Appendix - Computer Science IA

Index:

Appendix 1:

First Conversation with the client:

Me: Hello, I hope you are doing well. I spoke with one of your employees and he told me about the vehicle parking space company you managed. He told me you are doing everything very manual and that It will be very handy to digitize many of the processes. Tell me how can I help you?

Client: Hello, *My name*, it is great speaking with you, I'm keen to work with you. Thankfully the business is going great and we are getting a big amount of new clients which are both interested in renting their parking space and many others who need a space to park their cars. We now have operations in two cities (Cali and Bogota) and our traditional and manual way of recording data is just getting very time consuming and disorganized. We've already had some issues with renting some parking spaces which had already been rented before and keeping track of everything is getting too hard.
I think a program that allows me to easily add the parking spots to a sort of file where I could keep track of them will be amazing. As well the program should have an option to rent out the parking spots so that every time I rent one spot out it is deleted from the file and I will keep track of all the available spots in an organized way.

Me: Sounds good, thanks for your confidence and I will do my best to make an amazing and very functional project. Will you need any additional functionalities?

Client: Well maybe a place where I can just look at the available parking spots. Just to view them and no need to add or rent out. Those three functionalities will be the main things I will need if possible.
And, something which calculates the price I should charge the customer depending on the number of days they rent the parking place. I will give you the prices later on.

Me: Okay, I'm thinking of creating the main menu, where you can access those 3 functionalities and carry out any procedure you need. As well I will create a method or class which calculates the prices you need.
I'll come back and show you a draft soon.

Client: Sounds perfect. Thank you very much. Take care

Appendix 2:

Coding Complexity requirements:

Standard Level

The rules are slightly different for SL. The rule of thumb used to be that a project had to **have at least 10** of the **following 15 aspects** to be counted as 'sufficiently complicated'. They are

1. Arrays
2. User-defined objects
3. Objects as data records
4. Simple selection (if–else)
5. Complex selection (nested if, if with multiple conditions or switch)
6. Loops
7. Nested loops
8. User-defined methods
9. User-defined methods with parameters (the parameters have to be useful and used within the method body)
10. User-defined methods with appropriate return values (primitives or objects)
11. Sorting
12. Searching
13. File i/o
14. Use of additional libraries (such as utilities and graphical libraries not included in appendix 2 Java Examination Tool Subsets)
15. Use of sentinels or flags

Appendix 3:

Second meeting with the client:

Me: Hello, How are you? What do you think about the planning of the code?

Client: Hey there! I loved it, but I forgot to specify something regarding the price calculator. I need this to be calculated only when I rent out a parking spot so that I can have an organized procedure of the price I need to charge when someone rents out a spot. As well, the weekday's price is COP 15,000 for a day, the weekend price is COP 10,000 for a day and there is a monthly promotion plan which will cost COP 250,000 for the full month of parking. So this functionality should calculate the price depending on the day they are parking (weekday or weekends) and also depending on the number of days the client is parking.
I hope this won't be too much of an issue.

Me: Not at all, I can easily make that work, don't worry. We will keep in touch.

Client: Great, thank you very much.

Appendix 4:

Third meeting with the client:

Me: Hello, how are you doing? How is business?

Client: Hello, everything is going well, thankfully the business is growing. We are excited to start using your program, it will be of great help.

Me: Well I was going to talk to you about just that. The main idea is to have a program where you will be able to access three different main options, using the main menu. The option will be: Adding a spot to your files, renting out spots, and viewing the spots available in your files. As well, we will have an exit option, which you can choose whenever you wish to terminate or close the program. How do those ideas sound for you?

Client: Sounds great. Just remember about the price calculator for when I rent out spots. As well, keeping in mind organization is essential, in order to avoid mixing any parking spot addresses between days or cities.

Me: Yes, I forgot to mention that, sorry. We have all of these functionalities arranged and ready to go. I will let you know how our progress goes. Thank you very much.

Client: Thank you. Any questions please let me know.

Appendix 5:

Fourth meeting with the client:

Client: Hello, How are you doing? I saw the plan you sent me yesterday, it's that in the UML diagram the final idea for the program?

Me: Yes, that's the final idea, after your approval, I will begin with the coding portion. Any suggestions?

Client: Well my only additional comment will be if it's possible, to add a space before adding the parking spots to my files, where I'm able to input my client's details, such as email, phone number, name, etc.

Me: Great. I'll do that without any issue. Sounds good to me.

Client: Thank you very much, I really appreciate it.

Appendix 6:

Fifth meeting with the client:

Me: Hello, how are you? Today I will be showing you the first draft of the code.

Client: Sounds good, I'm keen to see how this is progressing.

*** Showing the program by sharing the screen.

Me: So, what do you think?

Client: I like it, it's looking good. But I notice that after you close the program, the data is erased and the addresses disappear. It's this normal.

Me: Yes, I'm missing the java serialization code, which will enable us to create files and store all of your addresses. This will eventually save all the data and keep it updated even if we close the program. I will go ahead and code this and I will show you as soon as possible.

Client: Okay, I don't know too much about this but it sounds good for me. See you soon.

Appendix 7:

Sixth meeting with the client:

Me: hello there, how are you doing today?

Client: Hi, great thanks. I'm keen to see how this program is doing.

Me: I'll show you right now.

*** Showing the program by sharing the screen.

Client: Wow, this is amazing, I really like this, it will definitely become a great tool for our business. I just noticed some bad error handling at some points, the screen turns black or the program just stopped.

