Assignment : 6 30/12/2020

Q:) Write a function to find maximum element in the stack.

→ in c++ :

```
#include<bits/stdc++.h>
using namespace std;
class stackWithMax
{
  stack<int> mainstack;
  stack<int> trackstack;
public:
  void push(int x)
  {
    mainstack.push(x);
    if(mainstack.size()==1)
    {
      trackstack.push(x);
      return;
    }
    if(x>trackstack.~~push(x)~~top())
      trackstack.push(x);
    else
      trackstack.push(trackstack.top());
  }
  int getMax()
  {
    return trackstack.top();
  }
  int pop()
  {
    mainstack.pop();
    trackstack.pop();
```

```
};
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;
}
```

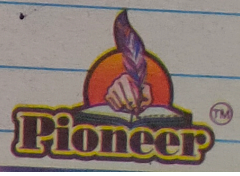
→ O/P: 20
20
50

Q.2) Write function to find the minimum element in the stack

→ in C++:

```
#include<bits/stdc++.h>
using namespace std;
struct myStack
{
    stack<int>S;
    int minEle;
    void getMin()
    {
        if(S.empty())
        cout<<"stack is empty \n";
        else
        cout<<"minimum Element in the stack is: "
            <<minEle<<"\n";
    }
```

```
void peek()
{
  if(s.empty())
   {
     cout<<"stack is empty";
     return;
   }
  int t=s.top();
  cout<<"Top most Element is:";
    (t<minele)? cout<<minele : cout<<t;
}
void pop()
{
  if (s.empty())
   {
     cout<<"stack is empty \n";
     return;
   }
  cout<<"Top Most Element Removed:";
   int t=s.top();
   s.pop();
  if (t<minele)
   {
     cout<<minele<<"\n";
     minele = 2*minele-t;
   }
```



```
        else
            cout<<t<<"\n";
    }
    void push(int x)
    {
        if(s.empty())
        {
            minEle = x;
            s.push(x);
            cout<<"No. Inserted: "<<x<<"\n";
            return;
        }
        if (x<minEle)
        {
            s.push(2*x-minEle);
            minEle=x;
        }
        else
            s.push(x);
        cout<<"No. Inserted: "<<x<<"\n";
    }
};
// Driver code
    int main()
{
```

```
    MyStack S;
    S.push(3);
    S.push(5);
    S.getmin();
    S.push(2);
    S.push(1);
    S.getmin();
    S.pop();
    S.getmin();
    S.pop();
    S.peek();
    return 0;
}
```

→ O/P:

```
No. Inserted : 3
No. Inserted : 5
minimum Element in Stack is : 3
No. Inserted : 2
No Inserted : 1
Minimum Element in STACK is : 1
Top most Element in Stack is : 2
Top most Element Removed : 2
Top most Element is : 5
```