

```
Simplilearn_projects - Practice_Project_5.20/src/Main.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help

Main.java
1 import java.util.ArrayList;
2 import java.util.Collections;
3 import java.util.InputMismatchException;
4 import java.util.Scanner;
5
6 public class Main {
7
8     private static ArrayList<Integer> expenses = new ArrayList<Integer>();
9
10    public static void main(String[] args) {
11        System.out.println("\n*****\n");
12        System.out.println("Welcome to TheDesk \n");
13        System.out.println("*****\n");
14        optionsSelection();
15    }
16
17    private static void optionsSelection() {
18        String[] arr = {
19            "1. I wish to review my expenditure",
20            "2. I wish to add my expenditure",
21            "3. I wish to delete my expenditure",
22            "4. I wish to sort the expenditures",
23            "5. I wish to search for a particular expenditure",
24            "6. Close the application"
25        };
26
27        for (int i = 0; i < arr.length; i++) {
28            System.out.println(arr[i]);
29        }
30
31        System.out.println("\nEnter your choice:\n");
32
33        try (Scanner sc = new Scanner(System.in)) {
34            int options = sc.nextInt();
35
36            switch (options) {
37                case 1:
38                    System.out.println("Your saved expenses are listed below: \n");
39                    System.out.println(expenses + "\n");
40                    optionsSelection();
41                    break;
42                case 2:
43                    System.out.println("Enter the value to add your Expense: \n");
44                    int value = sc.nextInt();
45                    expenses.add(value);
46                    System.out.println("Your value is updated\n");
47                    System.out.println(expenses + "\n");
48                    optionsSelection();
49                    break;
50                case 3:
51                    System.out.println("You are about to delete all your expenses! \nConfirm again by selecting the same option...\n");
52                    int con_choice = sc.nextInt();
53                    if (con_choice == options) {
54                        expenses.clear();
55                        System.out.println(expenses + "\n");
56                    } else {
57                        System.out.println("Oops... try again!");
58                    }
59                    optionsSelection();
60                    break;
61                case 4:
62                    sortExpenses(expenses);
63                    optionsSelection();
64                    break;
65                case 5:
66                    try (Scanner searchScanner = new Scanner(System.in)) {
67                        System.out.println("Enter the expense you need to search:\n");
68                        int expenseToSearch = searchScanner.nextInt();
69                        searchExpenses(expenses, expenseToSearch);
70                    } catch (InputMismatchException e) {
71                        System.out.println("Invalid input. Please enter a valid expense value.");
72                    }
73                    optionsSelection();
74                    break;
75                case 6:
76                    closeApp();
77                    break;
78                default:
79                    System.out.println("You have made an invalid choice!");
80                    break;
81            }
82        } catch (InputMismatchException e) {
83            System.out.println("Invalid input. Please enter a valid choice.");
84            optionsSelection();
85        }
86    }
87
88    private static void closeApp() {
89        System.out.println("Closing your application... \nThank you!");
90    }
91
92    private static void searchExpenses(ArrayList<Integer> arrayList, int expenseToSearch) {
93        int leng = arrayList.size();
94
95        boolean found = false;
96        for (int i = 0; i < leng; i++) {
97            if (arrayList.get(i) == expenseToSearch) {
98                found = true;
99                break;
100            }
101        }
102
103        if (found) {
104            System.out.println("Expense found: " + expenseToSearch);
105        } else {
106            System.out.println("Expense not found");
107        }
108    }
109
110    private static void sortExpenses(ArrayList<Integer> arrayList) {
111        Collections.sort(arrayList);
112        System.out.println("Sorted expenses: " + arrayList);
113    }
114}
```

```
Simplilearn_projects - Practice_Project_5.20/src/Main.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help

Main.java
50
51     case 3:
52         System.out.println("You are about to delete all your expenses! \nConfirm again by selecting the same option...\n");
53         int con_choice = sc.nextInt();
54         if (con_choice == options) {
55             expenses.clear();
56             System.out.println(expenses + "\n");
57             System.out.println("All your expenses are erased!\n");
58         } else {
59             System.out.println("Oops... try again!");
60         }
61         optionsSelection();
62         break;
63     case 4:
64         sortExpenses(expenses);
65         optionsSelection();
66         break;
67     case 5:
68         try (Scanner searchScanner = new Scanner(System.in)) {
69             System.out.println("Enter the expense you need to search:\n");
70             int expenseToSearch = searchScanner.nextInt();
71             searchExpenses(expenses, expenseToSearch);
72         } catch (InputMismatchException e) {
73             System.out.println("Invalid input. Please enter a valid expense value.");
74         }
75         optionsSelection();
76         break;
77     case 6:
78         closeApp();
79         break;
80     default:
81         System.out.println("You have made an invalid choice!");
82         break;
83     } catch (InputMismatchException e) {
84         System.out.println("Invalid input. Please enter a valid choice.");
85         optionsSelection();
86     }
87 }
88
89 private static void closeApp() {
90     System.out.println("Closing your application... \nThank you!");
91 }
92
93 private static void searchExpenses(ArrayList<Integer> arrayList, int expenseToSearch) {
94     int leng = arrayList.size();
95
96     boolean found = false;
97     for (int i = 0; i < leng; i++) {
98         if (arrayList.get(i) == expenseToSearch) {
99             found = true;
100             break;
101         }
102     }
103
104     if (found) {
105         System.out.println("Expense found: " + expenseToSearch);
106     } else {
107         System.out.println("Expense not found");
108     }
109 }
110
111 private static void sortExpenses(ArrayList<Integer> arrayList) {
112     Collections.sort(arrayList);
113     System.out.println("Sorted expenses: " + arrayList);
114 }
```

```
Simplilearn_projects - Practice_Project_5.20/src/Main.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Main.java x
61
62     case 4:
63         sortExpenses(expenses);
64         optionsSelection();
65         break;
66     case 5:
67         try (Scanner searchScanner = new Scanner(System.in)) {
68             System.out.println("Enter the expense you need to search:\t");
69             int expenseToSearch = searchScanner.nextInt();
70             searchExpenses(expenses, expenseToSearch);
71         } catch (InputMismatchException e) {
72             System.out.println("Invalid input. Please enter a valid expense value.");
73         }
74         optionsSelection();
75         break;
76     case 6:
77         closeApp();
78         break;
79     default:
80         System.out.println("You have made an invalid choice!");
81         break;
82 }
83 } catch (InputMismatchException e) {
84     System.out.println("Invalid input. Please enter a valid choice.");
85     optionsSelection();
86 }
87 }
88
89 private static void closeApp() {
90     System.out.println("Closing your application... \nThank you!");
91 }
92
93 private static void searchExpenses(ArrayList<Integer> arrayList, int expenseToSearch) {
94     int leng = arrayList.size();
95     boolean found = false;
96     for (int i = 0; i < leng; i++) {
97         if (arrayList.get(i) == expenseToSearch) {
98             found = true;
99             break;
100         }
101     }
102     if (found) {
103         System.out.println("Expense found in the list.");
104     } else {
105         System.out.println("Expense not found in the list.");
106     }
107 }
108
109 private static void sortExpenses(ArrayList<Integer> arrayList) {
110     Collections.sort(arrayList);
111     System.out.println("Expenses sorted in ascending order: " + arrayList);
112 }
113
114 }
115
116
4
Writab... Smart Insert 47:50:1773
```