Me: Thanks, I'm happy you like it. Yes, I will go to a fixed does error handling functions and shortly I will show you the finalized version.

Client: great thanks. Have a great rest of the afternoon.

Appendix 8:

Seventh meeting with the client:

Me: Hello, ready to see the final version?

Client: Oh yeah, I'm keen.

*** Showing the program by sharing the screen.

Client: I love this, it is totally what we needed and it actually overpast my expectations. This will definitely become a great and very useful tool for my business. I really like its functionalities and how easy it is to use. It's perfect. Thank you very much.

Me: I'm very glad you like it. It is my pleasure. I will now send this program to you, so that you can use it for a week or so and then you could give me some feedback. This will work as a Beta Testing.

Client: Sound great. I'm keen to begin using this program.

*** One week later:

Client: Hi there, the program has been working great, I just have some suggestions.

Me: Hello! Nice, yes please, what could I improve?

Client: My only suggestions are actually related to formatting and organization. It will be nice if the main menu could be displayed in a more visually appealing way as well as when the addresses are printed out. On the other hand, every time I try to input an address such as "road 34" in the Cali during weekdays option, I get an error message. I don't understand why this happens.

Me: Yes, I agree with you and I will work in order to make that mian menu more appealing. On the other hand, I guess I forgot to change that Arraylist into a String rather than a double, I think that might be the cause of the error. But no worries, I will fix things as soon as possible.

Client: Thank you very much. Other than that, I'm actually very impressed with this program, it has worked perfectly and has helped us a lot. I really appreciate it.

Me: I'm glad to hear that. See you soon.

Appendix 9:

Eight and final meeting with the client:

Me: Hello, how are you? How has the program been working?

Client: Hi! It has been amazing to use this technological tool in our business. It has definitely allowed us to become more organized and to improve our productivity.

Me: I'm happy to hear this. I just wanted to thank you for letting me use your company to develop this project and for believing in me to develop your solution. It's been an honor to work with you. As well I wanted to know what you liked and any recommendations or extensions I could do to the program for further improvements.

Client: The honor is mine. Thank you very much for your compromise and for developing such an amazing product.
I really like the main menu feature, being able to easily select from the options and the program performing the according tasks. As well, the fact that all addresses were safely stored and updated in separate files was amazing, I was able to keep track of them and even send these files via email or other mediums, whenever I needed to do certaina analytics or statistics for the business. The program was very practical and easy to use; I loved it! You are great at doing this. I really appreciate it.

For further additions or improvements, I will only have three main points:

1. It will be great if the program could become more visually appealing, I don't know, maybe having buttons and a text box for the main menu and these types of elements. I don't know the technical name but maybe improving the visuals of the interface and the presentation of the program.
2. Having a password in order to protect all my information will also be a good idea. As I'm storing all the addresses and business data in this program, I just want to avoid any accidents from occurring. I guess a password will be a good idea.
3. Maybe creating a more complex way of staring at data. It might be a database or something similar, which allows me to store not only the addresses but also my client's information. It will be of great use having a client's database, where I can store and access all of my client's details.

Other than that, you did a great job. I really admire you and I hope we get to work together again in the future.

Me: Thank you very much. I totally agree with all of your recommendations and feedback. I will do my best to implement this in your program. I hope to see you in the near future and thank you very much for enabling me to take part in this process. It was an honor.

Client: Thank you! You did great. Take care. See you soon.

Appendix 10:

Pseudocode compiler:

Mulkey, D. (2012). IB Comp - Pseudocode practice tool. Retrieved 16 September 2020, from http://ibcomp.fis.edu/pseudocode/pcode.html

Appendix 11:

APA References:

Criteria A references:

SnapLogic (2020) - Python vs. Java Performance. (2020). Retrieved 27 September 2020, from https://www.snaplogic.com/glossary/python-vs-java-performance#:~:text=Java%20is%20generally%20faster%20and,in%20fewer%20lines%20of%20code.

Stack Overflow. (2020). Stack Overflow - Where Developers Learn, Share, & Build Careers. Retrieved 5 November 2020, from https://stackoverflow.com/

Criteria B references:

GeeksforGeeks. (2020). Types of Software Testing - GeeksforGeeks. Retrieved 29 September 2020, from https://www.geeksforgeeks.org/types-software-testing/

Criteria C references:

GeeksforGeeks. (2019) Decision Making in Java (if, if-else, switch, break, continue, jump) - GeeksforGeeks. Retrieved 24 September 2020, from https://www.geeksforgeeks.org/decision-making-javaif-else-switch-break-continue-jump/

GeeksforGeeks. (2019). Serialization and Deserialization in Java with Example - GeeksforGeeks. Retrieved 21 October 2020, from https://www.geeksforgeeks.org/serialization-in-java/

GeeksforGeeks. (2020). Loops in Java - GeeksforGeeks. Retrieved 25 September 2020, from https://www.geeksforgeeks.org/loops-in-java/

Paste of Code. (2020). Retrieved 25 September 2020, from https://paste.ofcode.org/aMhcmGEqwAAjFfV8ctCHWh

Singh, C. (2019). ArrayList in java with example programs - Collections Framework. Retrieved 24 September 2020, from https://beginnersbook.com/2013/12/java-arraylist/

Singh, C. (2020). Search Algorithms in Java. Retrieved 28 September 2020, from https://stackabuse.com/search-algorithms-in-java/#:~:text=Searching%20is%20one%20of%20the,application%20for%20the%20end%2Duser.

