DATA STRUCTURE AND ALGORITHM
LAB SHEET 5 - STACK ADT
AM.EN.U4CSE20349
PATEL RAJKUMAR PANKAJBBHAI

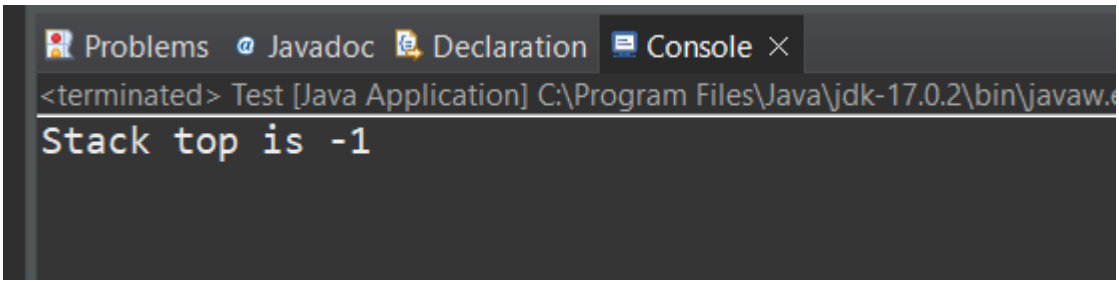
1. Declare a class (StackInt.java) for integer StackInt with two attributes: (a) An array 'arr' of size 5. (b) A variable 'top' initialized to -1;

```
package stack.ADT;

public class StackInt {
	int []arr = new int[5];
	int top = -1;
}
```

2. Write a separate test driver class (Test.java) and create an object instance of StackInt class. StackInt si = new StackInt(); Compile both the .java files and run Test.java. Ensure no errors.

```
package stack.ADT;

public class Test {

	public static void main(String[] args) {
		StackInt si = new StackInt();

		System.out.println("Stack top is "+si.top);

	}
}
```

3. Now try to access the 'top' attribute of StackInt from Test.java directly. System.out.println("Stack top is " + si.top); Compile and execute.



4. Add a default constructor that will create the stack of some standard top (say 10) in case user does not give the top. StackInt() { arr = new int[10]; top = -1; } Test cases: StackInt si = new StackInt(); System.out.println(si.arr.length);

```
package stack.ADT;

public class StackInt {
	int []arr;
	int top = -1;
```

```
        StackInt(){
                arr = new int[10];
                top = -1;
        }

}
```

StackInt.java Test.java

```java
package stack.ADT;

public class Test {

    public static void main(String[] args) {
        StackInt si = new StackInt();

        System.out.println("Stack top is "+si.top);
        System.out.println(si.arr.length);

    }

}
```

Problems Javadoc Declaration Console

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.
Stack top is -1
10
```

5. Add a parameterized constructor that will create the stack with specified size. StackInt(int sz) { arr = new int[sz]; top = -1; } Test cases: StackInt si2 = new StackInt(15); System.out.println(si2.arr.length);

StackInt.java × Test.java

```java
package stack.ADT;

public class StackInt {
    int []arr;
    int top;

    StackInt(){
        arr = new int[10];
        top = -1;
    }

    StackInt(int sz){
        arr = new int[sz];
        top = -1;
    }

}

```

StackInt.java Test.java ×

```java
package stack.ADT;

public class Test {

    public static void main(String[] args) {
        StackInt si = new StackInt();
        System.out.println("Stack top is "+si.top);
        System.out.println(si.arr.length);

        StackInt si2 = new StackInt(25);
        System.out.println(si2.arr.length);


    }

}

```

Problems Javadoc Declaration Console ×

<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe (07-Apr-2022, 7:56:57 pm – 7:56:57 pm

```
Stack top is -1
10
25
```

6. Add print function to StackInt.java that will print the contents up to the top of the stack. It should be public. should be public.

```java

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

```

7. Implement push() method first without checking the array length limit. public void push(int item) { // Your logic here // increment top and set 'top'th position in 'arr' to item }
Test cases:
(a) Try invoking push operations and check. StackInt si = new StackInt(); si.push(100); si.print(); si.push(200); si.print();
(b)Try to push beyond stack capacity StackInt si = new StackInt(); ..... si.push(900); si.print(); si.push(300); si.print();
The execution will abort as soon as the last push is invoked. ArrayIndexOutOfBoundsException.

