

LANDSCAPE PROJECT

Software Development
Continuous Assessment

- Semester 1, 2022/23 –

Submitted by

DONGHYEOK LEE (21234175)

Table of Contents

1. IPO (Input-Process-Output) Diagram	3
2. Class Diagram	4
3. Flow chart	5
4. Source code explanation	6
4.1. CaLandscapeApp.java	6
4.2. CaFirstLast.java	11
4.3. CaAverHigh.java	13
5. Screenshots of the application's screens/output	16
5.1. Run java file by terminal	16
5.2. The output of an application when the application starts and displays the menu.....	16
5.3. The output of the application after the user provided the information for an item.....	17
5.4. The output displaying the first item.....	18
5.5. The output displaying the last item	19
5.6. The average value of the mandatory state of all the items entered until that point.	20
5.7. The highest mandatory state	21
5.8. Others.....	22
6. Appendix (Source code)	23
6.1. CaLandscapeApp.java	23
6.2. CaFirstLast.java	26
6.3. CaAverHigh.java	28

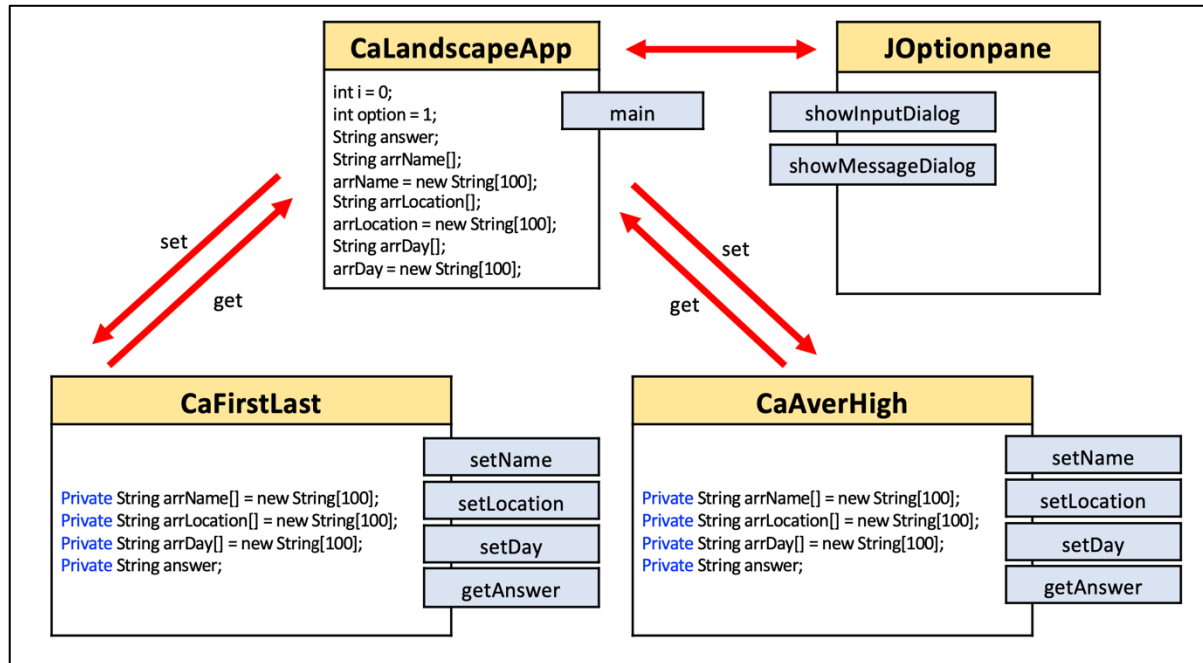
1. IPO (Input-Process-Output) Diagram

- The main input is the user's menu selection, and through the while loop, the process is repeated until option 6 is selected. Answer value will be outputted after the function of each menu, or the other sentences will be outputted through JOptionPane.
- This IPO diagram shows how the code is mapped overall. Design about instantiable classes is explained in the class diagram below.