Study.com. (2020). Retrieved 25 September 2020, from https://study.com/academy/lesson/methods-in-java-definition-example.html

tutorialspoint. (2020). Java - Files and I/O - Tutorialspoint. Retrieved 28 September 2020, from https://www.tutorialspoint.com/java/java_files_io.htm

VertexAcademy. (2016). What is the Java Library? • Vertex Academy. Retrieved 28 September 2020, from https://vertex-academy.com/tutorials/en/what-is-java-library/

WeboPedia. (2020). What is sentinel value? Webopedia Definition. Retrieved 14 October 2020, from https://www.webopedia.com/TERM/S/sentinel_value.html

Appendix 12:

Final Code

Main Class:

```
1.  package com.company;
2.
3.  //Imported libraries
4.  import java.util.Scanner;
5.  import java.util.ArrayList;
6.  import java.io.FileOutputStream;
7.  import java.io.IOException;
8.  import java.io.ObjectOutputStream;
9.  import java.io.FileInputStream;
10. import java.io.ObjectInputStream;
11. import java.io.IOException;
12.
13. public class Main {
14.
15.     // Array lists to store parking spot addresses
16.
17.     static ArrayList<String> CaliWeekDay = new ArrayList<>();
18.
19.     static ArrayList<String> CaliWeekEnds = new ArrayList<>();
20.
21.
22.     static ArrayList<String> BogotaWeekDay = new ArrayList<>();
23.
24.
25.     static ArrayList<String> BogotaWeekEnds = new ArrayList<>();
26.
27.     public static void main(String[] args) throws Exception {
28.         // Scanner - File.io
29.          Scanner keyboard = new Scanner(System.in);
30.
31.
32.          int whichday;
33.
34.          int exit = 0;
35.
36. // Deserealize for the Arraylist BogotaWeekDay
37.          try
38.          {
39.              FileInputStream fis = new FileInputStream("Bogota Week Days");
40.              ObjectInputStream ois = new ObjectInputStream(fis);
41.
42.              BogotaWeekDay = (ArrayList) ois.readObject();
43.
44.
45.              ois.close();
46.              fis.close();
47.          }
```

Java imported libraries

ArrayLists to store each of the addresses

Scanner - File i/o

```
48.         catch (IOException ioe)
49.         {
50.             ioe.printStackTrace();
51.             return;
52.         }
53.         catch (ClassNotFoundException c)
54.         {
55.             System.out.println("Class not found");
56.             c.printStackTrace();
57.             return;
58.         }
59.
60.
61.
62.
63.         // Deserealize for the Arraylist BogotaWeekEnds
64.         try
65.         {
66.             FileInputStream fis = new FileInputStream("Bogota Week Ends");
67.             ObjectInputStream ois = new ObjectInputStream(fis);
68.
69.             BogotaWeekEnds = (ArrayList) ois.readObject();
70.
71.
72.             ois.close();
73.             fis.close();
74.         }
75.         catch (IOException ioe)
76.         {
77.             ioe.printStackTrace();
78.             return;
79.         }
80.         catch (ClassNotFoundException c)
81.         {
82.             System.out.println("Class not found");
83.             c.printStackTrace();
84.             return;
85.         }
86.
87.
88.
89.         // Deserialize for the Arraylist CaliWeekDay
90.         try
91.         {
92.             FileInputStream fis = new FileInputStream("Cali Week
    Days");
93.             ObjectInputStream ois = new ObjectInputStream(fis);
94.
95.             CaliWeekDay = (ArrayList) ois.readObject();
96.
97.
98.             ois.close();
99.             fis.close();
100.           }
```

```
101.            catch (IOException ioe)
102.            {
103.                ioe.printStackTrace();
104.                return;
105.            }
106.            catch (ClassNotFoundException c)
107.            {
108.                System.out.println("Class not found");
109.                c.printStackTrace();
110.                return;
111.            }
112.
113.
114.
115.
116.            // Deserealize for the Arraylist CaliWeekEnds
117.            try
118.            {
119.                FileInputStream fis = new FileInputStream("Cali Week Ends");
120.                ObjectInputStream ois = new ObjectInputStream(fis);
121.
122.                CaliWeekEnds = (ArrayList) ois.readObject();
123.
124.
125.                ois.close();
126.                fis.close();
127.            }
128.            catch (IOException ioe)
129.            {
130.                ioe.printStackTrace();
131.                return;
132.            }
133.            catch (ClassNotFoundException c)
134.            {
135.                System.out.println("Class not found");
136.                c.printStackTrace();
137.                return;
138.            }
139.
140.
141.
142.            // Start of the program - Welcoming message
143.
144.            System.out.println("\n Hello! What is your name?");
145.            String name = keyboard.next();
146.            System.out.println("\n Hello! " + name + ", welcome to RentP! We
    will help" +
147.                    " you to managed your parking spot company!");
148.
149.
150.            // while loop                    } While(true) loop
151.
152.            while (true) {
153.
```

```
154.                // Main menu
155.
156.                System.out.println("\n =_=_=_=_=_=_=_=_=_=_=_=_=_=_=_=_=_=_=_=_=_=_=");
157.                System.out.println("\n \t MAIN MENU:");
158.                System.out.println("\n 1) Renting out Parking Spot \n 2)
    Adding Parking Spot to Data base " +
159.                        "\n 3) View Parking Spots \n 4) EXIT");
160.                System.out.println("\n Please choose the number of your
    option!");
161.                int option = keyboard.nextInt();
162.
163.
164.                // Option 1 - "Renting out parking spot"
165.
166.                if (option == 1) {
167.
168.                    System.out.println("\n Hello! In which city will your
    client like to park? Bogota or Cali?");
169.                    String cityChoice = keyboard.next();
170.
171.                    Looking_Spot lookingForSpot = new Looking_Spot();
172.
173.
174.                    if (cityChoice.equals("Bogota") ||
    cityChoice.equals("bogota")) {
175.
176.                        System.out.println();
177.                        System.out.println("\n Which day will your client want
    to rent the spot?");
178.                        System.out.println("\n 1) Weekdays \n 2) Weekends");
179.                        whichday = keyboard.nextInt();
180.
181.                        if (whichday == 1) {
182.
    lookingForSpot.availableSpotsBogotaWeekDay(BogotaWeekDay);
183.                        }
184.                        else if (whichday == 2) {
185.
186.
    lookingForSpot.availableSpotsBogotaWeekEnds(BogotaWeekEnds);
187.                        }
188.
189.                        else {
190.                            System.out.println("ERROR! Please input a correct
    day. ");
191.                        }
192.                      // Sentinels/Flags - Modifying loop condition
193.                        exit++;
194.                    } else if (cityChoice.equals("Cali") ||
    cityChoice.equals("cali")) {
195.
196.                        System.out.println();
197.                        System.out.println("\n Which day will your client want
    to rent the spot?");
```

