

## DSA Week 8 activities

This week, you are required to complete two questionnaires and two labs.

- a.** In your DSA textbook 1 page 141, answer questions 2 and 5.
- b.** In this print out, answer all Week 8 questions.
- c.** Also, in this print out, complete Week 8 lab 1 & 2 using the lab computers.

**Note:** You can complete the activities in any order, however, make afford to complete and understand everything which prepares you for well for test 2 & Final Exam.

## DSA Week 8 Questions

1. What is a Stack?
  
  
  
  
  
  
  
  
  
  
2. Discuss the five methods used with stacks.
  
  
  
  
  
  
  
  
  
  
3. A stack has the elements (10, 28, 31).
  - a. What happens to the stack when you push (30)?
  - b. What happens to the stack when you pop(30), pop(31) and pop(28)?
  - c. After performing the methods above, which elements are still in the stack?
  - d. After the three step above, if you run the method isEmpty() will it return true or false? Explain why it would return true or false.
  
  
  
  
  
  
  
  
  
  
4. Complete the sentence, Stacks LIFO are implemented through \_\_\_\_\_ and \_\_\_\_\_ data structures.
  
  
  
  
  
  
  
  
  
  
5. Referring to programming, explain the concept of generics.

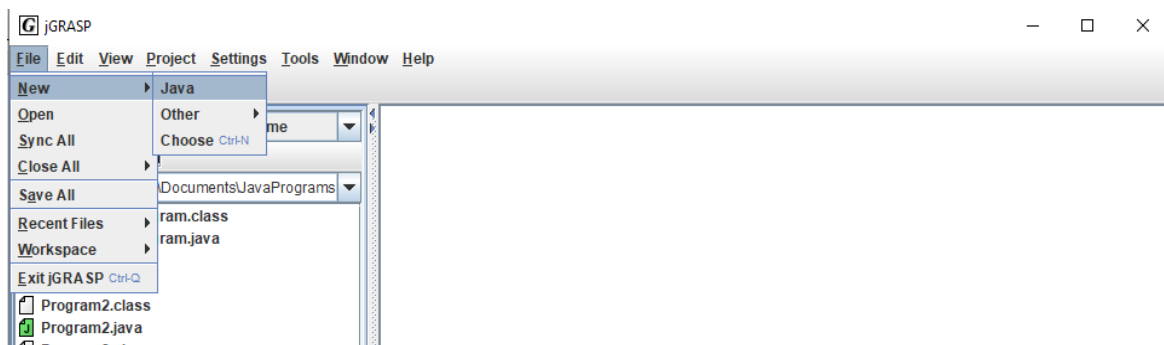
6. What is the main reason for using generic code or syntax compared to a non-generic code?

7. What is the LIFO principle?

## DSA Week 8 Lab Activity (Week8Lab1)

Using the lab computers create the following Java program using jGrasp!

