Day 6

1. Write a function to find the maximum element in the stack.

// C++ program to find maximum element in a //stack

#include <bits/stdc++.h>

using namespace std;

class StackWithMax

{

// main stack

stack<int> mainStack;

// stack to keep track of max element

stack<int> trackStack;

public:

void push(int x)

{

mainStack.push(x);

if (mainStack.size() == 1)

{

trackStack.push(x);

return;

}

// If current element is greater than

// the top element of track stack, push

// the current element to track stack

// otherwise push the element at top of

// track stack again into it.

if (x > trackStack.top())

trackStack.push(x);

else

trackStack.push(trackStack.top());

}

int getMax()

{

return trackStack.top();

}

int pop()

{

mainStack.pop();

trackStack.pop();

}

};

// Driver program to test above functions

int main()

{

StackWithMax s;

s.push(20);

cout << s.getMax() << endl;

s.push(10);

cout << s.getMax() << endl;

s.push(50);

cout << s.getMax() << endl;

return 0;

}

1. Write a function to find the minimum element in the stack.

#include <iostream>

#include <stack>

class Stack

{

// main stack to store elements

std::stack<int> s;

// variable to store minimum element

int min;

public:

// Inserts a given element on top of the stack

void push(int x)

{

if (s.empty()) {

s.push(x);

min = x;

}

else if (x > min) {

s.push(x);

}

else {

s.push(2 \* x - min);

min = x;

}

}

// Removes top element from the stack and returns it

void pop()

{

if (s.empty()) {

std::cout << "Stack underflow!!" << '\n';

}

int top = s.top();

if (top < min)

min = 2 \* min - top;

s.pop();

}

// Returns the minimum element from the stack in constant time

int minimum()

{

return min;

}

};

int main()

{

Stack s;

s.push(6);

std::cout << s.minimum() << '\n';

s.push(7);

std::cout << s.minimum() << '\n';

s.push(5);

std::cout << s.minimum() << '\n';

s.push(3);

std::cout << s.minimum() << '\n';

s.pop();

std::cout << s.minimum() << '\n';

s.pop();

std::cout << s.minimum() << '\n';

return 0;

}