User-Defined Objects

Complex Selection
- Nested if, else if

```
198.                         System.out.println("\n 1) Weekdays \n 2) Weekends");
199.                         whichday = keyboard.nextInt();
200.                             // if statement
201.                         if (whichday == 1) {
202.
    lookingForSpot.availableSpotsCaliWeekDay(CaliWeekDay);
203.
204.                         }
205.                         else if (whichday == 2) {
206.
    lookingForSpot.availableSpotsCaliWeekEnds(CaliWeekEnds);
207.                         }
208.
209.                         else {
210.                             System.out.println("ERROR! Please input a correct
    day. ");
211.                         }
212.
213.
214.                         exit++;
215.                     }
216.                     else {
217.                         System.out.println("\n ERROR! Please imput a correct
    city. Mind your spelling");
218.                     }
219.
220.
221.                 }
222.
223.
224.                 // Option 2 - "Adding Spot"
225.
226.                 else if (option == 2) {
227.
228.                     System.out.println("\n Adding Parking Spot to your data
    base...");
229.
230.                     System.out.println("\n \n \t YOUR CLIENT'S CONTACT
    INFORMATION:");
231.                     System.out.println("\n First Name: ");
232.                     String nameclient = keyboard.next();
233.
234.                     System.out.println("\n Last Name: ");
235.                     String lastName = keyboard.next();
236.
237.                     System.out.println("\n Email address: ");
238.                     String email = keyboard.next();
239.
240.                     System.out.println("\n Phone Number: ");
241.                     String phoneNumber = keyboard.next();
242.
243.
244.                     System.out.println("\n \t PARKING SPOT INFORMATION: ");
245.
```

```
246.                     System.out.println("\n City of your clients spot. Bogota
    or Cali?: ");
247.                     String City = keyboard.next();
248.
249.                      // User-defined object
250.                     Renting_Spot RentMySpot = new Renting_Spot();
251.
252.
253.                     if (City.equals("Bogota") || City.equals("bogota"))
    {
254.
255.                         System.out.println();
256.                         System.out.println("\n Which days will your
    client be renting the spot?");
257.                         System.out.println("\n 1) Weekdays \n 2) Weekends ");
258.                         whichday = keyboard.nextInt();
259.
260.                         if (whichday == 1) {
261.                             RentMySpot.addSpotBogotaWeekDay(BogotaWeekDay);
262.                         } else if (whichday == 2) {
263.                             RentMySpot.addSpotBogotaWeekEnds(BogotaWeekEnds);
264.                         }
265.
266.                         else {
267.                             System.out.println("ERROR! Please input a correct
    day. ");
268.                         }
269.
270.
271.                         System.out.println();
272.                         System.out.println("\n The parking spot of your client
    " + nameclient + " " + lastName + " has been " +
273.                                 "added to your data base and it will be
    available to rent it out");
274.
275.
276.                     } else if (City.equals("Cali") || City.equals("cali")) {
277.
278.                         System.out.println();
279.                         System.out.println("\n  Which days will your client be
    renting the spot?");
280.                         System.out.println("\n 1) Weekdays \n 2) Weekends ");
281.                         whichday = keyboard.nextInt();
282.
283.                         if (whichday == 1) {
284.                             RentMySpot.addSpotCaliWeekDay(CaliWeekDay);
285.                         } else if (whichday == 2) {
286.                             RentMySpot.addSpotCaliWeekEnds(CaliWeekEnds);
287.                         }
288.
289.                         else {
290.                             System.out.println("ERROR! Please input a correct
    day. ");
291.                         }
```

