```
package stack_project;
import java.util.*;
import java.io.*;
public class stack_project {

  public static void intro() { // a guidance to the user

    System.out.println("Please Enter:");
    System.out.println(" *(push) to add an element to the stack.");
    System.out.println(" *(peek) to return the value of the element on the top of the stack.");
    System.out.println(" *(pop) removes and returns the element on the top of the stack.");
    System.out.println(" *(size) to get the number of elements in the stack.");
    System.out.println(" *(empty) to know if the stack empty or not.");
    System.out.println(" *(show) to show the current stack.");
    System.out.println(" *(sort) to sort the stack.");
    System.out.println(" *(stop) to stop.");
    System.out.println("Enter an operation!");
  }
```

A method
to guide the user

```
  public static void string(String input) {
```

A method
That get the user
input and make
operations

```
         Scanner in = new Scanner(System.in);
         stack s = new stack();                 // the main stack
         int n=0;                               // Initial value
         String word = input;                   // to get the input value and
         while (!word.contains("stop"))  // to get the end of the program

{
              String ss;
         if(word.equals("push")) {
```

To push an element

```
                   if(s.top==s.capacity) {              // if the stack is full
                        System.out.println("the stuck is overload!");
                        System.out.println("please enter (pop) to remove last element or");

                   }
                   else {
                   System.out.println("Enter a number ");
                   try {
                   n=in.nextInt();
                     s.push(n);  }

                   // if there any error
                   catch (Exception e)
                   {System.out.println("Please enter a number correctly");
                    }
                    }
                             }

         else if (word.equals("peek")) {
```

To peek the element

```
         try { System.out.println(s.peek()); }
              catch (Exception e)  {
                   System.out.println("the stack is empty!");
```

```
                    }
    }
    else if(word.equals("pop")) {  // pop
            if(s.top==0) { // if the stack is empty
                System.out.println("the stuck is empty");
                System.out.println("please enter (push) to add an element or");
            }
            else
            System.out.println(s.pop()); // to pop the element
                }
```

To pop the element

```
        // size of the main stack
    else if(word.equals("size")) {
            System.out.println(s.getsize());
                }
```

To know the size

```
        // is the stack empty or not
    else if(word.equals("empty")) {
            if(s.isEmpty()) {
                System.out.println("Yes");  // instead of true
            }
            else if(s.isEmpty()==false) {
                System.out.println("No"); // instead of false
            }
                    }
```

If the stack is empty or not

```
        // sorting the stack that takes "the poped elements" from the main stack
    else if(word.equals("sort")) {
            s.sortAscending();

                }
        // show the main stack elements by the entrance priority
```

To sort the poped elements

```
    else if(word.equals("show")) {
            s.show();
            System.out.println();
                    }
```

To show the current stack

```
        // if the operation entered un-correctly

    else {
            System.out.println("Enter the operation correctly");
        }
System.out.println("-------------------------");
 System.out.println("enter an operation!");
System.out.println("-------------------------");

    ss =in.next();
     word = ss.toLowerCase();        // to keep the loop working
 }   // while curly bracket ending

    // after finishing while loop
    System.out.println("Thank you!");
 }      // the method curly bracket ending
```

Stack Class

```
package stack_project;

public class stack {
    public int capacity = 10; // public because if you want to change the capacity
    int [] stack  = new int [capacity]; // main stack
    public int top =0;
    int [] stack1 = new int [capacity]; // 1st one
    private int top1 =0;

    public stack() {
        int i=0;
        for(i =0 ; i <capacity;i++) {
            stack[i]= i+1; // default stack
        }
    }
        public void push(int n) {                                   Push method
                stack[top]=n;
                top++;
        }
    public int pop() {                                              pop method
            int n;
            top--; // because top > stack capacity by 1
            n = stack[top];
                //for first stack *not the main stack *
            stack1[top1] = stack[top]; // to be added on 2nd stack
                top1++; // top of 2nd stack
                stack[top]=0;
            return n;
        }
    public int peek() {                                             peek method
            int n;
            n = stack[top-1];
            return n;
        }

    public int getsize() {                                          getSize method
            return top;
        }
    public boolean isEmpty() {                                      isEmpty method
            return top<=0;
        }
    public void show() {                                            show method
            if(top==0){                                             (extra one)
                System.out.println("the stack is empty!");
            }
            for(int i =0; i<top;i++) {
                System.out.print(stack[i]+" ");
            }
        }
    public void sortAscending() {            →            sortAscending method
            // sorted stack
            if(top1==0){
                System.out.println("the stack is empty!");
                // beacase the user
                //didn't pop any element from the main Stack
            }
            else {
                boolean r = true;
                int value;
                while(r) {
                    r= false; // to end the while loop
                    for (int i=0; i < (top1-1); i++) {
                        if(stack1[i]>stack1[i+1]) {
                            value = stack1[i];
                            stack1[i]=stack1[i+1];
                            stack1[i+1]=value;
                            r = true;
                            } // if end bracket
                        } // for end bracket
                        } // while end bracket
                for(int i =0; i<top1;i++) {
                    System.out.print(stack1[i]+" ");
                        }
                System.out.println();
                } // else curly bracket end
        } // method curly bracket end
} // class curly bracket end
```

Main Class

```
 public static void main(String[] args) {
        intro(); // just a guide to the user
        Scanner in = new Scanner(System.in);


        String input = in.next(); // user input
        input.trim(); // if there are spaces
        string(input.toLowerCase());  // to lower to be compared with if conditions
   }
}
```

ID	الاسم
20181496699	يوسف محمد محمود عبد الله
20181495812	أحمد سليمان عبد الرحيم

Stack project