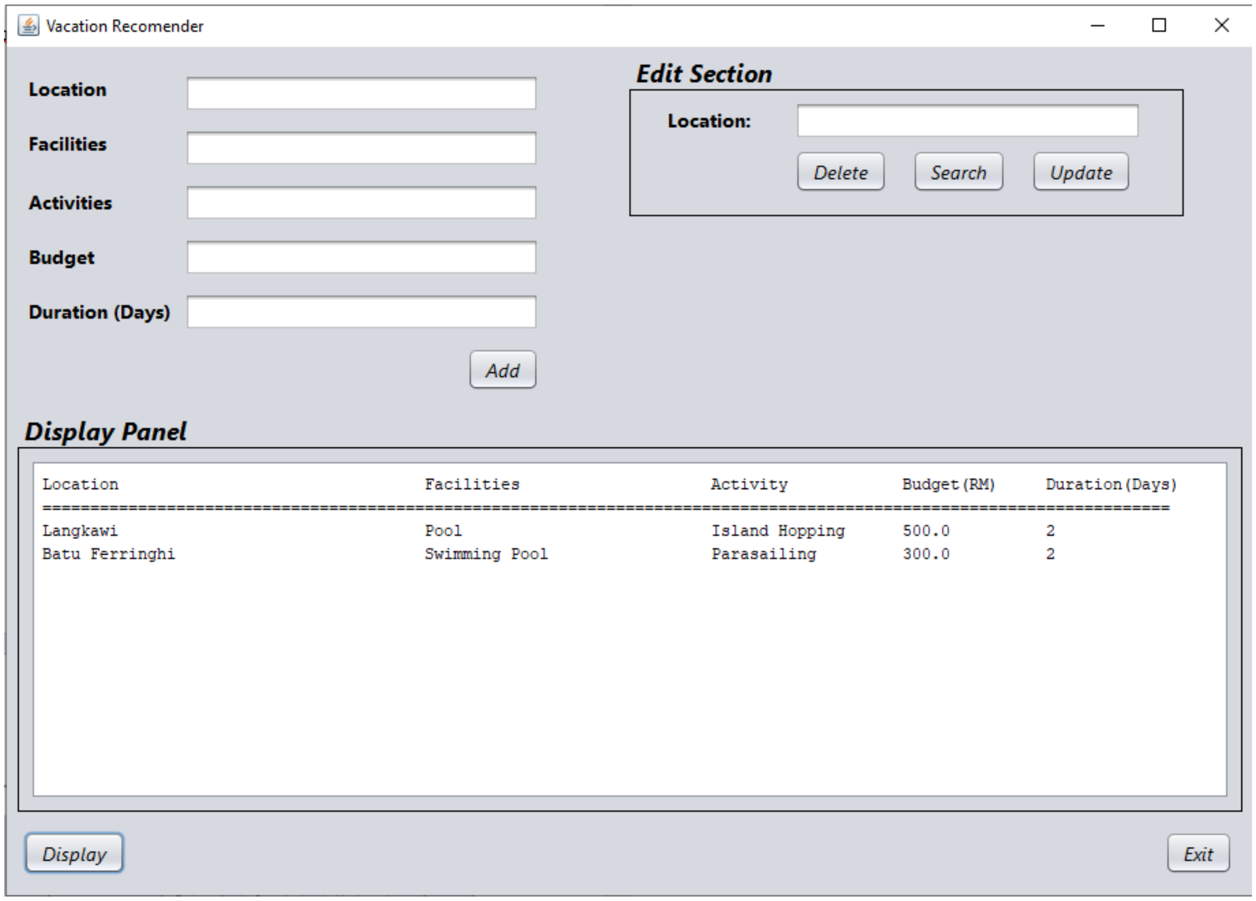
The delete was performed on Putrajaya.



```
Location                                Facilities                    Activity            Budget(RM)     Duration(Days)
=====================================================================================================================
Langkawi                                Pool                          Island Hopping      500.0          2
Batu Ferringhi                          Swimming Pool                 Parasailing         300.0          2
```