Simple selection - if, else

```
292.
293.
294.                     System.out.println();
295.                     System.out.println("\n The parking spot of your client
    " + nameclient + " " + lastName + " has been " +
296.                                 "added to your data base and it will be
    available to rent it out");
297.
298.
299.                 } else {
300.
301.                     System.out.println("\n ERROR! Please input a correct
    city. Mind your spelling");
302.                 }
303.
304.
305.             }
306.
307.             // Option 3 - View available parking spots in data base
308.
309.             else if (option == 3) {
310.
311.                 ViewParkingSpots();
312.             }
313.
314.
315.             // Option 4 - "EXIT"
316.
317.             else if (option == 4) {
318.
319.                 // Serealize Arralist BogotaWeekDay
320.                 try {
321.                     FileOutputStream fos = new FileOutputStream("Bogota
    Week Days");
322.                     ObjectOutputStream oos = new ObjectOutputStream(fos);
323.                     oos.writeObject(BogotaWeekDay);
324.                     oos.close();
325.                     fos.close();
326.                 } catch (IOException ioe) {
327.                     ioe.printStackTrace();
328.                 }
329.
330.
331.
332.                 // Serealize Arralist BogotaWeekEnds
333.                 try {
334.                     FileOutputStream fos = new FileOutputStream("Bogota
    Week Ends");
335.                     ObjectOutputStream oos = new
    ObjectOutputStream(fos);
336.                     oos.writeObject(BogotaWeekEnds);
337.                     oos.close();
338.                     fos.close();
339.                 } catch (IOException ioe) {
```

Serialization

```
340.                     ioe.printStackTrace();
341.                 }
342.
343.
344.                 // Serealize Arralist CaliWeekDay
345.                 try {
346.                     FileOutputStream fos = new FileOutputStream("Cali Week
    Days");
347.                     ObjectOutputStream oos = new ObjectOutputStream(fos);
348.                     oos.writeObject(CaliWeekDay);
349.                     oos.close();
350.                     fos.close();
351.                 } catch (IOException ioe) {
352.                     ioe.printStackTrace();
353.                 }
354.
355.
356.                 // Serealize Arralist CaliWeekEnds
357.                 try {
358.                     FileOutputStream fos = new FileOutputStream("Cali Week
    Ends");
359.                     ObjectOutputStream oos = new ObjectOutputStream(fos);
360.                     oos.writeObject(CaliWeekEnds);
361.                     oos.close();
362.                     fos.close();
363.                 } catch (IOException ioe) {
364.                     ioe.printStackTrace();
365.                 }
366.
367.
368.                 System.out.println("\n See you soon " + name + "!");
369.
370.                 break;
371.
372.
373.             } else {
374.                                                 System.out.println("\n
    ERROR! Please input a correct option. Between 1 and 4");
375.             }
376.
377.
378.         }
379.
380.
381.         // Viewing available parking spots method
382.     }
383.
384.     private static void ViewParkingSpots() {
385.
386.         System.out.println("\n These are the available parking spots on
    your data base:");
387.
388.         System.out.println("\n Bogota During Week Days: ");
389.
```

```
390.            System.out.println("\n \t " + BogotaWeekDay);
391.
392.            System.out.println("\n Bogota During Week Ends: ");
393.
394.            System.out.println("\n \t " + BogotaWeekEnds);
395.
396.            System.out.println("\n Cali During Week Days: ");
397.
398.            System.out.println("\n \t " + CaliWeekDay);
399.
400.            System.out.println("\n Cali During Week Ends: ");
401.
402.            System.out.println("\n \t " + CaliWeekEnds);
403.
404.
405.        }
406.
407.    }
```

