**FIFTH PROJECT: FIX BUGS OF THE APPLICATION**

DESCRIPTION

**Project objective:**

Fix the bugs in the application using the appropriate algorithmic techniques.

**Background of the problem statement:**

You have been assigned a few tasks during the sprint planning. Solving the bugs raised by the testing team is one among them. You are given the boilerplate code and are asked to complete it by fixing the bugs.

**Bugs to be fixed:**

* Add the missing source code to the application based on searching technique. Find the appropriate comments to code for the searching technique.
* Write source code for sorting the predefined array and ensure the functionality of the application. Find the appropriate comments to code for sorting the predefined array.
* You can download the boilerplate code by executing the command below in your **git bash**.

git clone https://github.com/Simplilearn-Edu/Full-Stack---The-Desk-Application-.git

**You must use the following:**

* **Eclipse/IntelliJ:** An IDE to code for the application
* **Java:** A programming language to develop the prototype
* **Git:** To connect and push files from local system to GitHub
* **GitHub:** To store the application code and track its versions
* **Search and Sort techniques:** Select the relevant data structure algorithms to fix the bugs

**Following requirements should be met:**

* The source code should be pushed to your GitHub repositories. You need to document the steps and write the algorithms in the Google Docs.
* The link of your GitHub repository is must. In order to track your task, you need to share the link of the repository. You can add a section in the Google Docs.
* Document the step-by-step process involved in completing this task.

**Creating a new project in Eclipse:**

* Open Eclipse
* Go to File -> New -> Project -> Java Project -> Next.
* Type in project name as **Fifth-Project** and click on “Finish.”
* Select your project and go to File -> New -> Package.
* Enter **com.simplilearn.test** as Package name.
* Select your project and go to File -> New -> Class.
* Enter **Main** as class name, check the checkbox “public static void main(String[] args)”., and click on “Finish.”

**Writing a Java code to fix the bugs in the application using the appropriate algorithmic techniques:**

**package** com.simplilearn.test;

**import** java.util.ArrayList;

**import** java.util.Scanner;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

/\*System.out.println("Hello World!");\*/

System.***out***.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

System.***out***.println("\tWelcome to TheDesk \n");

System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

*optionsSelection*();

}

**private** **static** **void** optionsSelection() {

String[] arr = {"1. I wish to review my expenditure",

"2. I wish to add my expenditure",

"3. I wish to delete my expenditure",

"4. I wish to sort the expenditures",

"5. I wish to search for a particular expenditure",

"6. Close the application"

};

**int**[] arr1 = {1,2,3,4,5,6};

**int** slen = arr1.length;

**for**(**int** i=0; i<slen;i++){

System.***out***.println(arr[i]);

// display the all the Strings mentioned in the String array

}

ArrayList<Integer> arrlist = **new** ArrayList<Integer>();

ArrayList<Integer> expenses = **new** ArrayList<Integer>();

expenses.add(1000);

expenses.add(2300);

expenses.add(45000);

expenses.add(32000);

expenses.add(110);

expenses.addAll(arrlist);

System.***out***.println("\nEnter your choice:\t");

Scanner sc = **new** Scanner(System.***in***);

**int** options = sc.nextInt();

sc.close();

**for**(**int** j=1;j<=slen;j++){

**if**(options==j){

**switch** (options){

**case** 1:

System.***out***.println("Your saved expenses are listed below: \n");

System.***out***.println(expenses+"\n");

*optionsSelection*();

**break**;

**case** 2:

System.***out***.println("Enter the value to add your Expense: \n");

**int** value = sc.nextInt();

expenses.add(value);

System.***out***.println("Your value is updated\n");

expenses.addAll(arrlist);

System.***out***.println(expenses+"\n");

*optionsSelection*();

**break**;

**case** 3:

System.***out***.println("You are about the delete all your expenses! \nConfirm again by selecting the same option...\n");

**int** con\_choice = sc.nextInt();

**if**(con\_choice==options){

expenses.clear();

System.***out***.println(expenses+"\n");

System.***out***.println("All your expenses are erased!\n");

} **else** {

System.***out***.println("Oops... try again!");

}

*optionsSelection*();

**break**;

**case** 4:

*sortExpenses*(expenses);

*optionsSelection*();

**break**;

**case** 5:

*searchExpenses*(expenses);

*optionsSelection*();

**break**;

**case** 6:

*closeApp*();

**break**;

**default**:

System.***out***.println("You have made an invalid choice!");

**break**;

}

}

}

}

**private** **static** **void** sortExpenses(ArrayList<Integer> arrayList) {

ArrayList<Integer> expenses = **new** ArrayList<Integer>();

expenses.add(1000);

expenses.add(2300);

expenses.add(45000);

expenses.add(32000);

expenses.add(110);

**int** arrlength = expenses.size();

Integer[] arr = **new** Integer[expenses.size()];

arr = expenses.toArray(arr);

System.***out***.println(" List of Expenses to Sort: \n "+expenses);

*bubbleSort*(arr);//sorting array elements using bubble sort

System.***out***.println(" List of Expenses Sorted in Ascending order: \n ");

**for**(**int** i=0; i < arrlength; i++)

{

System.***out***.print(arr[i] + " ");

}

}

**static** **void** bubbleSort(Integer[] arr)

{

**int** n = arr.length;

**int** temp = 0;

**for**(**int** i=0; i < n; i++)

{

**for**(**int** j=1; j < (n-i); j++)

{

**if**(arr[j-1] > arr[j])

{

//swap elements

temp = arr[j-1];

arr[j-1] = arr[j];

arr[j] = temp;

