IT 314 – Software Engineering

Lab 7

Akshat Jindal | 202201299

Stack Implementation Code

Program Inspection

Question 1. How many errors are there in the program? Mention the errors you have identified.

Once we figure out the error in the printing of the stack in the for loop, and correct the condition, we can simply correct the logic of the stack implementation, namely the top variable of the stack.

Question 2. Which category of program inspection would you find more effective?

For this question, the Category A of the program inspection, the Data Reference Errors were the most effective, as the failure of the code was due to the wrong access of the stack data, due to the errors in calculating the top variable value.

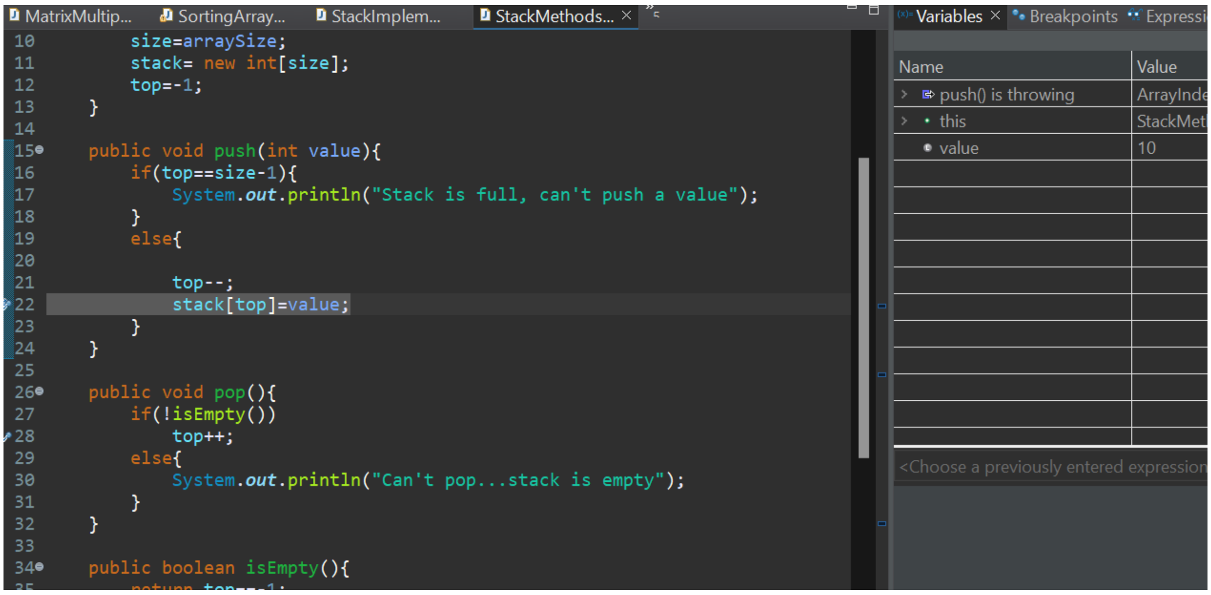
Question 3. Which type of error you are not able to identified using the program inspection?

In this example, all the errors in the document were identifiable using the program inspection method.

Question 4. Is the program inspection technique is worth applicable?

Due to a slightly longer code, with many functions at that, it was a bit complicated to find the errors using program inspection method.

Code Debugging



```java
        size=arraySize;
        stack= new int[size];
        top=-1;
    }

    public void push(int value){
        if(top==size-1){
            System.out.println("Stack is full, can't push a value");
        }
        else{

            top--;
            stack[top]=value;
        }
    }

    public void pop(){
        if(!isEmpty())
            top++;
        else{
            System.out.println("Can't pop...stack is empty");
        }
    }

    public boolean isEmpty(){
```