```java

    public void push(int item) {

        arr[++top] = item;
    }

```

StackInt.java Test.java

```java
package stack.ADT;

public class Test {

    public static void main(String[] args) {
        StackInt si = new StackInt();

        si.push(100);
        si.push(200);
        si.push(300);
        si.push(400);
        si.push(500);
        si.push(600);
        si.push(700);
        si.push(800);
        si.push(900);
        si.push(100);
        si.print();
        si.push(101);

    }

}
```

Problems Javadoc Declaration Console

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe  (07-Apr-2022, 8:25:26 pm – 8:25:27 pm)
100 200 300 400 500 600 700 800 900 100
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 10 out of bounds for length 10
        at Stack.ADT/stack.ADT.StackInt.push(StackInt.java:27)
        at Stack.ADT/stack.ADT.Test.main(Test.java:19)
```

8. Implement the check to push() method. Don't add an item to the array if top exceeds arr.length. Instead print "can't push" message. public void push(int item) { // Your enhanced logic here } Test cases: (a) Run this time. No exception will be thrown this time around.

```java

    public void push(int item) {

        if(top == arr.length - 1) {
            System.out.println("Stack OverFlow");
            return;
        }
        arr[++top] = item;
    }

```

StackInt.java Test.java

```java
package stack.ADT;

public class Test {

    public static void main(String[] args) {
        StackInt si = new StackInt();

        si.push(100); si.push(200); si.push(300); si.push(400);
        si.push(500); si.push(600); si.push(700); si.push(800);
        si.push(900); si.push(100);
        si.print();
        si.push(101);
        si.push(102);

    }

}
```

Problems Javadoc Declaration Console

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe  (07-Apr-2022, 8:36:00 pm – 8:36:00 pm)
100 200 300 400 500 600 700 800 900 100
Stack OverFlow
Stack OverFlow
```

9. Add a getter method getTop() which returns the current top of the stack. public int getTop() { // return top; } Test cases: (a) Invoke getTop() from test file and print the top. System.out.println(si2.getTop());

```java
    public int getTop() {

        return arr[top];
    }

```

StackInt.java Test.java

```java
package stack.ADT;

public class Test {

    public static void main(String[] args) {
        StackInt si = new StackInt();

        si.push(100); si.push(200); si.push(300); si.push(400);

        System.out.print("Top element = "+si.getTop());

    }

}

```

Problems Javadoc Declaration Console

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe  (07-Apr-2022, 8:42:13 pm – 8:42:13 pm)
Top element = 400
```

10. Now implement pop() method that removes the topmost item in the stack and returns it. First without lower bound checking logic. public int pop() { // Your logic here } Test cases: (a) Write push and pop few items to check it's working. int item = si.pop(); si.print(); (b) Write pop more items than what were pushed. int item1 = si.pop(); si.print(); .... The last call to pop should throw Array out of bounds exception.

```java
    public int pop() {

        return arr[top--];
    }

```

StackInt.java | Test.java

```java
package stack.ADT;

public class Test {

    public static void main(String[] args) {
        StackInt si = new StackInt();

        si.push(100); si.push(200); si.push(300); si.push(400);
        si.print();

        int poppedItem = si.pop();
        System.out.println("Popped item = "+poppedItem);
        si.print();

```

Problems | Javadoc | Declaration | Console

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe  (08-Apr-2022, 8:52:58 am – 8:52:58
100 200 300 400
Popped item = 400
100 200 300
```

StackInt.java | *Test.java

```java
public class Test {

    public static void main(String[] args) {
        StackInt si = new StackInt();

        si.push(100); si.push(200); si.push(300); si.push(400);
        si.print();

        si.pop();
        si.pop();
        si.pop();
        si.pop();
        si.print();
        si.pop();

```

Problems | Javadoc | Declaration | Console

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe  (08-Apr-2022, 8:55:28 am – 8:55:29 am)
100 200 300 400
The stack is empty

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index -1 out of bounds for length 10
        at Stack.ADT/stack.ADT.StackInt.pop(StackInt.java:45)
        at Stack.ADT/stack.ADT.Test.main(Test.java:16)
```

11. Now implement the check for lower limit (< 0) and print "can't pop" message. Run test file. Note that no exception will be thrown this time.

```java
    public int pop() {

        if(top == -1) {
            System.out.println("Can't PoP");
            return -1;
        }
        return arr[top--];
    }

```

StackInt.java Test.java

```java
    public static void main(String[] args) {
        StackInt si = new StackInt();

        si.push(100); si.push(200); si.push(300); si.push(400);
        si.print();

        si.pop();
        si.pop();
        si.pop();
        si.pop();
        si.print();
        si.pop();

    }
}
```

Problems Javadoc Declaration Console

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe  (08-Apr-2022, 8:59:06 am – 8:59:07 ar
100 200 300 400
The stack is empty

Can't PoP
```

12. Implement peek() method to return topmost item without removing it from the stack public int peek() { // Your logic here } Test cases: (a) Write test file to check the working of peek. This also needs <0 check. StackInt si = newStackInt[5]; System.out.println(si.peek()); si.push(100); System.out.println(si.peek()); si.push(200);
tem.out.println(si.peek()); si.push(300); System.out.println(si.peek());

