Caleb Millard

CMPT 360 Spring 2023

Assignment 3

ADT data structure implementation and use

Assignment	Due Date	group(s)	Language	Language	Platform
1	Monday, Jan 23	1 & 2	Java	Delphi	Windows
2	Monday, Feb 6	1 & 2	C#	Visual basic	Windows
3	Monday, Feb 27th	3	Javascript		Windows
4					
5					
6					
7					

This assignment fulfills the following goals:
Group 3 Language: JavaScript(Node.Js)
Title:
Create and implement an ADT data structure and apply it to a program

Problem:
Learn a new language and create an ADT data structure, then put it into an application

Documentation:

Run the program
will output the average of the five grades

Imports used:

Imports:
Math Library

Pseudo Code:

```
Get input
Randomizer(not a real one just from Math library) to prove the concept of running without specific input.
Multiply by 100 and add 1 to make the numbers 1- 100 and use the floor to make sure they are whole numbers
Then push these into the stack for storing

After this create a running total by popping the numbers out of the stack one at a time and then dividing by the number of items popped

Output the average.
```

Variables used in program

Integers:	number, i, total, peeknum
Stacks:	newstack

JavaScript Program Start:

```
//Caleb Millard
//613362
//Title: ADT data structure Lab
//Lab 3 CMPT 360
//Dr. Rick Sutcliffe


//This ADT was fairly simple to create since javascript already has
similar actions within its array manipulation.
//the key different is that we have to make the data type immutable to
outside influence other than by the methods we provide
//so you need to be able to see what is inside but not be able to
manipulate what is within the stack, and since this is its own ADT
//you need to use the methods to access
class Stack {
    constructor() {
        this.stack = [];
    }
    push(item) {
        //pushes input item into the stack
        this.stack.push(item);
    }
    pop() {
        //removes and returns the last entered item in the stack
        number = this.stack.pop();
        return number
    }
    peek() {
        //returns the last item on the stack without removing it
        peeknum = this[this.length()];
        return peeknum
    }
}
const newstack = new Stack();
//creating a stack data type to use

var i = 0;
```

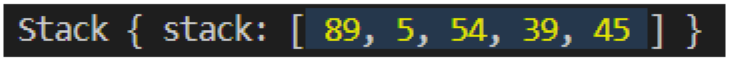
```
while (i < 5) {
    var number = Math.floor(Math.random() * 100 + 1);
    newstack.push(number);
    i++;
    //this inputs 5 inputs into the stack randomly from between 1-100
}
console.log(newstack);
i = 0;
var total = 0;
number = 0;

for (var i = 0; i < 5; i++) {
    //calculates the average of the 5 numbers from the stack
    number = newstack.pop();
    total += number;
}
total = total / 5;
console.log("your Average Grade is " + total);
```

End of required code

Input:

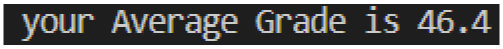
89
5
54
39
45



```
Stack { stack: [ 89, 5, 54, 39, 45 ] }
```

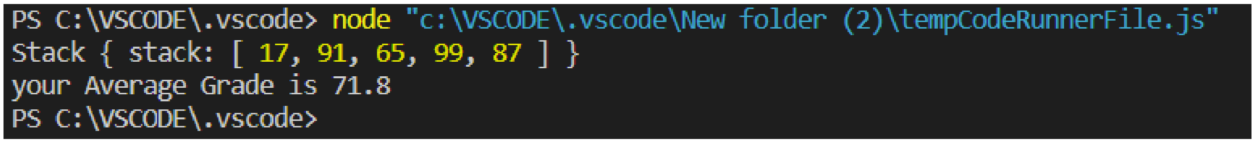
Output:

Your Average Grade is 46.4



```
your Average Grade is 46.4
```

Screenshots:



```
PS C:\VSCODE\.vscode> node "c:\VSCODE\.vscode\New folder (2)\tempCodeRunnerFile.js"
Stack { stack: [ 17, 91, 65, 99, 87 ] }
your Average Grade is 71.8
PS C:\VSCODE\.vscode>
```

Conclusions:

Overall this assignment was decent for me since Javascript already included some of the base methods that are needed to create a stack I was able to create a proper stack. While it looks very simple, the main issue with javascript and creating a stack is that the arrays function similarly to stacks, with the main difference being that more than just the last entered element is mutable which is not ok for a stack. So I created a separate class that restricts the mutability. And along with this to test its functionality I used the math random library to prove it works with lower and higher numbers, not just high numbers. I think creating a data structure would have been significantly more difficult if I had used another language in this lab since javascript is very similar to Java.