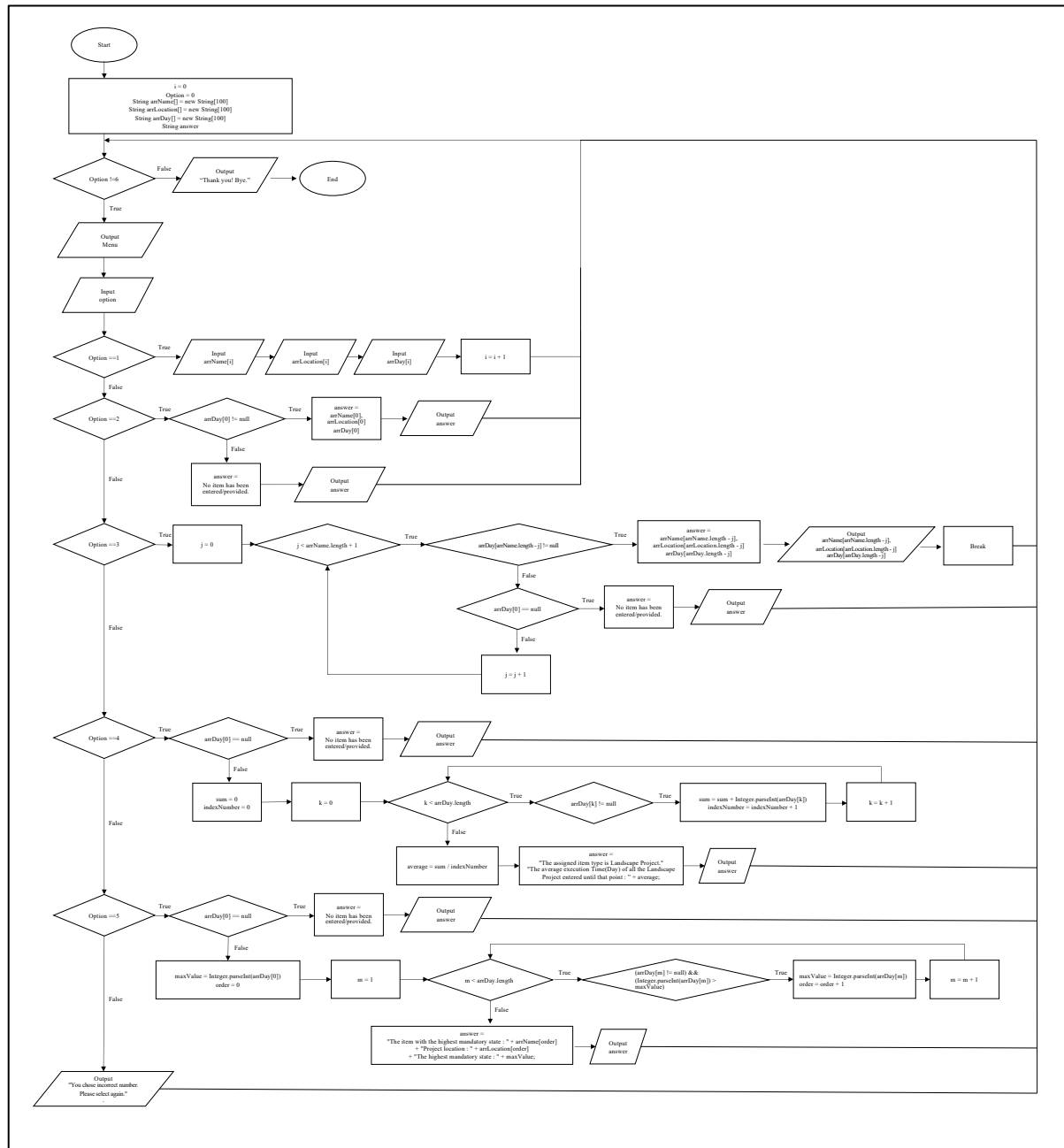
I	P	O
Ask option between 1 ~ 6	<pre> while (option != 6) option = Integer.parseInt(JOptionPane.showInputDialog(null, "Application Menu - Item Type: Landscape" "1 - Add an item." "2 - Display the details of the first entered item." "3 - Display the details of the last entered item." "4 - Calculate and display the average value of the mandatory state of all the items entered until that point." "5 - Calculate and display the item with the highest mandatory state." "6 - Exit application." "Enter your choice:")); Option = 1 arrName[i] = JOptionPane.showInputDialog(null, "Enter the project name"); arrLocation[i] = JOptionPane.showInputDialog(null, "Enter the project location"); arrDay[i] = JOptionPane.showInputDialog(null, "Enter the execution time(Day) of the project"); while ((Integer.parseInt(arrDay[i]) < 2 Integer.parseInt(arrDay[i]) > 42)) { arrDay[i] = JOptionPane.showInputDialog(null, "Invalid mandatory state! Please enter the execution time of the project again"); i = i + 1; } Option = 2 if (arrDay[0] != null) { answer = "The project name : " + arrName[0] + "Project location : " + arrLocation[0] + "Execution Time(Day) of the project : " + arrDay[0]; } else { answer = "No item has been entered/provided."; } Option = 3 for (int j = 1; j < arrName.length + 1; j++) { if (arrDay[arrName.length - j] != null) { answer = "The project name : " + arrName[arrName.length - j] + "\nProject location : " + arrLocation[arrLocation.length - j] + "\nExecution Time(Day) of the project : " + arrDay[arrDay.length - j]; break; } else if ((arrName[0] == null && arrLocation[0] == null && arrDay[0] == null)) { answer = "No item has been entered/provided."; } } Option = 4 if (arrDay[0] == null) { answer = "No item has been entered/provided."; } else { double sum = 0; int indexNumber = 0; double average; for (int k = 0; k < arrDay.length; k++) { if (arrDay[k] != null) { sum = sum + Integer.parseInt(arrDay[k]); indexNumber = indexNumber + 1; } } average = sum / indexNumber; answer = "the assigned item type is Landscape Project" + "the average execution Time(Day) of all the Landscape Project entered until that point : " + average; } } Option = 5 if (arrDay[0] == null) { answer = "No item has been entered/provided."; } else { int maxValue = Integer.parseInt(arrDay[0]); int order = 0; for (int m = 1; m < arrDay.length; m++) { if ((arrDay[m] != null) && (Integer.parseInt(arrDay[m]) > maxValue)) { maxValue = Integer.parseInt(arrDay[m]); order = order + 1; } } answer = "the item with the highest mandatory state : " + arrName[order] + "Project location : " + arrLocation[order] + "the highest mandatory state : " + maxValue; } Option = 6 JOptionPane.showMessageDialog(null, "Thank you! Bye."); Option != 1 ~ 6 JOptionPane.showMessageDialog(null, "You chose incorrect number. Please select again."); </pre>	Display - answer

2. Class Diagram

- This class diagram shows how 3 java files interact on each other (set, get method).
- This program uses 3 Java files, and functions of Option 2 and 3 are implemented through the CaFirstLast class, functions of Option 4 and 5 are implemented through the CaAverHigh class, and the result value is conveyed to the CaLandscapeApp file through the get method.



3. Flow chart



4. Source code explanation

4.1. CaLandscapeApp.java

```
/*
 * CaLandscapeApp.java
 * @author Donghyeok.Lee
 * 27/11/2022
 */
import javax.swing.JOptionPane;

public class CaLandscapeApp {

    public static void main(String args[]) {
        // declare variables
        int i = 0;
        int option = 0;

        String arrName[];
        arrName = new String[100];

        String arrLocation[];
        arrLocation = new String[100];

        String arrDay[];
        arrDay = new String[100];

        String answer;
```

[Explanation]

- Declare 6 variables like above code.
 - int i : It's used for inputting array data [0] ~ [99]. So, it's initialized to '0' at the first time.
 - int option : it's used for user to select functions from 1 to 6. So, it's initialized to '0'. (Actually, it can be initialized to any number because it's replaced into other number that customer will input in following code (JOptionPane)).
 - String arrName[] / arrName = new String[100] : Make array with size of 100 (for enough size).
 - String arrLocation[] / arrLocation = new String[100] : Make array with size of 100 (for enough size).
 - String arrDay[] / arrDay = new String[100] : Make array with size of 100 (for enough size).

```
// declare objects
CaFirstLast myFirstLast;
CaAverHigh myAverHigh;

// create objects
myFirstLast = new CaFirstLast();
myAverHigh = new CaAverHigh();
```

[Explanation]

- Declare and create objects : Because 'CaLandscapeApp.java' file use 2 instantiable class for functions, declare and create 2 objects like above.

```
while (option != 6) {
    option = Integer.parseInt(JOptionPane.showInputDialog(null,
        "Application Menu - Item Type: Landscape"
        + "\n1 - Add an item."
        + "\n2 - Display the details of the first entered item."
        + "\n3 - Display the details of the last entered item."
        + "\n4 - Calculate and display the average value of the mandatory state of
all the items entered until that point."
        + "\n5 - Calculate and display the item with the highest mandatory state."
        + "\n6 - Exit application."
        + "\nEnter your choice:"));
```

[Explanation]

- while (option != 6) : Because the function from option 1 to option 6 is repetitive until user selects option 6 (Exit), While loop is used. This loop will be end when the user selects option 6.

```
if (option == 1) {
    arrName[i] = JOptionPane.showInputDialog(null, "Enter the project name.");
    arrLocation[i] = JOptionPane.showInputDialog(null, "Enter the project location.");
    arrDay[i] = JOptionPane.showInputDialog(null, "Enter the execution time(Day) of
the project.");
    while ((Integer.parseInt(arrDay[i]) < 2 || Integer.parseInt(arrDay[i]) > 42)) {
        arrDay[i] = JOptionPane.showInputDialog(null,
            "Invalid mandatory state! Please enter the execution time of the project
again.");
    }
    i = i + 1;
}
```

[Explanation]