**Step 1:** Login to your lab computer and create a new java file in jGrasp.



**Step 2:** When the window below appears. Type the following code into jGrasp.

```
/* DSA Week 8 Lab 1 */

import java.util.Stack;

public class Week8Lab1 {

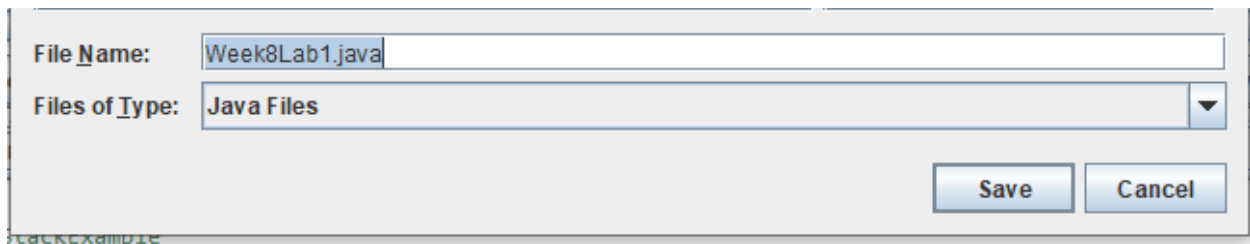
    public static void main(String[] args) {
        // Create a stack object called itiCourses
        Stack<String> itiCourses = new Stack<>();

        // Push elements onto the stack
        itiCourses.push("DIT");
        itiCourses.push("DHRM");
        itiCourses.push("DACC");
        itiCourses.push("DICT");

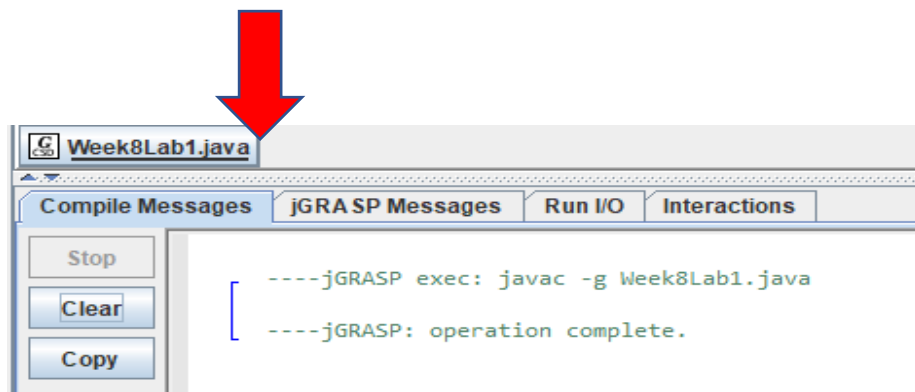
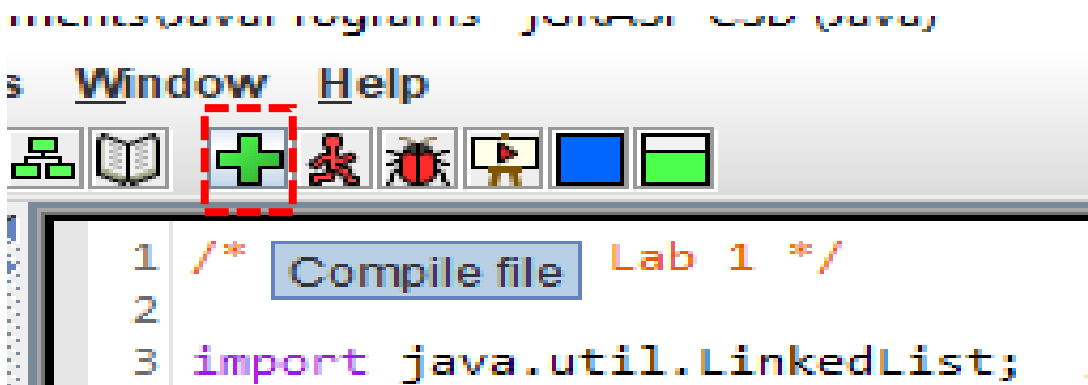
        // Peek or see the top element of the stack
        System.out.println("Top element: " + itiCourses.peek());

        //if stack is empty print message stack is empty, else print elements in the stack
        if(itiCourses.isEmpty()){
            System.out.print("Stack is empty");
        }else{
            // Print the stack (it will display elements in LIFO order)
            System.out.print("Stack contents: " + itiCourses);
        }
    }
}
```