Adding_Spot Class:

```
1. package com.company;
2. import java.util.Scanner;
3. import java.util.ArrayList;
4. import java.io.Serializable;
5.
6. // Adding Spot to Data base
7. public class Renting_Spot  {
8.
9.
10.     public Renting_Spot(){
11.
12.
13.     }
14.
15. // Scanner - File.io
16.     Scanner keyboard = new Scanner(System.in);
17.
18.     static String adress;
19.
20.     int exit = 0;
21.
22.     ArrayList<String> CaliWeekDay = new ArrayList<>();
23.
24.     ArrayList<String> CaliWeekEnds = new ArrayList<>();
25.
26.     ArrayList<String> BogotaWeekDay = new ArrayList<>();
27.
28.     ArrayList<String> BogotaWeekEnds = new ArrayList<>();
29.
30.
31.
32.     // Method to add spot in Cali during Week Days
33.
34.     public ArrayList addSpotCaliWeekDay(ArrayList CaliWeekDay){
35.
36.         int yes = 0;
37.
38.
39.        // Loop with numerical condition
40.         while(yes == 0) {
41.
42.             System.out.println("\n Parking Spot Address: ");
43.             adress = keyboard.nextLine();
44.
45.             System.out.println();
46.             System.out.println("\n The parking spot address is: " + adress);
47.             System.out.println("\n Is this address correct? 'yes' or 'no' ");
48.             String correct = keyboard.next();
49.
50.             if (correct.equals("yes")) {
51.
```

```
52.                 yes = yes + 1;
53.
54.                 CaliWeekDay.add(adress);
55.
56.
57.                 System.out.println("\n These are all the available spots in
    Cali during Weekdays: ");
58.                 System.out.println();
59.
60.                 for (int i = 0; i < CaliWeekDay.size(); i++) {
61.
62.                     System.out.println((i + 1) + ") " +
    CaliWeekDay.get(i));
63.
64.                 }
65.
66.             } else if (correct.equals("no")) {
67.
68.                 adress = keyboard.nextLine();
69.             }
70.
71.         }
72.
73.         return CaliWeekDay;
74.
75.     }
76.
77.
78.     // Method to add spot in Cali during Week Ends
79.
80.     public ArrayList addSpotCaliWeekEnds(ArrayList CaliWeekEnds){
81.
82.         int x = 0;
83.         while(x == 0) {
84.             System.out.println("\n Parking Spot Address: ");
85.             adress = keyboard.nextLine();
86.
87.             System.out.println();
88.             System.out.println("\n The parking spot address is: " + adress);
89.             System.out.println("\n Is this address correct? 'yes' or 'no' ");
90.             String correct = keyboard.next();
91.
92.             if(correct.equals("yes")) {
93.               // Sentinel/Flag - Changing condition
94.                 x = x + 1;
95.
96.                 // Adding element to arraylist
97.                 CaliWeekEnds.add(adress);
98.
99.
100.                    System.out.println("\n These are all the available spots
    in Cali during WeekEnds: ");
101.                    System.out.println();
102.
```

```
                    for (int i = 0; i < CaliWeekEnds.size(); i++) {

                        System.out.println((i + 1) + ") " +
CaliWeekEnds.get(i));

                    }

                }

                else if (correct.equals("no")){
                    adress = keyboard.nextLine();
                }
            }
            return CaliWeekEnds;

        }




        // Method to add spot in Bogota during Week Days

        public ArrayList addSpotBogotaWeekDay(ArrayList BogotaWeekDay){

            int y = 0;

            while (y == 0) {
                System.out.println("\n Parking Spot Address: ");
                adress = keyboard.nextLine();

                System.out.println();
                System.out.println("\n The parking spot address is: " +
adress);
                System.out.println("\n Is this address correct? 'yes' or 'no'
");
                String correct = keyboard.next();

                if(correct.equals("yes")) {

                    y = y + 1;

                    BogotaWeekDay.add(adress);


                    System.out.println("\n These are all the available spots
in Bogota during WeekDays: ");
                    System.out.println();

                    for (int i = 0; i < BogotaWeekDay.size(); i++) {

                        System.out.println((i + 1) + ") " +
BogotaWeekDay.get(i));

                    }
```

```
152.
153.                 }
154.
155.                 else if (correct.equals("no")){
156.                     adress = keyboard.nextLine();
157.                 }
158.
159.             }
160.             return BogotaWeekDay;
161.
162.         }
163.
164.         // Method to add spot in Bogota during Week Ends
165.
166.         public ArrayList addSpotBogotaWeekEnds(ArrayList BogotaWeekEnds){
167.
168.             int z = 0;
169.
170.            while (z == 0) {
171.                 System.out.println("\n Parking Spot Address: ");
172.                 adress = keyboard.nextLine();
173.
174.                 System.out.println();
175.                 System.out.println("\n The parking spot address is: " +
    adress);
176.                 System.out.println("\n Is this address correct? 'yes' or
    'no' ");
177.                 String correct = keyboard.next();
178.
179.                 if (correct.equals("yes")) {
180.
181.                     z = z + 1;
182.
183.                     BogotaWeekEnds.add(adress);
184.
185.
186.                     System.out.println("\n These are all the available spots in
    Bogota during WeekEnds: ");
187.                     System.out.println();
188.
189.                     for (int i = 0; i < BogotaWeekEnds.size(); i++) {
190.
191.                         System.out.println((i + 1) + ") " +
    BogotaWeekEnds.get(i));
192.
193.                     }
194.
195.                 }
196.
197.                 else if (correct.equals("no")){
198.                     adress = keyboard.nextLine();
199.                 }
200.
201.             }
```

User methods

```
202.
203.             return BogotaWeekEnds;
204.
205.         }
206.
207.
208.     }
```

RentingOut_Spot Class:

```
1. package com.company;
2. import java.util.Scanner;
3. import java.util.ArrayList;
4.
5. // Renting Out Spot
6. public class Looking_Spot {
7.
8.
9.
10.     public Looking_Spot(){
11.
12.
13.     }
14.         // Scanner - File.io
15.     Scanner keyboard = new Scanner(System.in);
16.
17.     static int adressimput;
18.
19.     public static int days;
20.
21.     int exit = 0;
22.         // User-defined object
23.     Extras extras = new Extras();
24.
25.
26.
27.     // Method to show available spots in Cali, during week days
28.
29.     public ArrayList availableSpotsCaliWeekDay(ArrayList CaliWeekDay) {
30.
31.
32.         System.out.println();
33.
34.         if(CaliWeekDay.size() >= 1 ) {
35.             System.out.println();
36.             System.out.println("\n These are the available parking spots in
    Cali During Week Days:");
37.
38.             System.out.println();
39.             for (int i = 0; i < CaliWeekDay.size(); i++) {
40.
41.                 System.out.println((i + 1) + ") " + CaliWeekDay.get(i));
42.
43.             }
44.
45.             System.out.println();
46.             System.out.println("\n Please choose an option. Type the number
    that corresponds" +
```

```
                    " to the address which is being rented out");
            adressimput = keyboard.nextInt();

            for (int i = 0; i <= CaliWeekDay.size(); i++) {
                    // Searching - comparing values
                if ((adressimput - 1) <= CaliWeekDay.size()) {
                    System.out.println("\n You choose address: " +
CaliWeekDay.get(adressimput-1)
                            + " as the parking spot being rented out");
                    CaliWeekDay.remove((adressimput - 1));
                }


                System.out.println();
                System.out.println("\n Weekdays parking is $10");
                System.out.println("\n For how many days will your client
rent the parking spot?");
                days = keyboard.nextInt();
                if (days < 30) {
                    System.out.println("\n Your client's total cost will be:
$" + extras.priceWeekDays());
                } else if (days >= 30) {
                    System.out.println("\n Your client's monthly promotion
will have a total cost of: $" + extras.priceMonth());
                }

                System.out.println("\n Please choose your client's payment
method. \n 1)Credit card \n 2) Cash");
                int paymentmethod = keyboard.nextInt();

                exit = exit + 1;

                if (paymentmethod == 1) {
                    System.out.println("\n The parking spot has being rented
out. Congratulations!");

                } else if (paymentmethod == 2) {
                    System.out.println("\n The parking spot has being rented
out. Congratulations!");

                    exit++;
                } else {
                    System.out.println("\n ERROR! Your imput is not valid.
Please enter the number that corresponds to your adress choice ");
                    adressimput = keyboard.nextInt();

                }

                if (exit > 0) {
                    break;
                }
            }

        }
```