- When the user selects option 1, the user can input data (Project name, project location, and execution time of the project) from array index 0 to index 99. So, after input data, int i (index number) increases 1.
- Because there is a condition that execution time must be between 2 days and 45 days, there is while loop for user to input valid data.

```
else if (option == 2) {
    // input
    myFirstLast.setName(arrName);
    myFirstLast.setLocation(arrLocation);
    myFirstLast.setDay(arrDay);
```

[Explanation]

- Arrays (arrName, arrLocation, and arrDay) data are conveyed to 'myFirstLast' object through 'set' method.

```
// process  
myFirstLast.First();
```

[Explanation]

- Option 2 (what is the first entered item in array) is processed through 'First' method of 'myFirstLast' object.

```
// output  
answer = myFirstLast.getAnswer();  
JOptionPane.showMessageDialog(null, answer);  
}
```

[Explanation]

- String answer is obtained through 'get' method of myFirstLast' object.
- String answer is outputted through JOptionPane.showMessageDialog.

```
else if (option == 3) {  
    // input  
    myFirstLast.setName(arrName);  
    myFirstLast.setLocation(arrLocation);  
    myFirstLast.setDay(arrDay);
```

[Explanation]

- Arrays (arrName, arrLocation, and arrDay) data are conveyed to 'myFirstLast' object through 'set' method.

```
// process  
myFirstLast.Last();
```

[Explanation]

- Option 3(what is the last entered item in array) is processed through 'Last' method of 'myFirstLast' object.

```
// output  
answer = myFirstLast.getAnswer();  
JOptionPane.showMessageDialog(null, answer);  
}
```

[Explanation]

- String answer is obtained through 'get' method of myFirstLast' object.
- String answer is outputted through JOptionPane.showMessageDialog.

```
else if (option == 4) {  
    // input  
    myAverHigh.setName(arrName);
```



```
myAverHigh.setLocation(arrLocation);
myAverHigh.setDay(arrDay);
```

[Explanation]

- Arrays (arrName, arrLocation, and arrDay) data are conveyed to 'myAverHigh' object through 'set' method.

```
// process
myAverHigh.Aver();
```

[Explanation]

- Option 4 (what is the average value of the mandatory state of all the items entered until that point) is processed through 'Aver' method of 'myAverHigh' object.

```
// output
answer = myAverHigh.getAnswer();
JOptionPane.showMessageDialog(null, answer);
}
```

[Explanation]

- String answer is obtained through 'get' method of myAverHigh' object.
- String answer is outputted through JOptionPane.showMessageDialog.

```
else if (option == 5) {
    // input
    myAverHigh.setName(arrName);
    myAverHigh.setLocation(arrLocation);
    myAverHigh.setDay(arrDay);
```

[Explanation]

- Arrays (arrName, arrLocation, and arrDay) data are conveyed to 'myAverHigh' object through 'set' method.

```
// process
myAverHigh.High();
```

[Explanation]

- Option 5(what is the highest mandatory state) is processed through 'High' method of 'myAverHigh' object.

```
// output
answer = myAverHigh.getAnswer();
JOptionPane.showMessageDialog(null, answer);
}
```

[Explanation]

- String answer is obtained through 'get' method of myAverHigh' object.
- String answer is outputted through JOptionPane.showMessageDialog.

```
else if (option == 6) {  
    JOptionPane.showMessageDialog(null, "Thank you! Bye.");  
}
```

[Explanation]

- If the user selects option 6, the message (Thank you! Bye.) will be displayed using `JOptionPane.showMessageDialog` and while loop will be ended.

```
else {  
    JOptionPane.showMessageDialog(null, "You chose incorrect number. Please select  
again.");  
}  
}  
}  
}
```

[Explanation]

- If the user doesn't select option 1 to 6, the message (You chose incorrect number. Please select again.) will be displayed using `JOptionPane.showMessageDialog` for user to select option 1 to 6.

4.2. CaFirstLast.java

```
/*
 * CaFirstLast.java
 * @author Donghyeok.Lee
 * 27/11/2022
 */
public class CaFirstLast {
    // data members
    private String arrName[] = new String[100];
    private String arrLocation[] = new String[100];
    private String arrDay[] = new String[100];
    private String answer;
```

[Explanation]

- Declare 4 variables like above code and these variables is protected and only used by this class using private.

```
// constructor
public CaFirstLast() {
}

// set methods - one for every input
public void setName(String arrName[]) {
    this.arrName = arrName;
}

public void setLocation(String arrLocation[]) {
    this.arrLocation = arrLocation;
}

public void setDay(String arrDay[]) {
    this.arrDay = arrDay;
}
```

[Explanation]

- Pull Arrays (arrName, arrLocation, and arrDay) data from Instantiable class from input.

```
// compute - process
public void First() {
    if (arrDay[0] != null) {
        answer = "The project name : " + arrName[0]
            + "\nProject location : " + arrLocation[0]
            + "\nExecution Time(Day) of the project : " + arrDay[0];
    } else {
        answer = "No item has been entered/provided.";
    }
}
```

[Explanation]

- Because the first entered item in arrays (arrName, arrLocation, and arrDay) are arrName[0], arrLocation[0], and arrDay[0]. So above method outputs these data.
- But it's possible for user not to input any data in array, So, 'if (arrDay[0] != null) and else' is used.

```
public void Last() {  
    for (int j = 1; j < arrName.length + 1; j++) {  
        if (arrDay[arrName.length - j] != null) {  
            answer = "The project name : " + arrName[arrName.length - j]  
                + "\nProject location : " + arrLocation[arrLocation.length - j]  
                + "\nExecution Time(Day) of the project : " + arrDay[arrDay.length - j];  
            break;  
        }  
        else if (arrDay[0] == null) {  
            answer = "No item has been entered/provided.";  
        }  
    }  
}
```

[Explanation]

- Because to check the last entered item in arrays (arrName, arrLocation, and arrDay), arrays data needs to be checked from array index 99 to index 0. So, for loop is used like above.
- But it's possible for user not to input any data in array, so, 'if (arrDay[arrName.length - j] != null) and else' is used.
- In case of '(arrDay[arrName.length - j] != null)', after finding the last entered item, the for loop needs to be ended immediately. So, break is used.

```
// get methods - one for every output  
public String getAnswer() {  
    return answer;  
}  
}
```

[Explanation]

- Send answer data back to CaLandscapeApp.java

4.3. CaAverHigh.java

```
/*
 * CaAverHigh.java
 * @author Donghyeok.Lee
 * 27/11/2022
 */
public class CaAverHigh {
    // data members
    private String arrName[] = new String[100];
    private String arrLocation[] = new String[100];
    private String arrDay[] = new String[100];
    private String answer;
```

[Explanation]

- Declare 4 variables like above code and these variables is protected and only used by this class using private.

```
// constructor
public CaAverHigh() {
}

// set methods - one for every input
public void setName(String arrName[]) {
    this.arrName = arrName;
}

public void setLocation(String arrLocation[]) {
    this.arrLocation = arrLocation;
}

public void setDay(String arrDay[]) {
    this.arrDay = arrDay;
}
```

[Explanation]