```java
    public int peek() {
        if(top == -1) {
            System.out.println("Can't Peek");
            return -1;
        }
        else {
            return arr[top];
        }
    }
```

StackInt.java Test.java

```java
package stack.ADT;

public class Test {

    public static void main(String[] args) {
        StackInt si = new StackInt(5);
        System.out.println(si.peek());
        si.push(100);
        System.out.println(si.peek()); si.push(200);
        System.out.println(si.peek()); si.push(300);
        System.out.println(si.peek());

    }
}
```

Problems Javadoc Declaration Console

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe  (08-Apr-2022, 9:08:04 am – 9:08
Can't Peek
-1
100
200
300
```

13. You can't check if contents of two stacks are same by using = =. Test cases: StackInt si1 = new StackInt(); StackInt si2 = new StackInt(); si1.push(100); si2.push(100); si1.push(200); si2.push(200);if (si1 == si2) System.out.println("Both si1 and si2 are same"); else System.out.println("Both si1 and si2 are not the same"); Run it and check. It will print si1 and si2 are not same. Why? Because = = operator will only compare 2 addresses. Then how to check the contents?

```
StackInt.java    Test.java ×
3  public class Test {
4
5      public static void main(String[] args) {
6          StackInt si1 = new StackInt();
7          StackInt si2 = new StackInt();
8          si1.push(100);      si2.push(100);
9          si1.push(200);      si2.push(200);
10
11         if (si1 == si2)
12             System.out.println("Both si1 and si2 are same");
13         else
14             System.out.println("Both si1 and si2 are not the same");
15
16     }
17 }
18
<
Problems  Javadoc  Declaration  Console ×
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe  (08-Apr-2022, 9:12:58 am – 9:12:58 am)
Both si1 and si2 are not the same
```

Although both the stack have same elements but still we get the output “Both si1 and si2 are not same because the == operator compares the address of both the stacks and not their respective content or data.

How will we compare for equality of two stacks in java ?

Compare the size of both the stacks , if not equal then both stacks are not same.
Now if size are same then compare the top most element for equality if it same then pop them and compare the next immediate elements and keep doing this untill the stack gets empty.

14. Implement equals() method which will first compare the top. If not same, return false. If same, scan through arrays of both stacks to check if each item is one stack is same as an item in another stack. If so, return true. Else return false. public boolean equals(Stack another) { // Your logic here } Test cases: (a) Now run test file . It will print si1 and si2 are same since top and contents are same. (b) Now write test file to do same number of pushes but contents different. StackInt si1 = new StackInt(5); StackInt si2 = new StackInt(5); si1.push(100); si2.push(100); si1.push(200); si2.push(300);
(code) System.out.println("Both si1 and si2 are same"); else System.out.println("Both si1 and si2 are not the same");

*StackInt.java × Test.java

```java
    public boolean equals(StackInt si2) {
        boolean result = true;

        if(this.top != si2.top) {
            return false;
        }
        else {
            while(top != -1) {
                if(this.arr[top] != si2.arr[top]) {
                    return false;
                }
                top--;
            }
            return true;
        }

    }
```

StackInt.java Test.java ×

```java
package stack.ADT;

public class Test {

    public static void main(String[] args) {
        StackInt si1 = new StackInt();
        StackInt si2 = new StackInt();
        si1.push(100);      si2.push(100);
        si1.push(200);      si2.push(200);

        if (si1.equals(si2))
            System.out.println("Both si1 and si2 are same");
        else
            System.out.println("Both si1 and si2 are not the same");

    }
}
```

Problems Javadoc Declaration Console ×

<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe (08-Apr-2022, 9:28:22 am – 9:28:25 am)

```
Both si1 and si2 are same
```

15. Implement a function getminElement() to return the minimum element in a stack.

```java
    public int getMinElement() {

        int Min = 99999999;
        if(top == -1) {
            System.out.println("The stack is empty");
            return -1;
        }
        else {
            while(top != -1) {
                if(this.peek() < Min ) {
                    Min = this.pop();
                    top--;
                }
            }
            return Min;
        }
```

StackInt.java Test.java

```java
package stack.ADT;

public class Test {

    public static void main(String[] args) {
        StackInt si = new StackInt();
        si.push(100);      si.push(300);
        si.push(200);      si.push(400);

        System.out.println("The min element is: "+si.getMinElement());

    }
}
```

Problems Javadoc Declaration Console

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe  (08-Apr-2022, 6:46:37 pm – 6:46:37 pm)
The min element is: 300
```

16. Implement a copyStack() function to return a duplicate stack of original stack.

StackInt.java × Test.java