Searching

```
93.
94.         else if(CaliWeekDay.size() < 1){
95.
96.             System.out.println("\n Their are no available parking spots.
   Please add spots in order to rent them out");
97.
98.             }
99.
100.
101.            return CaliWeekDay;
102.
103.        }
104.
105.
106.        // Method to show available spots in Cali, during week ends
107.
108.        public ArrayList availableSpotsCaliWeekEnds(ArrayList CaliWeekEnds) {
109.
110.
111.            System.out.println();
112.
113.            if (CaliWeekEnds.size() >= 1) {
114.                System.out.println("\n These are the available parking spots
   in Cali During Week Ends:");
115.                System.out.println();
116.
117.                System.out.println();
118.                for (int i = 0; i < CaliWeekEnds.size(); i++) {
119.
120.                    System.out.println((i + 1) + ") " + CaliWeekEnds.get(i));
121.
122.                }
123.
124.                System.out.println();
125.                System.out.println("\n Please choose an option. Type the
   number that corresponds to the address which is being rented out");
126.                adressimput = keyboard.nextInt();
127.
128.                for (int i = 0; i <= CaliWeekEnds.size(); i++) {
129.
130.                    if ((adressimput - 1) <= CaliWeekEnds.size()) {
131.                        System.out.println("\n You choose address: " +
   CaliWeekEnds.get(adressimput - 1) + " as the parking spot being rented out");
132.                        CaliWeekEnds.remove((adressimput - 1));
133.                    }
134.
135.
136.                    System.out.println();
137.                    System.out.println("\n WeekEnds parking is $20");
138.                    System.out.println("\n For how many days will your client
   want to rent the parking spot?");
139.                    days = keyboard.nextInt();
140.                    if (days < 30) {
```

```
141.                         System.out.println("\n Your client's total cost will
    be: $" + extras.priceWeekends());
142.                     } else if (days >= 30) {
143.                         System.out.println("\n Your client's monthly promotion
    will have a total cost of: $" + extras.priceMonth());
144.                     }
145.
146.                     System.out.println("\n Please choose your client's payment
    method. \n 1)Credit card \n 2) Cash");
147.                     int paymentmethod = keyboard.nextInt();
148.
149.                     exit = exit + 1;
150.
151.                     if (paymentmethod == 1) {
152.                         System.out.println("\n The parking spot has being
    rented out. Congratulations!");
153.
154.                     } else if (paymentmethod == 2) {
155.                         System.out.println("\n The parking spot has being
    rented out. Congratulations!");
156.
157.                         exit++;
158.                     } else {
159.                         System.out.println("\n ERROR! Your imput is not valid.
    Please enter the number that corresponds to your adress choice ");
160.                         adressimput = keyboard.nextInt();
161.
162.                     }
163.
164.                     if (exit > 0) {
165.                         break;
166.                     }
167.
168.                 }
169.
170.             }
171.     // else if statement
172.             else if(CaliWeekEnds.size() < 1){
173.
174.                 System.out.println("\n Their are no available parking spots.
    Please add spots in order to rent them out");
175.             }
176.
177.
178.             return CaliWeekEnds;
179.
180.         }
181.
182.
183.
184.
185.         // Method to show available spots in Bogota, during week days
186.
```

```
187.       public ArrayList availableSpotsBogotaWeekDay(ArrayList BogotaWeekDay)
   {
188.
189.
190.           System.out.println();
191.
192.           if(BogotaWeekDay.size() >= 1) {
193.               System.out.println("\n These are the available parking spots
   in Bogota During Week Days.");
194.               System.out.println();
195.
196.               for (int i = 0; i < BogotaWeekDay.size(); i++) {
197.
198.                   System.out.println((i + 1) + ") " + BogotaWeekDay.get(i));
199.
200.               }
201.
202.               System.out.println();
203.               System.out.println("\n Please choose an option. Type the
   number that corresponds to the address which is being rented out");
204.               adressimput = keyboard.nextInt();
205.
206.               for (int i = 0; i <= BogotaWeekDay.size(); i++) {
207.
208.                   if ((adressimput - 1) <= BogotaWeekDay.size()) {
209.                       System.out.println("\n You choose address: " +
   BogotaWeekDay.get(adressimput - 1) + " as your parking place");
210.                       BogotaWeekDay.remove((adressimput - 1));
211.                   }
212.
213.
214.                   System.out.println();
215.                   System.out.println("\n Weekdays parking is $10");
216.                   System.out.println("\n For how many days will you want to
   rent the parking spot?");
217.                   days = keyboard.nextInt();
218.                   if (days < 30) {
219.                       System.out.println("\n Your total cost will be: $" +
   extras.priceWeekDays());
220.                   } else if (days >= 30) {
221.                       System.out.println("\n Your monthly promotion will
   have a total cost of: $" + extras.priceMonth());