- Pull Arrays (arrName, arrLocation, and arrDay) data from Instantiable class from input.

```
// compute - process
public void Aver() {
    if (arrDay[0] == null) {
        answer = "No item has been entered/provided.";
    } else {
        double sum = 0;
        int indexNumber = 0;
        double average;

        for (int k = 0; k < arrDay.length; k++) {
            if (arrDay[k] != null) {
                sum = sum + Integer.parseInt(arrDay[k]);
            }
        }
        average = sum / indexNumber;
    }
}
```

```

        indexNumber = indexNumber + 1;
    }
}
average = sum / indexNumber;
answer = "The assigned item type is Landscape Project."
        + "\nThe average execution Time(Day) of all the Landscape Project entered until
that point : "
        + average;
    }
}

```

[Explanation]

- It's possible for user not to input any data in array, So, 'if (arrDay[0] == null) and else' is used.
- In case of that there is data in array, the average value of the mandatory state of all the items entered until that point is calculated as 'sum / the number of entered data (indexNumber)' in array by using for loop.
- arrDay[] is String. So, it's needed to change data type into integer by using 'Integer.parseInt'.

```

public void High() {
    if (arrDay[0] == null) {
        answer = "No item has been entered/provided.";
    } else {
        int maxValue = Integer.parseInt(arrDay[0]);
        int order = 0;

        for (int m = 1; m < arrDay.length; m++) {
            if ((arrDay[m] != null) && (Integer.parseInt(arrDay[m]) > maxValue)) {
                maxValue = Integer.parseInt(arrDay[m]);
                order = order + 1;
            }
        }
        answer = "The item with the highest mandatory state : " + arrName[order]
                + "\nProject location : " + arrLocation[order]
                + "\nThe highest mandatory state : " + maxValue;
    }
}

```

[Explanation]

- It's possible for user not to input any data in array, So, 'if (arrDay[0] == null) and else' is used.
- In case of that there is data in array, the highest mandatory state is calculated by comparing 'maxValue = Integer.parseInt(arrDay[0])' with 'Integer.parseInt(arrDay[0]) ~ Integer.parseInt(arrDay[99])'. So, for loop is used.
- arrDay[m] is only compared with 'maxValue' in case of that '(arrDay[m] != null)' and 'Integer.parseInt(arrDay[m]) > maxValue'.
- When there are multiple items that meet the same requirement, the first item entered that meets the requirement should be displayed. So, int m increases from 1 to 99 (Not from 99 to 1) and Integer.parseInt(arrDay[m]) '>' maxValue is used (Not '>= ').

- 'int order' is calculated for outputting 'The item with the highest mandatory state (Project name)' and 'Project location' with 'The highest mandatory state'.

```
// get methods - one for every output
public String getAnswer() {
    return answer;
}
}
```

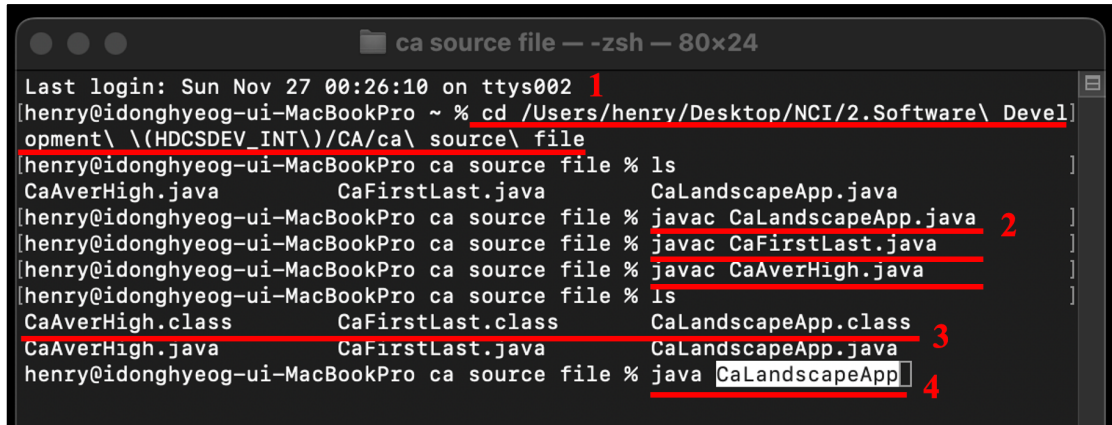
[Explanation]

- Send answer data back to CaLandscapeApp.java

5. Screenshots of the application's screens/output

5.1. Run java file by terminal

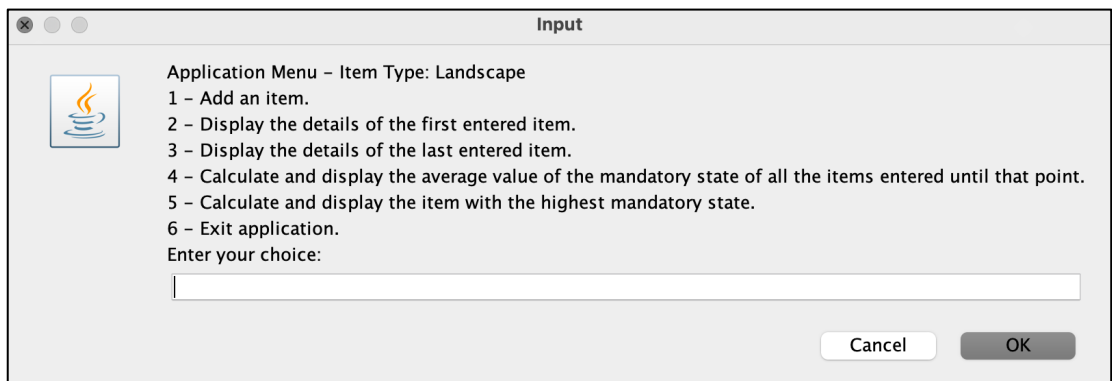
- 1) After opening terminal, write the folder address where java files are stored.
- 2) Compile java files by writing javac (java file name).
- 3) Class files are created.
- 4) Run 'CaLandscapeApp' java file.



```
ca source file — -zsh — 80x24
Last login: Sun Nov 27 00:26:10 on ttys002 1
henry@idonghyeog-ui-MacBookPro ~ % cd /Users/henry/Desktop/NCI/2.Software\ Devel\
opment\ \((HDCSEV_INT\)/CA/ca\ source\ file
henry@idonghyeog-ui-MacBookPro ca source file % ls
CaAverHigh.java      CaFirstLast.java      CaLandscapeApp.java
henry@idonghyeog-ui-MacBookPro ca source file % javac CaLandscapeApp.java 2
henry@idonghyeog-ui-MacBookPro ca source file % javac CaFirstLast.java
henry@idonghyeog-ui-MacBookPro ca source file % javac CaAverHigh.java
henry@idonghyeog-ui-MacBookPro ca source file % ls
CaAverHigh.class      CaFirstLast.class      CaLandscapeApp.class
CaAverHigh.java      CaFirstLast.java      CaLandscapeApp.java 3
henry@idonghyeog-ui-MacBookPro ca source file % java CaLandscapeApp 4
```