Name	Value
push() is throwing	ArrayInde
this	StackMet
value	10

<Choose a previously entered expression

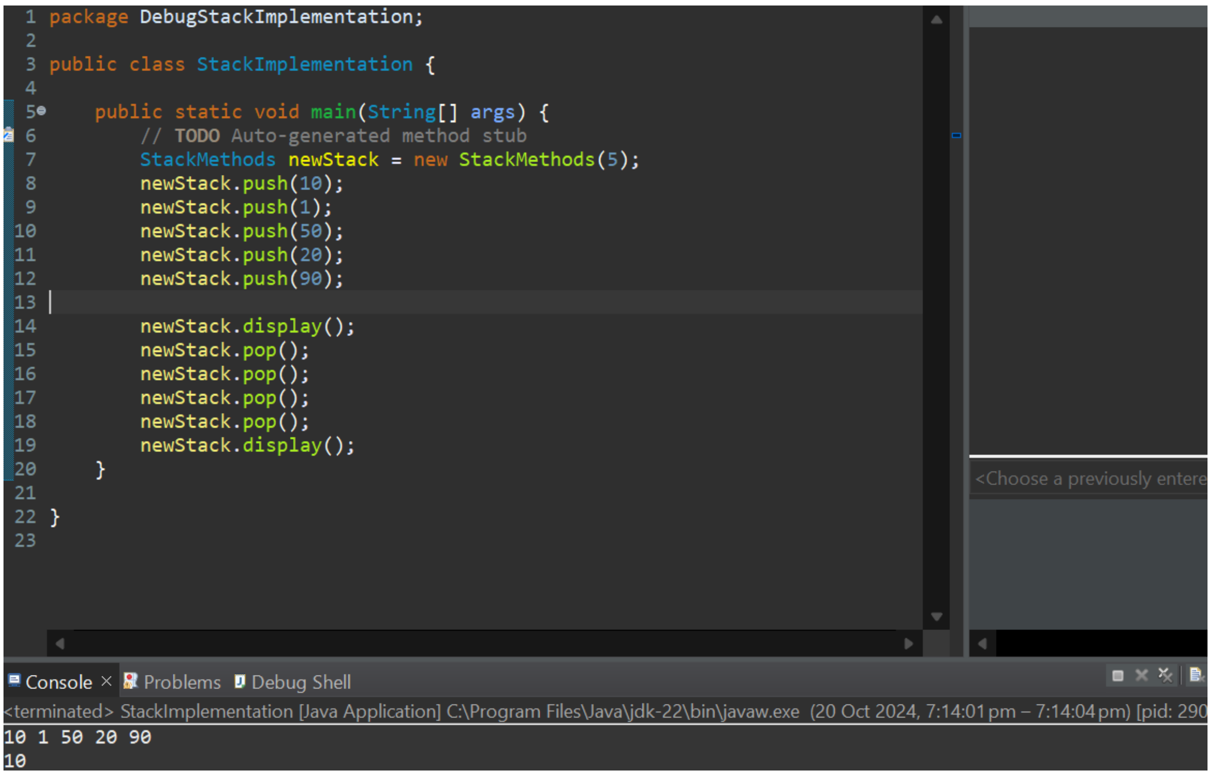
Question 1. How many errors are there in the program? Mention the errors you have identified.

There were 2 errors in this program. The for loop condition at the end for printing, as well as the changes in the top variable when implementing stack pop or push.

Question 2. How many breakpoints you need to fix those errors? What are the steps you have taken to fix the error you identified in the code fragment?

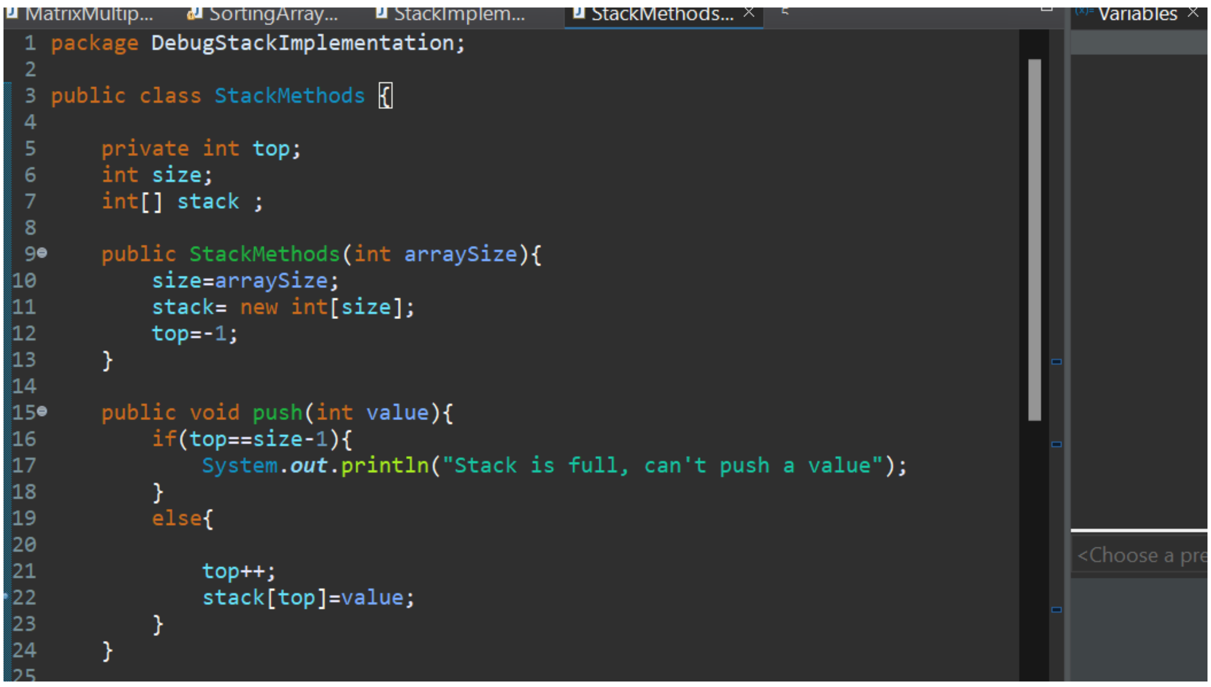
With the line breaks at the end of the push, pop and in the loop of the display functions, we can iteratively observe the values of top, and how the referenced value changes. Once we figure out the calculation error, we can simply update the top variable correctly.