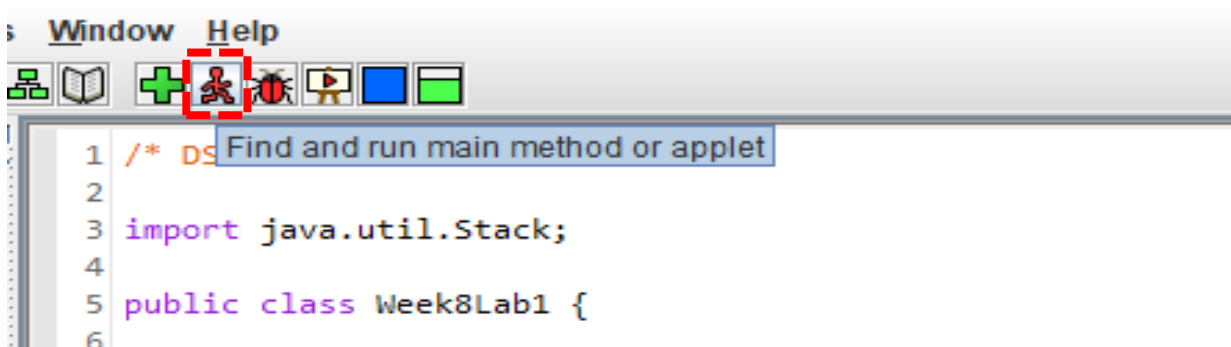
**Step 3:** Go to **file/save** to save your java program as **Week8Lab1**



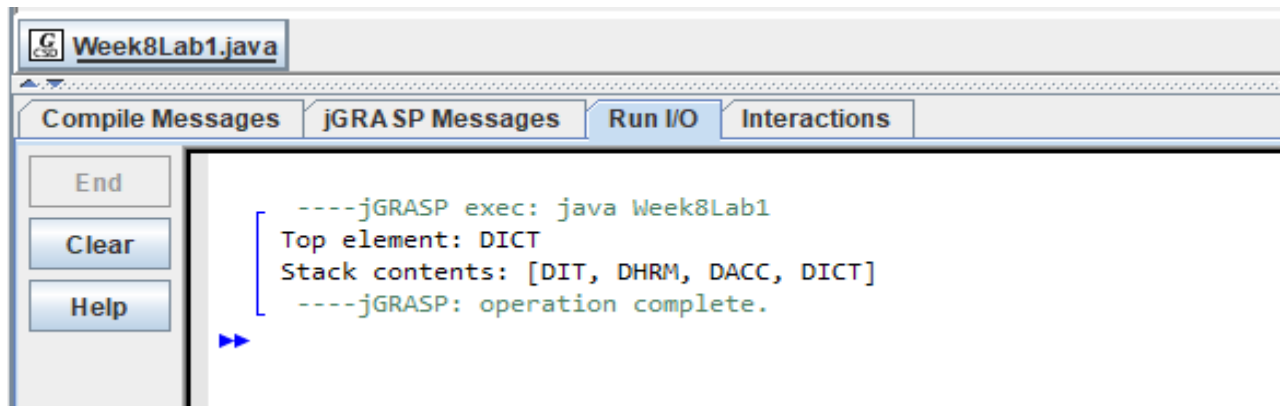
**Step 4:** After saving, compile (click on compile icon or on your keyword hold **Ctrl + B**) to check for syntax errors.



**Step 5:** If compiling is successfully then run (click on the find and run main method icon or on your keyboard hold **Ctrl + R**) your program.



