```
import java.util.ArrayList;
import java.util.Collections;
import java.util.Scanner;
import java.util.TreeSet;
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();
            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");
                        System.out.println(expenses + "\n");
                        expenses.addAll(arrlist);
                        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 closeApp() {
```

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


        private static void searchExpenses(ArrayList<Integer>
arrayList) {
            int leng = arrayList.size();
            System.out.println("Enter the expense you need to
search:\t");
            // Complete the method
            Scanner s = new Scanner(System.in);
            Integer Expenses = s.nextInt();
            if(arrayList.contains(Expenses))
            {
                System.out.println("Expenses Found at
"+arrayList.indexOf(Expenses));
            }
            else
            {
                System.out.println("No Expenses Found");
            }


        }


        private static void sortExpenses(ArrayList<Integer> arrayList) {
            int arrlength = arrayList.size();
```

```
                // Complete the method. The expenses should be sorted in ascending order.

//                TreeSet<Integer> t= new TreeSet<Integer>(arrayList);
                  System.out.print("Expenses in ascending order is : ");
//                for(Integer i :t)
//                {
//                System.out.print(i+" ");
//                }
//                System.println();
                 Collections.sort(arrayList);
                 for(Integer i:arrayList)
                 {
                        System.out.print(i+" ");
                 }
                System.out.println();

        }

}
```