Implement stack with max and min function

Posted on September 10, 2019 | by Prashant Yadav

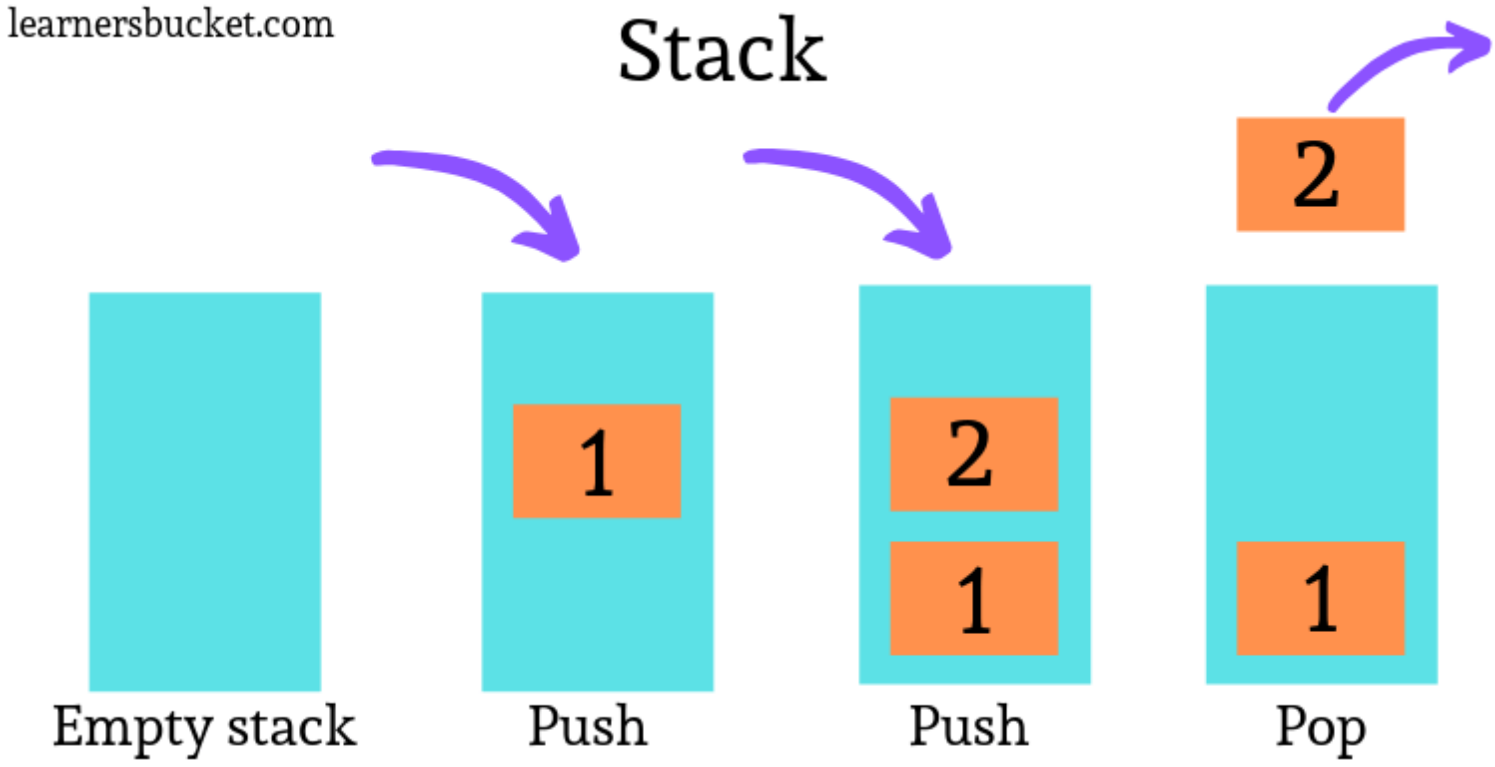
Posted in Algorithms, Stack | Tagged Easy

Learn how to implement the stack data structure in which we can get the max and min value through function in O(1) time.

Example

```
Input:
2 5 17 23 88 54 1 22

Output:
max: 88
min: 1
```



Implement stack with min and max function

- The idea is very simple, instead of storing a single value in the stack we will store an object with `current`, `max` and `min` values.
- While adding the new value in the stack we will check if stack is empty or not.
- If it is empty then add the current value as `current`, `max` and `min` values.

Practically prepare for your JavaScript interview

JavaScript Revision
JavaScript-Concept Based Problems
Data Structures
Algorithms
Machine Coding
Web Fundamentals

- Else get the previous value and compare it with the current item, if it is greater than the existing then replace the `max`, If it is less than the existing the replace the `min`.

Adding a new item in the stack (Push operation)

```
this.push = (item) => {
   //Check if stack is empty
   //Then add the current value at all place
   if(length === 0){
     items[length++] = {value: item, max: item, min: item};
   }else{
     //Else get the top data from stack
     const data = this.peek();
     let {max, min} = data;

     //If it is greater than previous then update the max
     max = max < item ? item : max;

     //If it is lesser than previous then update the min
     min = min > item ? item : min;

     //Add the new data
     items[length++] = {value: item, max, min};
   }
 }
```

Complete code

Copy

```
function stackWithMax(){
    let items = [];
    let length = 0;

    this.push = (item) => {
      //Check if stack is empty
      //Then add the current value at all place
      if(length === 0){
        items[length++] = {value: item, max: item, min: item};
      }else{
        //Else get the top data from stack
        const data = this.peek();
        let {max, min} = data;

        //If it is greater than previous then update the max
        max = max < item ? item : max;

        //If it is lesser than previous then update the min
        min = min > item ? item : min;

        //Add the new data
        items[length++] = {value: item, max, min};
      }
    }

    //Remove the item from the stack
    this.pop = () => {
      return items[--length];
    }

    //Get the top data
    this.peek = () => {
      return items[length - 1];
    }

    //Get the max value
    this.max = () => {
      return items[length - 1].max;
    }

    //Get the min value
    this.min = () => {
      return items[length - 1].min;
    }

    //Get the size
    this.size = () => {
      return length;
    }

    //Check if its empty
    this.isEmpty = () => {
      return length === 0;
    }

    //Clear the stack
    this.clear = () => {
      length = 0;
      items = [];
```

```
    }
}
```

Copy

```
Input:
let SM = new stackWithMax();
SM.push(4);
SM.push(7);
SM.push(11);
SM.push(23);
SM.push(77);
SM.push(3);
SM.push(1);
SM.pop();
console.log(`max: ${SM.max()}`, `min: ${SM.min()}`);

Output:
"max: 77" "min: 3"
```

Time Complexity

#	Access	Search	Insert	Delete	Max	Min
Average	Θ(N)	Θ(N)	Θ(1)	Θ(1)	Θ(1)	Θ(1)
Worst	O(N)	O(N)	O(1)	O(1)	Θ(1)	Θ(1)

Space Complexity

#	space
Worst	O(N)

Recommended Posts:

javascript program to find largest of 2 numbers

Maximum consecutive one’s in a binary array

Implement queue using two stack

Program to find the GCD of two numbers in javascript.

Print matrix in L pattern

Find the maximum sum of products of two arrays.

Right circular rotation on an array of integers

Program to check the prime number

Check if two string are anagram of each other

Find largest subarray with equal numbers of 0’s and 1’s

Prev Next