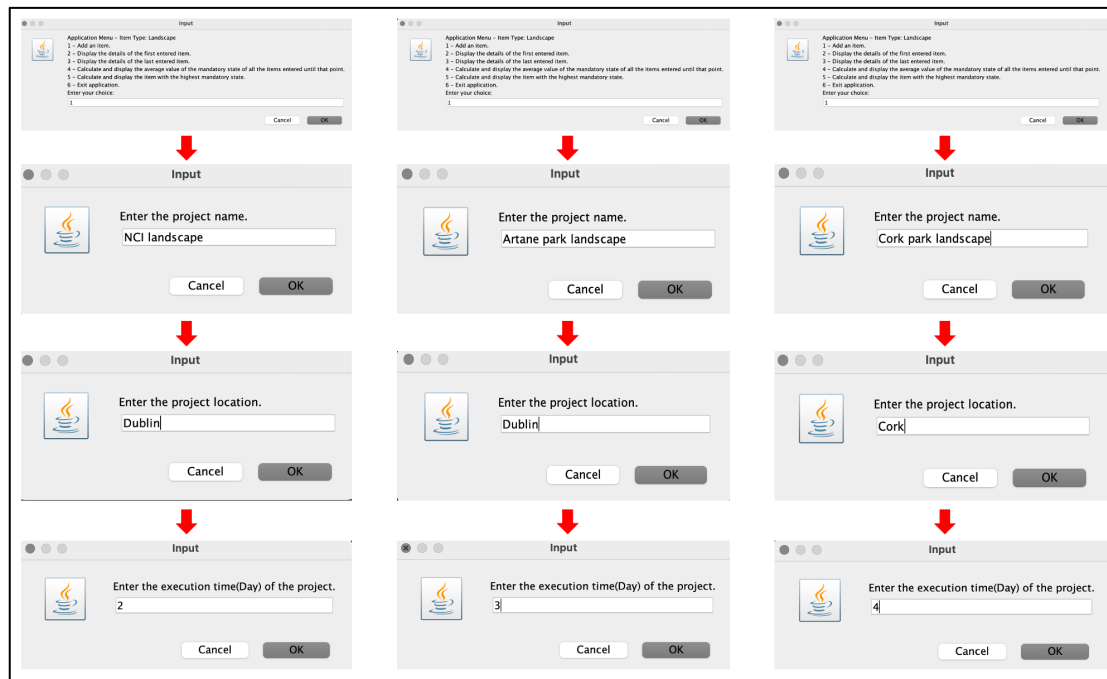
5.2. The output of an application when the application starts and displays the menu

- 1) The menu is displayed on the screen like below.



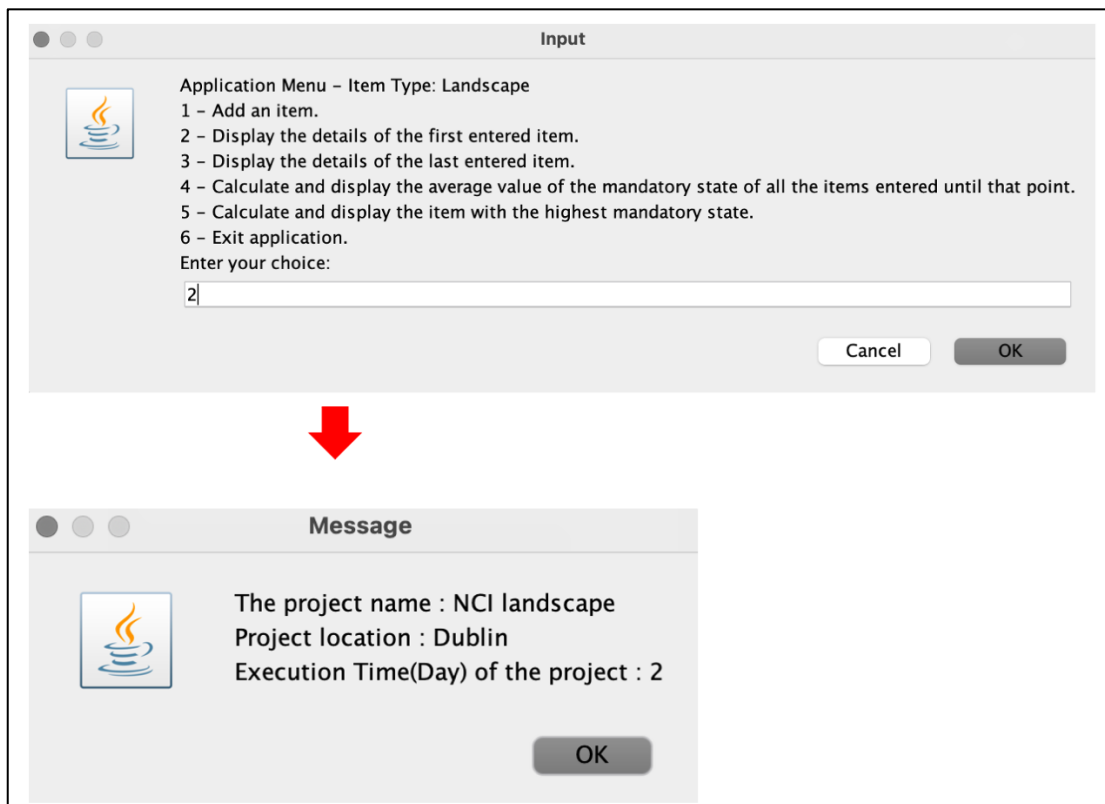
5.3. The output of the application after the user provided the information for an item

- 1) In menu screen, after choosing option 1, user can provide information for an item (Project name, Project location, and Execution time (Day))
- 2) when the user chooses the Option 1, only the information for a single item is provided. If the user wants to provide information about another item, user has to choose Option 1 again like below.

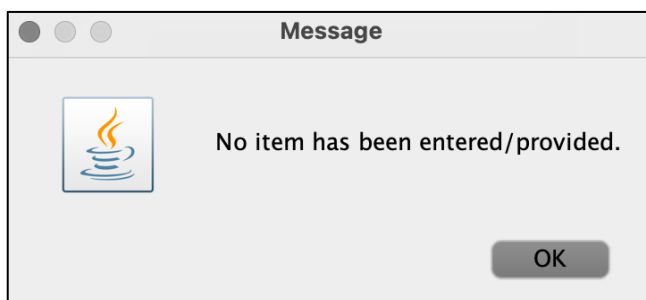


5.4. The output displaying the first item.

- 1) In menu screen, after choosing option 2, user can see the first entered item information like below.