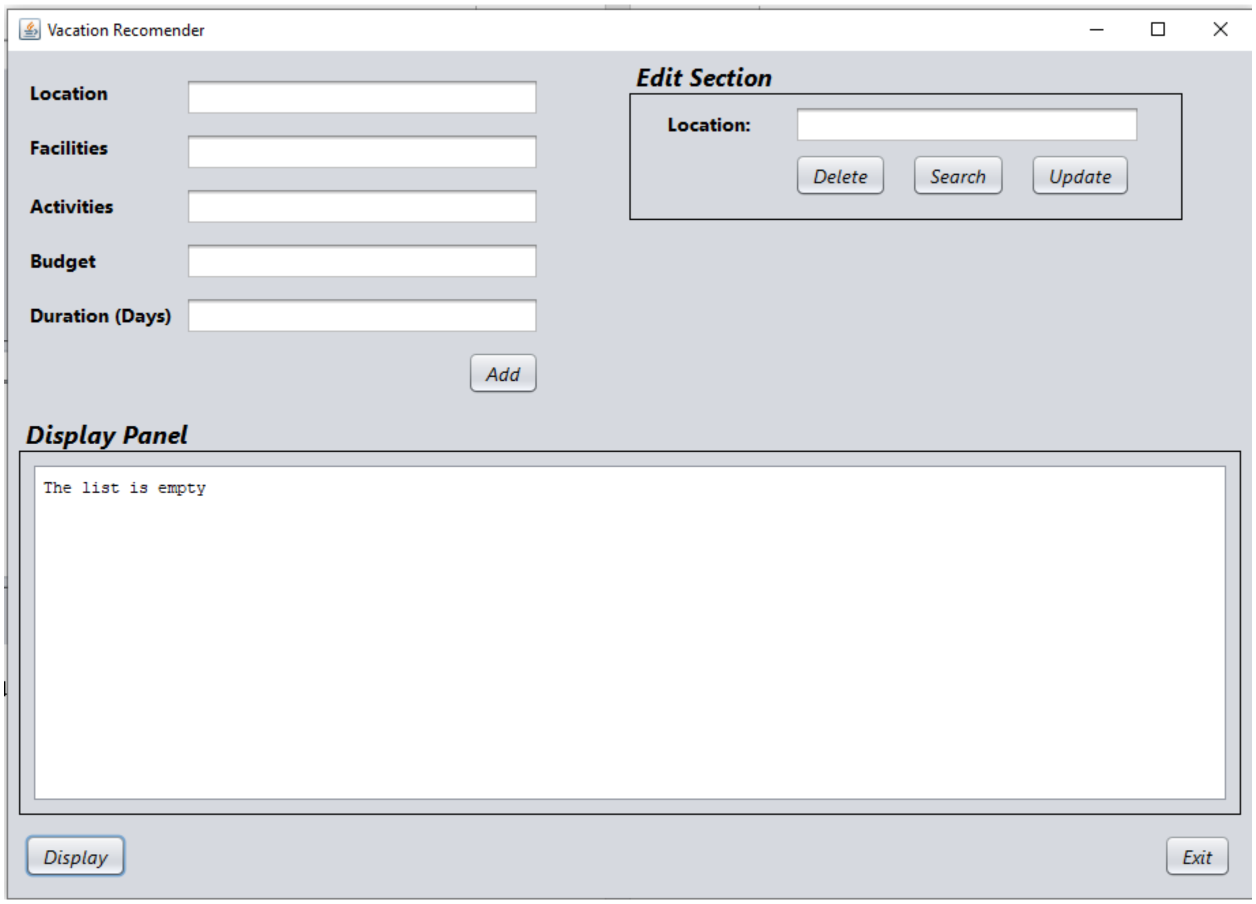
After the delete on Putrajaya was performed, there were only 2 remaining objects in the linked list.

Display Object



```
Location                                    Facilities                       Activity              Budget(RM)       Duration(Days)
===================================================================================================================================
Langkawi                                    Pool                             Island Hopping        500.0            2
Batu Ferringhi                              Swimming Pool                    Parasailing           300.0            2
```

The Display button will list out all the object in a clear and easy-to-read format.



```
The list is empty
```

When the list is empty while the display button is performed, the display panel will show “The list is empty”.