**ASSIGNMENT 5 | 8th January 2021 DSA**

**Question 1**

**Write the function for insertion sort.**

int A[6] = {5,1,6,2,4,3};

int i, j, key;

for(i=1;i<6;i++)

{

key=a[i];

j=i-1;

while(j>=0 && key<a[j])

{

a[j+1]=a[j];

j--;

}

a[j+1]=key;

}

**Question 2**

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

#include <iostream>

#include <stack>

using namespace std;

class CustomStack {

stack<int> stk;

int stack\_max;

public:

void getMax() {

if (stk.empty())

cout << "Stack is empty"<<endl;

else

cout << "Maximum Element in the stack is: "<< stack\_max <<endl;

}

void peek() {

if (stk.empty()) {

cout << "Stack is empty ";

return;

}

int top = stk.top(); // Top element.

cout << "Top Most Element is: "<<endl;

(top > stack\_max) ? cout << stack\_max : cout << top;

}

void pop() {

if (stk.empty()) {

cout << "Stack is empty"<<endl;

return;

}

cout << "Top Most Element Removed: ";

int top = stk.top();

stk.pop();

if (top > stack\_max) {

cout << stack\_max <<endl;

stack\_max = 2 \* stack\_max - top;

} else

cout << top <<endl;

}

void push(int element) {

if (stk.empty()) {

stack\_max = element;

stk.push(element);

cout << "Element Inserted: " << element <<endl;

return;

}

if (element > stack\_max) {

stk.push(2 \* element - stack\_max);

stack\_max = element;

} else

stk.push(element);

cout << "Element Inserted: " << element <<endl;

}

};

int main() {

CustomStack stk;

stk.push(4);

stk.push(6);

stk.getMax();

stk.push(8);

stk.push(20);

stk.getMax();

stk.pop();

stk.getMax();

stk.pop();

stk.peek();

}

**Question 3**

**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;

}