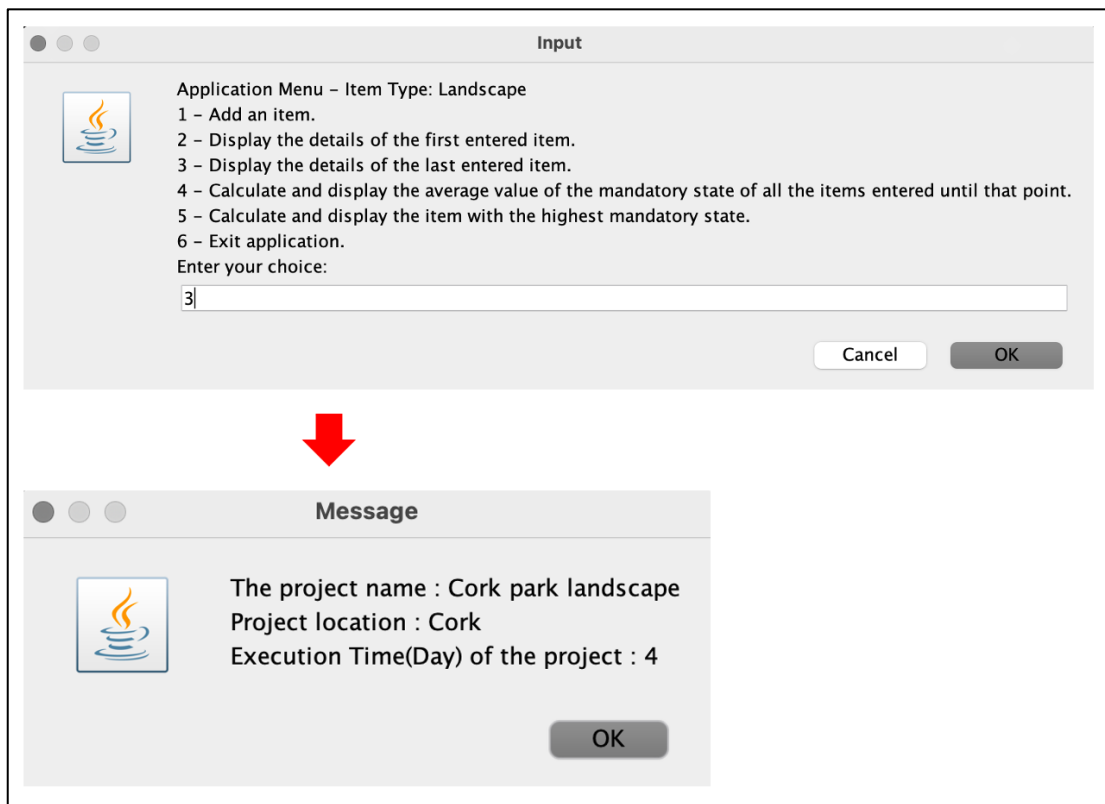


- 2) In case of that there is no entered data, a message will display like below.



5.5. The output displaying the last item .

- 1) In menu screen, after choosing option 3, user can see the last entered item information like below.

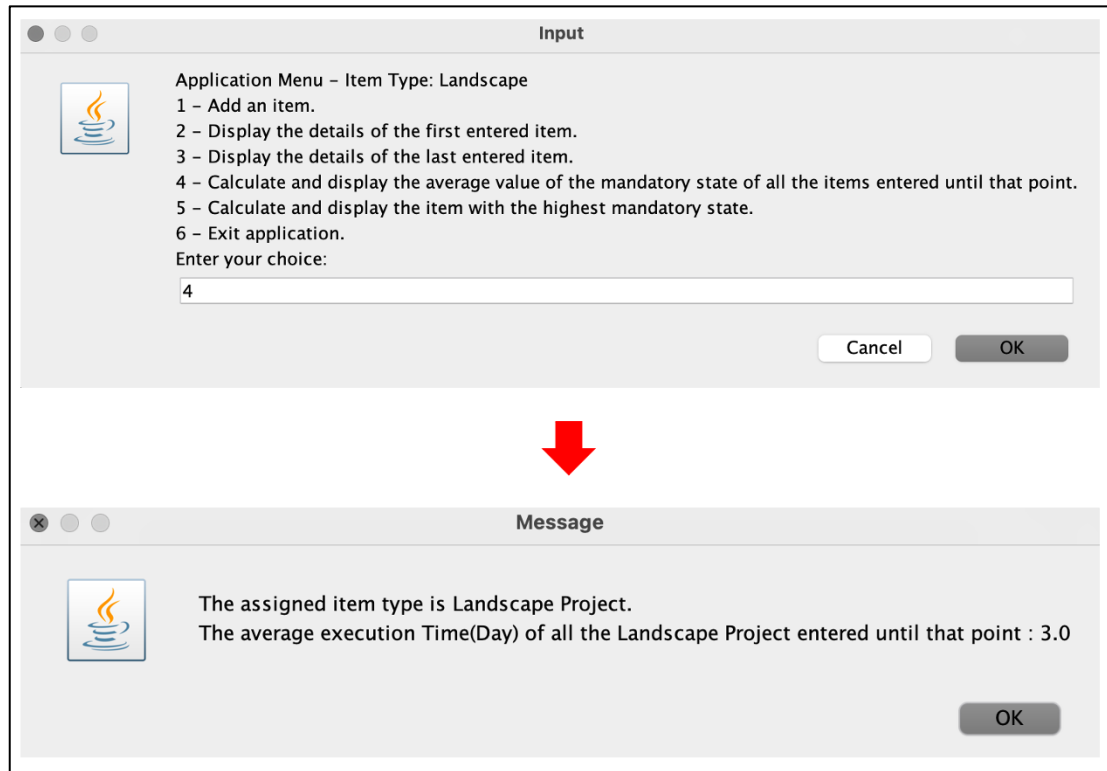


- 2) In case of that there is no entered data, a message will display like below.



5.6. The average value of the mandatory state of all the items entered until that point.

- 1) In menu screen, after choosing option 4, user can see the average value of the mandatory state of all the items entered until that point like below.