Question 3. Submit your complete executable code.

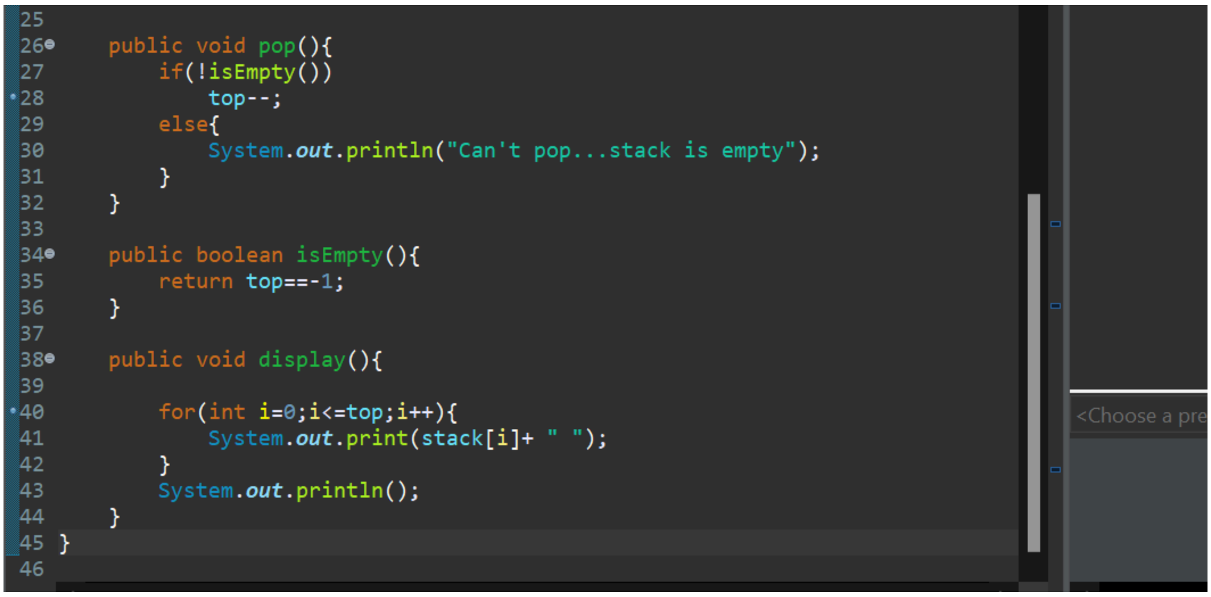


```java
package DebugStackImplementation;

public class StackImplementation {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        StackMethods newStack = new StackMethods(5);
        newStack.push(10);
        newStack.push(1);
        newStack.push(50);
        newStack.push(20);
        newStack.push(90);

        newStack.display();
        newStack.pop();
        newStack.pop();
        newStack.pop();
        newStack.pop();
        newStack.display();
    }

}
```

```
<terminated> StackImplementation [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe  (20 Oct 2024, 7:14:01 pm – 7:14:04 pm) [pid: 290
10 1 50 20 90
10
```



```java
package DebugStackImplementation;

public class StackMethods {

    private int top;
    int size;
    int[] stack ;

    public StackMethods(int arraySize){
        size=arraySize;
        stack= new int[size];
        top=-1;
    }

    public void push(int value){
        if(top==size-1){
            System.out.println("Stack is full, can't push a value");
        }
        else{

            top++;
            stack[top]=value;
        }
    }
```



```

    public void pop(){
        if(!isEmpty())
            top--;
        else{
            System.out.println("Can't pop...stack is empty");
        }
    }

    public boolean isEmpty(){
        return top==-1;
    }

    public void display(){

        for(int i=0;i<=top;i++){
            System.out.print(stack[i]+ " ");
        }
        System.out.println();
    }
}

```