**Step 6:** If run is successful then you should see the following output in the console

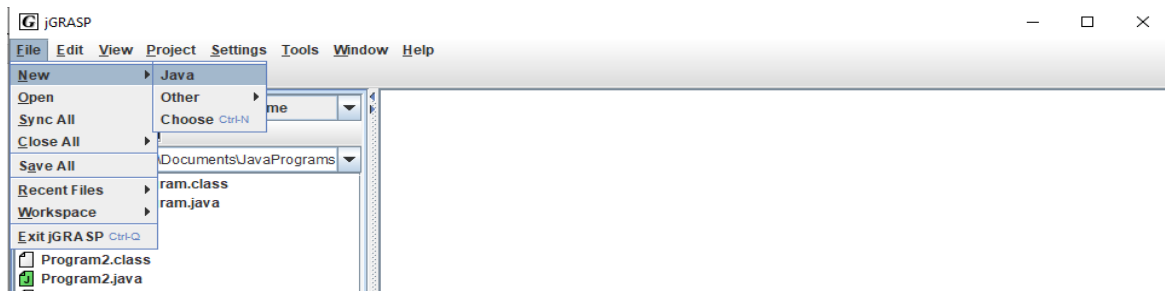


**Step 7:** Week8Lab1 Completed! Save your file for future Java lab activities.

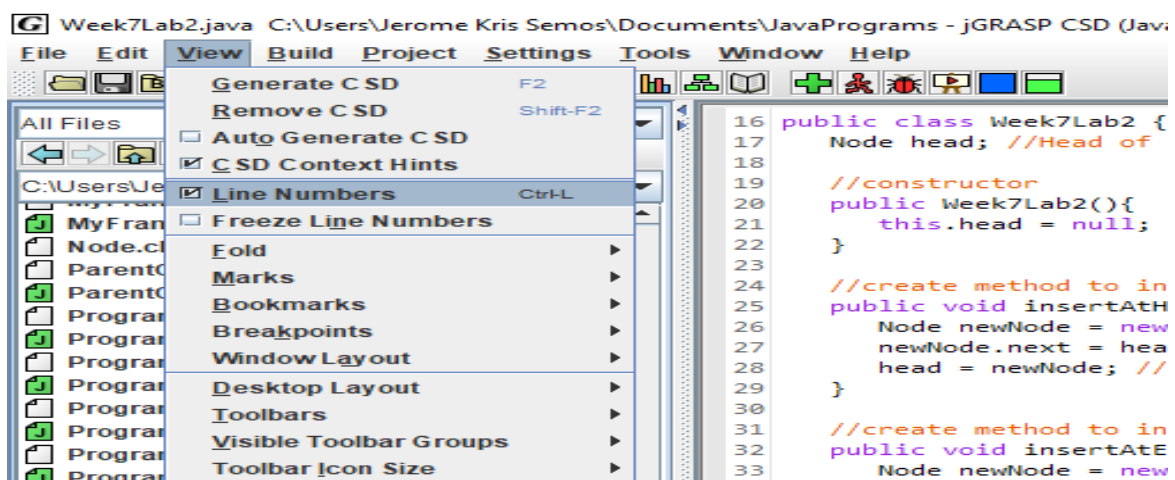
## DSA Week 8 Lab Activity (Week8Lab2)

Using the lab computers create the following Java program using jGrasp!

**Step 1:** Login to your lab computer and create a new java file in jGrasp.



**Step 2:** Switch on line numbers, Go to View\Line Numbers or hit Ctrl + L to enable line numbers.



**Step 3:** Now, type the following code into jGrasp. **Note:** this program has 78 lines of Java code, hence, we have to enable line numbers to assist our coding and resolve errors when coding, compiling or debugging.