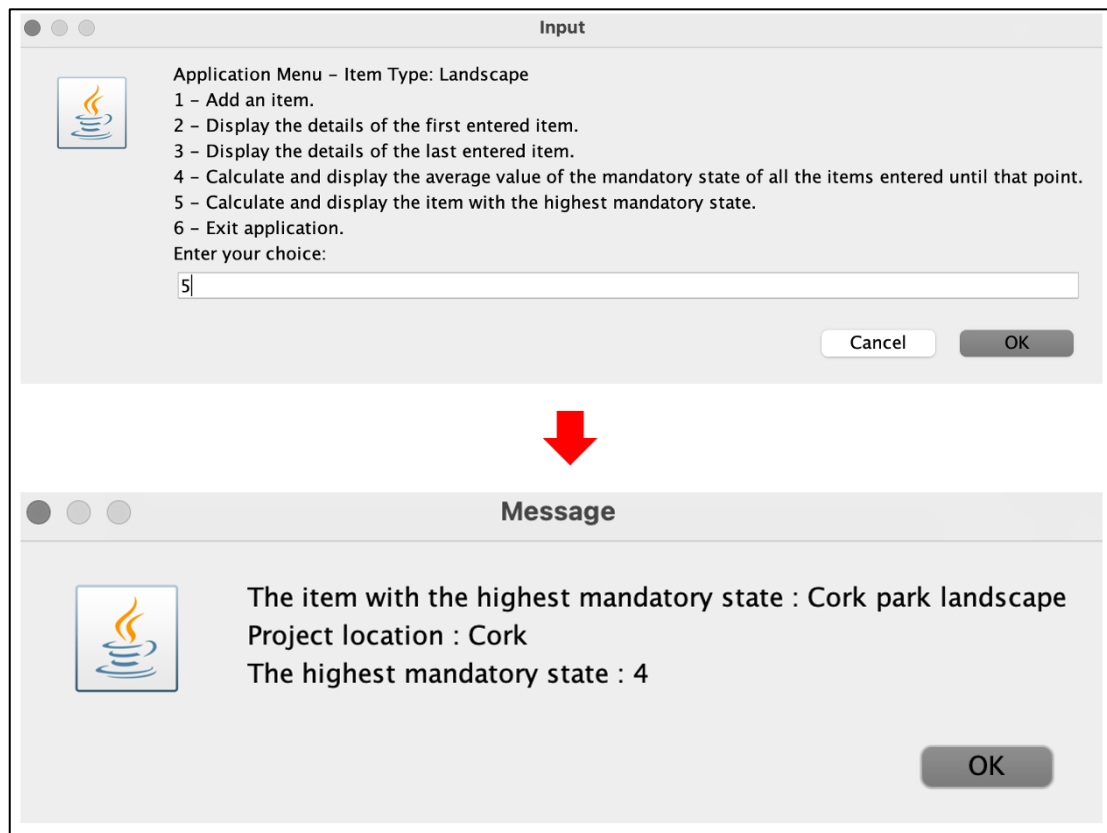


- 2) In case of that there is no entered data, a message will display like below.



5.7. The highest mandatory state

- 1) In menu screen, after choosing option 5, user can see the highest mandatory state among all the items entered until that point like below.

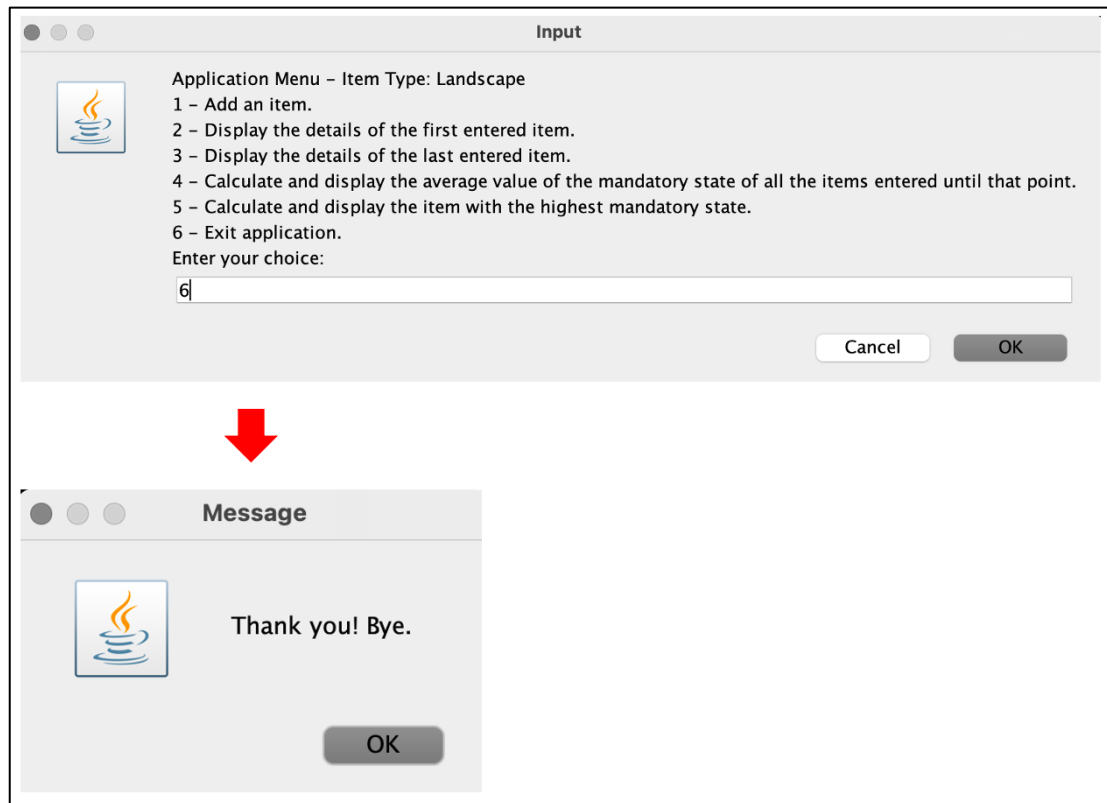


- 2) In case of that there is no entered data, a message will display like below.

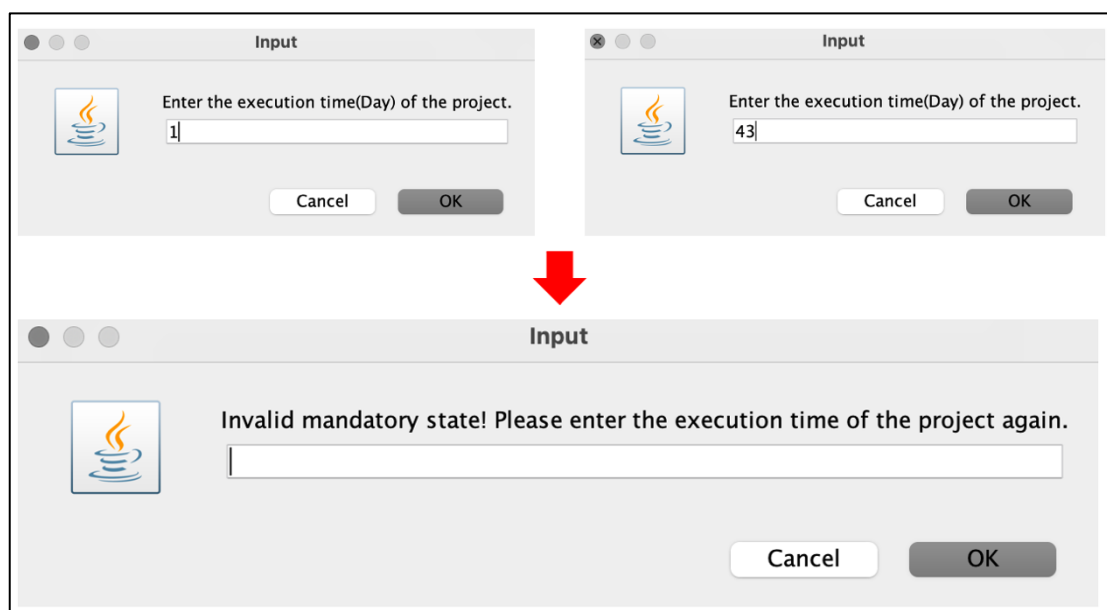


5.8. Others

- 1) In menu screen, after choosing option 6, user can exit the system.



