Criterion C

Introduction

In making this program, Netbeans was used as the development environment, due to it's easy to use Swing GUI designer. It allowed for the creation of the Beach Picker program, which gives summaries of information about potential beaches/resorts for potential family trips.

All Techniques

- parameter passing
- for loop
- nested loops
- Arrays / arraylists
- method returning a value
- making an array of objects
- Encapsulation
- Overloading constructors
- sorting (bubble sort etc.), and in particular sorting an array of objects based on one key attribute
- searching (linear search, binary search…)
- Error handling (for example catching a divide by 0 error, or a null pointer while using an array of object…)
- GUI tabs
- GUI popup dialog
- Use of a flag value (such as -999, or "not set yet")
- Simple and compound selection
- use of some other specialized library you imported

Structure of the Program

For this program, the main class is GUI.java. It handles the displaying of the graphics, and defines the functions for elements such as buttons and tabs. This program also uses org.json, which is a library created for handling, manipulating, and generating JSON data in Java. When the program's main class interacts with web APIs, the data returned is JSON, and therefore the library is used to parse and display the data returned from said API. The data is then taken from the JSONArray and each entry is read as a JSONObject, and finally the data is moved to a Place object, which is held in an ArrayList for displaying in the table.

I chose to split the program into different classes so that I could take advantage of the benefits of OOP. Specifically, I was able to create a result object, which could hold all relevant information for displaying later in the table view. Having the Place object meant that a method could be called from each one, and the information needed for the table would be returned in an expected fashion, allowing for easier coding and leaving less room for errors in code. Also, the information stored in each Place object is read-only, meaning the data cannot be changed once created. This is purposeful, as accidental edits may lead to further errors in the program. Through OOP practices such as encapsulation, I was able to restrict the editing of the raw data in the Result object instances.

Data Structures Used

In this program, the main data structures were JSONArray, JSONObject, and Place. The JSON structures were used to parse data from the web APIs, and came from an external library called org.json. The Place object was created to hold each result, in a way that was beneficial for displaying the data in the results table.

Main Unique Algorithms

The main algorithms in this program are the interactions with the web APIs. Using JSON parsing and specific queries via URLs, user input is sent online and relevant results are returned to the program for processing and parsing. Results are held in RAM as arrays and arraylists of Place objects, which themselves hold the names and locations of each entry, and can be sorted using the selection sort algorithm.

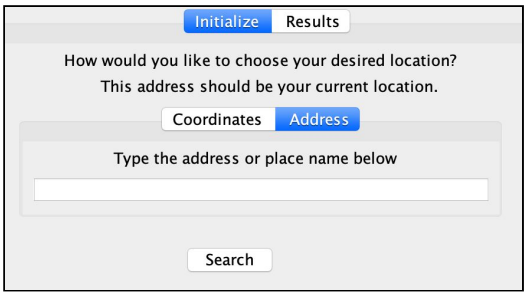
In essence, the web API interactions happens as follows: (for example, searching coordinates)

Pseudo Code:

```
function queryAPI(latitude, longitude) {
        String url = baseAPIURL;
        url +=  userInput;
        url += otherFlags; // tells the API to output JSON
        urlObject = openURL(url);
        stream = urlObject.getStream();
        String output = stream.getData();
        convertToJSON(output);
}
```

Actual Code:

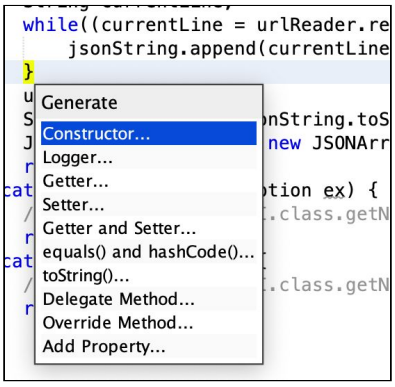
User Interface/GUI Work



For the user interface, the Java Swing API was used for a *graphical* user interface. For the main portions of user input, text boxes were used, as they offer direct input from the user. Initially combo boxes were used for sorting, but were eventually taken out due to the sorting code having errors. Additionally, buttons and tabbed panes were used. Buttons are ubiquitous, and tabbed panes offer an elegant way to separate the program into sections.

Software Tools Used

When creating this program, Netbeans was used as the development environment. Netbeans is an IDE used by hobbyists and professionals alike, and is a useful tool in creating graphical components for Java programs. When typing code, it also offers many shortcuts and macros to easily and quickly create boilerplate code, which would otherwise waste time in the creation of the program. Finally, when writing code, it provides useful hints and warnings pertaining to each line of code, helping the coder to understand their program and fix many potential errors.



The Code Generation Dropdown Menu in Netbeans

Aside from Netbeans, Google Drive and Google Docs were used for the planning and writing portions of this IA. Most of the documents written for this project were easily organized and maintained due to the use of this software. Additionally, using Drive meant that in the case of data loss or corruption on the main programming device (which held both the program and writing data), complete backups of the text would be available without delay.

Word Count: 824