```java
    public StackInt copyStack() {
        StackInt newStack = new StackInt(this.arr.length);
        StackInt duplicate = new StackInt(this.arr.length);

        while(!this.isEmpty()) {
            newStack.push(this.peek());
            duplicate.push(this.pop());
        }

        while(!duplicate.isEmpty()) {
            this.push(duplicate.pop());

        }

        while(!newStack.isEmpty()) {
            duplicate.push(newStack.pop());

        }

        return duplicate;

    }
```

StackInt.java Test.java

```java
package stack.ADT;

public class Test {

    public static void main(String[] args) {
        StackInt si = new StackInt();
        si.push(300);       si.push(200);
        si.push(100);       si.push(700);
        si.push(600);

        System.out.println("Orginal Stack: ");
        si.print();

        System.out.println("Duplicate Stack: ");
        StackInt duplicateStack = si.copyStack();
        duplicateStack.print();

    }
}
```

Problems Javadoc Declaration Console

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe  (08-Apr-2022, 8:50:34 pm – 8:5
Orginal Stack:
300 200 100 700 600
Duplicate Stack:
300 200 100 700 600
```

17. Given a stack with push(), pop(), empty() operations, delete the middle element in the stack. Input : Stack[] = [1, 2, 3, 4, 5] Output : Stack[] = [1, 2, 4, 5] Input : Stack[] = [1, 2, 3, 4, 5, 6] Output : Stack[] = [1, 2, 4, 5, 6]

```java
    public int deleteMiddle() {
        StackInt temp = new StackInt(top);
        int middle = top/2;

        while(this.top > middle) {
            temp.push(this.pop());
        }
        int deletedElement = this.pop();

        while(temp.top != -1) {
            this.push(temp.pop());
        }

        return deletedElement;

    }
```

*StackInt.java Test.java

```java
package stack.ADT;

public class Test {

    public static void main(String[] args) {
        StackInt si = new StackInt();
        si.push(100);      si.push(300);
        si.push(200);      si.push(400);
        si.push(500);

        si.print();

        System.out.println("The deleted element is: "+si.deleteMiddle());
        si.print();

    }
}
```

Problems Javadoc Declaration Console

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe  (08-Apr-2022, 6:59:15 pm – 6:59:16 pm)
100 300 200 400 500
The deleted element is: 200
100 300 400 500
```

18. Implement a function to sort the elements in a stack.

```java
    public StackInt sort() {
        StackInt tempStack = new StackInt(this.arr.length);
        while(!(this.isEmpty())){
            if(tempStack.isEmpty()) {
                tempStack.push(this.pop());
            }
            else {
                int element = this.pop();
                while(tempStack.top !=-1 && (tempStack.peek() > element)) {
                    this.push(tempStack.pop());
                }
                tempStack.push(element);
            }
        }

        return tempStack;
    }
```

StackInt.java Test.java

```java
package stack.ADT;

public class Test {

    public static void main(String[] args) {
        StackInt si = new StackInt();
        si.push(300);      si.push(200);
        si.push(100);      si.push(700);
        si.push(600);

        System.out.println("Stack before sorting: ");
        si.print();

        System.out.println("Stack after sorting: ");
        StackInt sortedStack = si.sort();
        sortedStack.print();

    }
}
```

Problems Javadoc Declaration Console

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe  (08-Apr-2022, 8:31:44 pm – 8:31:44 p
Stack before sorting:
300 200 100 700 600
Stack after sorting:
100 200 300 600 700
```

19. Implement Two Stacks in a Single Array

```
package twoStack;

public class TwoStack {
	int []arr;
	int top1,top2;

	TwoStack(int sz){
		arr = new int[sz*2];
		top1 = top2 = -1;
	}

	public void push1(int ele) {
		if(top1 < arr.length/2) arr[++top1] = ele;
	}

	public void push2(int ele) {
		if(top2 == -1) top2 = 4;
		if(top2 < arr.length) arr[++top2] = ele;
	}

	public void print1() {
		while(top1 != -1) {
			System.out.print(arr[top1--]+" ");
		}
		System.out.println();
	}

	public void print2() {
		while(top2 != 4) {
			System.out.print(arr[top2--]+" ");
		}
		top2 = -1;
		System.out.println();
	}
}

package twoStack;

public class Test {

	public static void main(String[] args) {
		TwoStack ts = new TwoStack(5);

		ts.push1(1);
		ts.push1(3);
		ts.push1(5);
		System.out.print("Printing Stack1: ");
		ts.print1();

		ts.push2(0);
		ts.push2(2);
```

```
        ts.push2(4);
        ts.push2(6);
        System.out.print("Printing Stack2: ");
        ts.print2();

    }

}
```