222.                   }
223.
224.                   System.out.println("\n Please choose your payment method.
   \n 1)Credit card \n 2) Cash");
225.                   int paymentmethod = keyboard.nextInt();
226.
227.                   exit = exit + 1;
228.
229.                   if (paymentmethod == 1) {
230.                       System.out.println("\n Thank you! Your parking spot
   confirmation will be sent shortly");
231.
```

```
232.                     } else if (paymentmethod == 2) {
233.                         System.out.println("\n Thank you! Your parking spot
    confirmation will be sent shortly");
234.
235.                         exit++;
236.                     } else {
237.                         System.out.println("\n ERROR! Your imput is not valid.
    Please enter the number that corresponds to your adress choice ");
238.                         adressimput = keyboard.nextInt();
239.
240.                     }
241.
242.                     if (exit > 0) {
243.                         break;
244.                     }
245.
246.                 }
247.
248.             }
249.
250.             else if (BogotaWeekDay.size() < 1){
251.
252.                 System.out.println("\n Their are no available parking spots.
    Please add spots in order to rent them out");
253.
254.             }
255.
256.
257.             return BogotaWeekDay;
258.
259.         }
260.
261.
262.         // Method to show available spots in Bogota, during week ends
263.
264.         public ArrayList availableSpotsBogotaWeekEnds(ArrayList
    BogotaWeekEnds) {
265.
266.
267.             System.out.println();
268.
269.             if (BogotaWeekEnds.size() >=1) {
270.                 System.out.println("\n These are the available parking spots
    in Bogota During Week Ends.");
271.                 System.out.println();
272.
273.                 for (int i = 0; i < BogotaWeekEnds.size(); i++) {
274.
275.                     System.out.println((i + 1) + ") " +
    BogotaWeekEnds.get(i));
276.
277.                 }
278.
279.                 System.out.println();
```

```
280.                System.out.println("\n Please choose an option. Type the
    number that corresponds to the address which is being rented out");
281.                adressimput = keyboard.nextInt();
282.
283.                for (int i = 0; i <= BogotaWeekEnds.size(); i++) {
284.
285.                    if ((adressimput - 1) <= BogotaWeekEnds.size()) {
286.                        System.out.println("\n You choose address: " +
    BogotaWeekEnds.get(adressimput - 1) + " as your parking place");
287.
288.                       //Removing element from arraylist
289.                        BogotaWeekEnds.remove((adressimput - 1));
290.                    }
291.
292.
293.                    System.out.println();
294.                    System.out.println("\n WeekEnds parking is $20");
295.                    System.out.println("\n For how many days will you want to
    rent the parking spot?");
296.                    days = keyboard.nextInt();
297.                    if (days < 30) {
298.                        System.out.println("\n Your total cost will be: $" +
    extras.priceWeekends());
299.                    } else if (days >= 30) {
300.                        System.out.println("\n Your monthly promotion will
    have a total cost of: $" + extras.priceMonth());
301.                    }
302.
303.                    System.out.println("\n Please choose your payment method.
    \n 1)Credit card \n 2) Cash");
304.                    int paymentmethod = keyboard.nextInt();
305.
306.                    exit = exit + 1;
307.
308.                    if (paymentmethod == 1) {
309.                        System.out.println("\n Thank you! Your parking spot
    confirmation will be sent shortly");
310.
311.                    } else if (paymentmethod == 2) {
312.                        System.out.println("\n Thank you! Your parking spot
    confirmation will be sent shortly");
313.
314.                        exit++;
315.                    } else {
316.                        System.out.println("\n ERROR! Your imput is not valid.
    Please enter the number that corresponds to your adress choice ");
317.                        adressimput = keyboard.nextInt();
318.
319.                    }
320.
321.                    if (exit > 0) {
322.                        break;
323.                    }
324.
```

```
                }

            }

            else if (BogotaWeekEnds.size() < 1){

                System.out.println("\n Their are no available parking spots. Please add spots in order to rent them out");

            }


            return BogotaWeekEnds;

        }



    }
```

Extras Class:

```
package com.company;

public class Extras {

    public Extras(){


    }

    static int priceParkWeekDays = 15000;

    static int priceParkWeekends = 10000;

    static int priceMonth = 250000;


// method to calculate parking spot price during weekdays
    public static int priceWeekDays() {

        int pwd = priceParkWeekDays * Looking_Spot.days;

        return pwd;

    }
```

```
26.
27.
28.     // Method to calculate Parking Spot price during weekends.
29.
30.     public static int priceWeekends() {
31.
32.         int pwe = priceParkWeekends * Looking_Spot.days;
33.
34.         return pwe;
35.     }
36.  // method to calculate monthly promotion price
37.     public static int priceMonth(){
38.
39.
40.         return priceMonth;
41.
42.     }
43.
44.
45. }
```