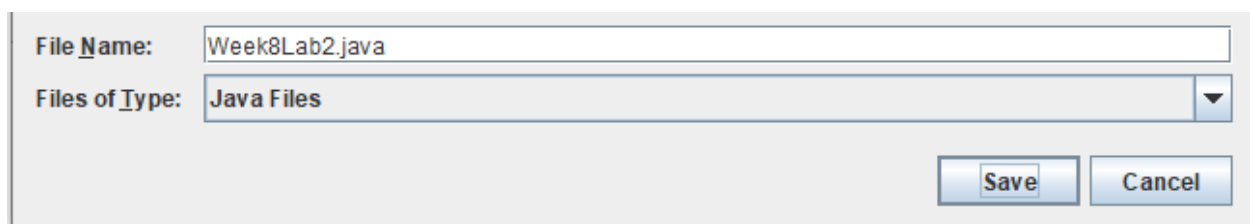
```
1  /* DSA Week 8 Lab 2 */
2
3  import java.util.Stack; //import the Stack class or Java Package
4
5  public class Week8Lab2<E> {
6
7      private Stack<E> stack;
8
9      // Constructor
10     public Week8Lab2() {
11         this.stack = new Stack<>();
12     }
13
14     // Push method
15     public void push(E item) {
16         stack.push(item);
17     }
18
19     // Peek method
20     public E peek() {
21         if (!stack.isEmpty()) {
22             return stack.peek();
23         } else {
24             System.out.println("Stack is empty.");
25             return null;
26         }
27     }
28
29     // Check if stack is empty
30     public boolean isEmpty() {
31         return stack.isEmpty();
32     }
33
34     // Print stack elements
35     public void printStack() {
36         if (stack.isEmpty()) {
37             System.out.println("Stack is empty.");
38         } else {
39             System.out.println("Stack contents:");
40             for (E item : stack) {
41                 System.out.println(item);
42             }
43         }
44     }
45 }
```

```

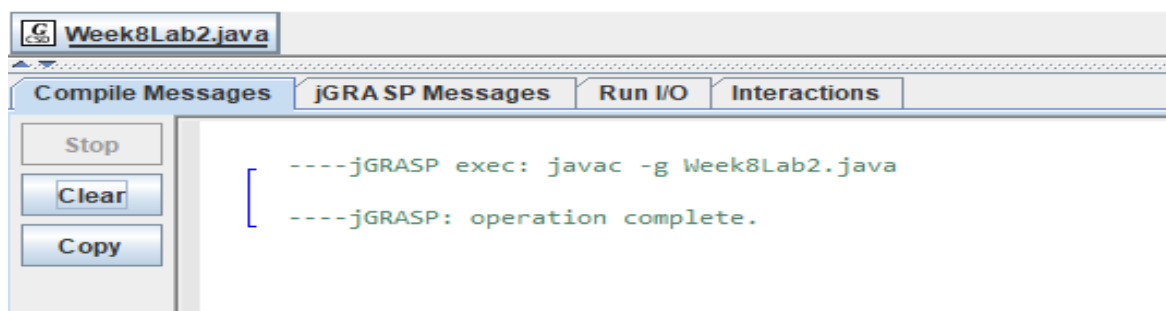
46     public static void main(String[] args) {
47         // Create a generic stack for Strings
48         Week8Lab2<String> stringStack = new Week8Lab2<>();
49         stringStack.push("DIT");
50         stringStack.push("DHRM");
51         stringStack.push("DACC");
52         stringStack.push("DICT");
53
54         System.out.println("String Stack:");
55         System.out.println("Top element: " + stringStack.peek());
56         stringStack.printStack();
57
58         // Create a generic stack for Integers
59         GenericStackExample<Integer> intStack = new GenericStackExample<>();
60         intStack.push(10);
61         intStack.push(20);
62         intStack.push(30);
63
64         System.out.println("\nInteger Stack:");
65         System.out.println("Top element: " + intStack.peek());
66         intStack.printStack();
67
68         // Create a generic stack for Doubles
69         GenericStackExample<Double> doubleStack = new GenericStackExample<>();
70         doubleStack.push(10.5);
71         doubleStack.push(20.75);
72         doubleStack.push(30.25);
73
74         System.out.println("\nDouble Stack:");
75         System.out.println("Top element: " + doubleStack.peek());
76         doubleStack.printStack();
77     }
78 }