}

}

}

}

**private** **static** **void** searchExpenses(ArrayList<Integer> arrayList) {

ArrayList<Integer> expenses = **new** ArrayList<>();

expenses.add(1000);

expenses.add(2300);

expenses.add(45000);

expenses.add(32000);

expenses.add(110);

Integer[] arr = **new** Integer[expenses.size()];

arr = expenses.toArray(arr);

System.***out***.println(" List of Expenses to Search\n "+expenses);

System.***out***.println("Enter the expense you need to search:\t");

Scanner sc1 = **new** Scanner(System.***in***);

**int** key = sc1.nextInt();

sc1.close();

**int** result = *searchexpenses*(expenses,key);

**if**(result ==-1) {

System.***out***.println("Element not in the list of expenses");

}

**else**

{

System.***out***.println("Element found at " +result+ " and the Searched Expense is "+arr[result] );

}

}

**private** **static** **int** searchexpenses(ArrayList<Integer> arrayList, **int** key) {

**int** leng = arrayList.size();

Integer[] arr = **new** Integer[arrayList.size()];

arr = arrayList.toArray(arr);

**for** (**int** i = 0; i < leng; i++) {

**if** (arr[i]== key) {

**return** i;

}

}

**return** -1;

}

**private** **static** **void** closeApp() {

System.***out***.println("Closing your application... \nThank you!");

}

}

**Search and Sort techniques: Select the relevant data structure algorithms to fix the bugs**

**Expenses Sorted in Ascending order** **using Bubble Sort algorithm:**

We can create a java program to sort array elements using bubble sort. Bubble sort algorithm is known as the simplest sorting algorithm.

In bubble sort algorithm, array is traversed from first element to last element. Here, current element is compared with the next element. If current element is greater than the next element, it is swapped.

**private** **static** **void** sortExpenses(ArrayList<Integer> arrayList) {

ArrayList<Integer> expenses = **new** ArrayList<Integer>();

expenses.add(1000);

expenses.add(2300);

expenses.add(45000);

expenses.add(32000);

expenses.add(110);

**int** arrlength = expenses.size();

Integer[] arr = **new** Integer[expenses.size()];

arr = expenses.toArray(arr);

System.***out***.println(" List of Expenses to Sort: \n "+expenses);

*bubbleSort*(arr);//sorting array elements using bubble sort

System.***out***.println(" List of Expenses Sorted in Ascending order: \n ");

**for**(**int** i=0; i < arrlength; i++)

{

System.***out***.print(arr[i] + " ");

}

}

**static** **void** bubbleSort(Integer[] arr)

{

**int** n = arr.length;

**int** temp = 0;

**for**(**int** i=0; i < n; i++)

{

**for**(**int** j=1; j < (n-i); j++)

{

**if**(arr[j-1] > arr[j])

{

//swap elements

temp = arr[j-1];

arr[j-1] = arr[j];

arr[j] = temp;

}

}

}

}

**Output:**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to The Desk

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. I wish to review my expenditure

2. I wish to add my expenditure

3. I wish to delete my expenditure

4. I wish to sort the expenditures

5. I wish to search for a particular expenditure

6. Close the application

Enter your choice:

4

List of Expenses to Sort:

[1000,2300,45000,32000,110]

List of Expenses Sorted in Ascending order:

110,1000,2300,32000,45000

**Expenses Searched**  **using Linear Search algorithm:**

Linear search is used to search a key element from multiple elements. Linear search is less used today because it is slower than binary search and hashing.

**Algorithm:**

* Step 1: Traverse the array
* Step 2: Match the key element with array element
* Step 3: If key element is found, return the index position of the array element
* Step 4: If key element is not found, return -1

It is shown using the following program for searching the particular expenditure from list of expenses

**private** **static** **void** searchExpenses(ArrayList<Integer> arrayList) {

ArrayList<Integer> expenses = **new** ArrayList<>();

expenses.add(1000);

expenses.add(2300);

expenses.add(45000);

expenses.add(32000);

expenses.add(110);

Integer[] arr = **new** Integer[expenses.size()];

arr = expenses.toArray(arr);

System.***out***.println(" List of Expenses to Search\n "+expenses);

System.***out***.println("Enter the expense you need to search:\t");

Scanner sc1 = **new** Scanner(System.***in***);

**int** key = sc1.nextInt();

sc1.close();

**int** result = *searchexpenses*(expenses,key);

**if**(result ==-1) {

System.***out***.println("Element not in the list of expenses");

}

**else**

{

System.***out***.println("Element found at " +result+ " and the Searched Expense is "+arr[result] );

}

}

**private** **static** **int** searchexpenses(ArrayList<Integer> arrayList, **int** key) {

**int** leng = arrayList.size();

Integer[] arr = **new** Integer[arrayList.size()];

arr = arrayList.toArray(arr);

**for** (**int** i = 0; i < leng; i++) {

**if** (arr[i]== key) {

**return** i;

}

}

**return** -1;

}

**Output:**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to The Desk

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. I wish to review my expenditure

2. I wish to add my expenditure

3. I wish to delete my expenditure

4. I wish to sort the expenditures

5. I wish to search for a particular expenditure

6. Close the application

Enter your choice:

5

List of Expenses to Search

[1000,2300,45000,32000,110]

Enter the expense you need to search:

45000

Element found at 2 and the Searched Expense is 45000

**Pushing the code to your GitHub repositories:**

* Open your command prompt and navigate to the folder where you have created your files.

**cd <folder path>**

* Initialize your repository using the following command:

**git init**

* Add all the files to your git repository using the following command:

**git add .**

* Commit the changes using the following command:

**git commit . -m “Changes have been committed.”**

* Push the files to the folder you initially created using the following command:

**git push -u origin master**