- 2) In case of that user enter execution time (Day) less than 2 or more than 42, message that requests to enter valid data is displayed.



6. Appendix (Source code)

6.1. CaLandscapeApp.java

```
/*
 * CaLandscapeApp.java
 * @author Donghyeok.Lee
 * 27/11/2022
 */
import javax.swing.JOptionPane;

public class CaLandscapeApp {

    public static void main(String args[]) {
        // declare variables
        int i = 0;
        int option = 0;

        String arrName[];
        arrName = new String[100];

        String arrLocation[];
        arrLocation = new String[100];

        String arrDay[];
        arrDay = new String[100];

        String answer;

        // declare objects
        CaFirstLast myFirstLast;
        CaAverHigh myAverHigh;

        // create objects
        myFirstLast = new CaFirstLast();
        myAverHigh = new CaAverHigh();

        while (option != 6) {
            option = Integer.parseInt(JOptionPane.showInputDialog(null,
                "Application Menu - Item Type: Landscape"
                + "\n1 - Add an item."
                + "\n2 - Display the details of the first entered item."
                + "\n3 - Display the details of the last entered item."
                + "\n4 - Calculate and display the average value of the mandatory state of
all the items entered until that point."
                + "\n5 - Calculate and display the item with the highest mandatory state."
                + "\n6 - Exit application."
                + "\nEnter your choice:"));

            if (option == 1) {
```

```

arrName[i] = JOptionPane.showInputDialog(null, "Enter the project name.");
arrLocation[i] = JOptionPane.showInputDialog(null, "Enter the project location.");
arrDay[i] = JOptionPane.showInputDialog(null, "Enter the execution time(Day) of
the project.");
while ((Integer.parseInt(arrDay[i]) < 2 || Integer.parseInt(arrDay[i]) > 42)) {
    arrDay[i] = JOptionPane.showInputDialog(null,
        "Invalid mandatory state! Please enter the execution time of the project
again.");
}
i = i + 1;
}

else if (option == 2) {
    // input
    myFirstLast.setName(arrName);
    myFirstLast.setLocation(arrLocation);
    myFirstLast.setDay(arrDay);

    // process
    myFirstLast.First();

    // output
    answer = myFirstLast.getAnswer();
    JOptionPane.showMessageDialog(null, answer);
}

else if (option == 3) {
    // input
    myFirstLast.setName(arrName);
    myFirstLast.setLocation(arrLocation);
    myFirstLast.setDay(arrDay);

    // process
    myFirstLast.Last();

    // output
    answer = myFirstLast.getAnswer();
    JOptionPane.showMessageDialog(null, answer);
}

else if (option == 4) {
    // input
    myAverHigh.setName(arrName);
    myAverHigh.setLocation(arrLocation);
    myAverHigh.setDay(arrDay);

    // process
    myAverHigh.Aver();

    // output

```



```

        answer = myAverHigh.getAnswer();
        JOptionPane.showMessageDialog(null, answer);
    }

    else if (option == 5) {
        // input
        myAverHigh.setName(arrName);
        myAverHigh.setLocation(arrLocation);
        myAverHigh.setDay(arrDay);

        // process
        myAverHigh.High();

        // output
        answer = myAverHigh.getAnswer();
        JOptionPane.showMessageDialog(null, answer);
    }

    else if (option == 6) {
        JOptionPane.showMessageDialog(null, "Thank you! Bye.");
    }

    else {
        JOptionPane.showMessageDialog(null, "You chose incorrect number. Please select
again.");
    }
}
}
}
}

```

6.2. CaFirstLast.java

```
/*
 * CaFirstLast.java
 * @author Donghyeok.Lee
 * 27/11/2022
 */
public class CaFirstLast {
    // data members
    private String arrName[] = new String[100];
    private String arrLocation[] = new String[100];
    private String arrDay[] = new String[100];
    private String answer;

    // constructor
    public CaFirstLast() {
    }

    // set methods - one for every input
    public void setName(String arrName[]) {
        this.arrName = arrName;
    }

    public void setLocation(String arrLocation[]) {
        this.arrLocation = arrLocation;
    }

    public void setDay(String arrDay[]) {
        this.arrDay = arrDay;
    }

    // compute - process
    public void First() {
        if (arrDay[0] != null) {
            answer = "The project name : " + arrName[0]
                + "\nProject location : " + arrLocation[0]
                + "\nExecution Time(Day) of the project : " + arrDay[0];
        } else {
            answer = "No item has been entered/provided.";
        }
    }

    public void Last() {
        for (int j = 1; j < arrName.length + 1; j++) {
            if (arrDay[arrName.length - j] != null) {
                answer = "The project name : " + arrName[arrName.length - j]
                    + "\nProject location : " + arrLocation[arrLocation.length - j]
                    + "\nExecution Time(Day) of the project : " + arrDay[arrDay.length - j];
                break;
            }
        }
    }
}
```

```
        } else if (arrDay[0] == null) {  
            answer = "No item has been entered/provided.";  
        }  
    }  
}  
  
// get methods - one for every output  
public String getAnswer() {  
    return answer;  
}  
}
```

6.3. CaAverHigh.java

```
/*
 * CaAverHigh.java
 * @author Donghyeok.Lee
 * 27/11/2022
 */
public class CaAverHigh {
    // data members
    private String arrName[] = new String[100];
    private String arrLocation[] = new String[100];
    private String arrDay[] = new String[100];
    private String answer;

    // constructor
    public CaAverHigh() {
    }

    // set methods - one for every input
    public void setName(String arrName[]) {
        this.arrName = arrName;
    }

    public void setLocation(String arrLocation[]) {
        this.arrLocation = arrLocation;
    }

    public void setDay(String arrDay[]) {
        this.arrDay = arrDay;
    }

    // compute - process
    public void Aver() {
        if (arrDay[0] == null) {
            answer = "No item has been entered/provided.";
        } else {
            double sum = 0;
            int indexNumber = 0;
            double average;

            for (int k = 0; k < arrDay.length; k++) {
                if (arrDay[k] != null) {
                    sum = sum + Integer.parseInt(arrDay[k]);
                    indexNumber = indexNumber + 1;
                }
            }
            average = sum / indexNumber;
            answer = "The assigned item type is Landscape Project."
                + "\n\nThe average execution Time(Day) of all the Landscape Project entered until
that point : "
```

```

        + average;
    }
}

public void High() {
    if (arrDay[0] == null) {
        answer = "No item has been entered/provided.";
    } else {
        int maxValue = Integer.parseInt(arrDay[0]);
        int order = 0;

        for (int m = 1; m < arrDay.length; m++) {
            if ((arrDay[m] != null) && (Integer.parseInt(arrDay[m]) > maxValue)) {
                maxValue = Integer.parseInt(arrDay[m]);
                order = order + 1;
            }
        }
        answer = "The item with the highest mandatory state : " + arrName[order]
            + "\nProject location : " + arrLocation[order]
            + "\nThe highest mandatory state : " + maxValue;
    }
}

// get methods - one for every output
public String getAnswer() {
    return answer;
}
}

```