```

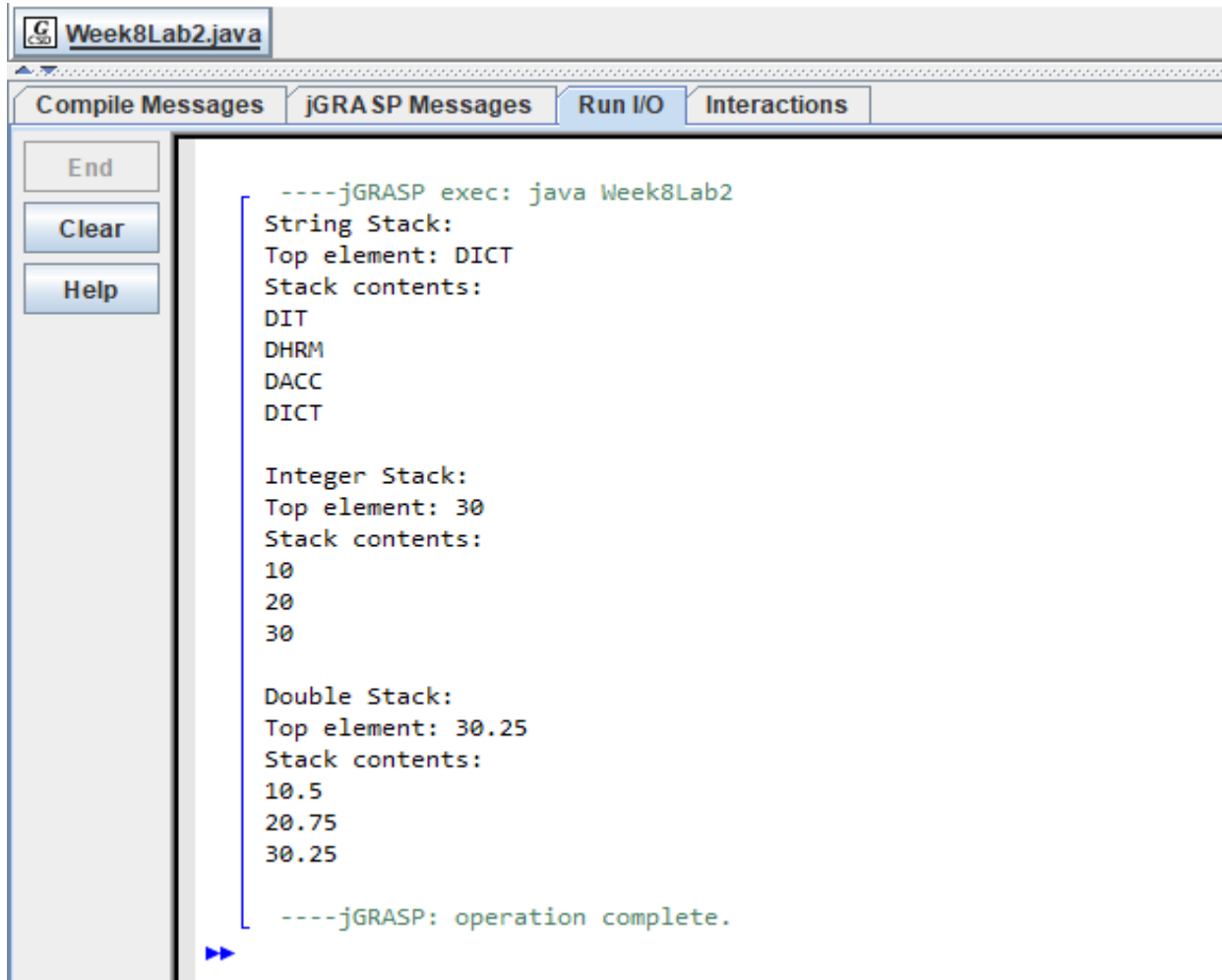
**Step 4:** After coding the program, go to **file/save** to save your java program as **Week8Lab2**



**Step 5:** After saving, compile (click on compile icon or on your keyword hold **Ctrl + B**) to check for syntax errors.



**Step 6:** If compiling is successfully then run (click on the find and run main method icon or on your keyboard hold **Ctrl + R**) your program.



```
Week8Lab2.java
Compile Messages  jGRASP Messages  Run I/O  Interactions

End
Clear
Help

----jGRASP exec: java Week8Lab2
String Stack:
Top element: DICT
Stack contents:
DIT
DHRM
DACC
DICT

Integer Stack:
Top element: 30
Stack contents:
10
20
30

Double Stack:
Top element: 30.25
Stack contents:
10.5
20.75
30.25

----jGRASP: operation complete.
```

**Step 7:** If successful your program should display an output like shown in the screenshot above.

**Step 8:** Week5Lab2 Completed! You have created your first generic Stack data structure program.

## Summary of application of stacks in the Real World

1. The undo/redo operations in Microsoft Word
2. Forward or backward loading of a website in a web browser
3. All messages or log of messages in a smartphone
4. Gallery or image library of a smartphone
5. Some music player – playlist which play previous or next song in the queue