Mohammad Arshad Hossain Ratul (1930319)

NO.1

```
#include<iostream>
#include<stack>
Using namespace std;
void display (stack<int>);
void pushStack(int);
Stack<int> s; // Global stack.
int main()
{   int a,inp;
    do
    {  cout<<"\n1. push. \n2. display. \n3. Exit. \n Enter choice:";
       cin>>a;
      switch(a)
      {
       case 1:
          { cout<<"Enter a value you want to push: ";
            cin >> inp;
            pushStack(inp);
            break;
          }
       case 2:
          { display(s);
            break;
          }
       case 3:
          { cout<<"Thank You. \n";
            break;
          }
       default:
          { cout<<"Invalid option try again.\n";
          }
      }while(a!=3);
```

```
    return 0;
}

void pushstack(int n)
{   stack<int> b; // help stack.
    int stack_size = s.size()+1; // to know the initial size.
    if(stack_size>1) // taking input if the size of ~~app~~ stack <1.
    {   while(s.top()<n) // storing the values in stack 'b' till n is greater.
        {   b.push(s.top());
            s.pop();
        }
        s.push(n); // pushing the value of 'n';
        while(!b.empty()) // re-storing the values which are small.
        {   s.push(b.top());
            b.pop();
        }
    }
    else {s.push(n);} // For taking 1st input
    display(s);
}
void display(stack<int>a) // Function to print stack.
{   stack<int> b = a;
    while(!b.empty())
    {   cout<<b.top()<<"\n";
        b.pop();
    }
}
```

NO.2

```
#include<iostream>
#include<stack>
using namespace std;
void print_stack(stack<int>);
int main()
{
    int temp,top;
    stack<int> stac;
    stack<int> helperstac;

    Stacl
    Stac.push(44);
    Stac.push(99);
    Stac.push(22);
    Stac.push(44);
    Stac.push(19);
    Stac.push(56);
    Stac.push(53);
    Stac.push(23);
    print_stack(Stac); // printing the stack before making the top highest.
    temp=Stac.top(); // storing the top value in a variable.
    While(!Stac.empty()) // runs till the stack is empty.
    {
        top=Stac.top();
        helperStac.push(top); //storing the values in another stack.
        if(Stac.top()>temp)
        {   temp=Stac.top(); // storing the highest value
        }
```

```
    Stac.pop();
}

    While(!helperStac.empty())
    {
        top = helperStac.top();
        if(helperStac.top() != temp) // pushing all values other than-
                                     the highest.
        {
            Stac.push(top);
        }

        helperStac.pop();
    }

    Stac.push(temp); // pushing the highest value on top position.
    cout<< "\n After making the top highest\n";
    print_stack(Stac); // printing after making the top highest.
    return 0;
}

Void print_stack(stack<int> a) // Function to print stack.
{   stack<int> b = a;
    while(!b.empty())
    {
        cout<< b.top() << "\n";
        b.pop();
    }
}
```

NO. 3

```
#include<iostream>
#include<stack>
using namespace std;
void print_stack(stack<int>);
int main()
{
    int inp, x;
    stack<int> s;
    stack<int> b;
    for(int i=0;i<10;i++) // For taking 10 elements in the stack.
    {
        cout<<"Enter a stack value: ";
        cin >> inp;
        s.push(inp);
    }
    cout<<"The initial stack: \n";
    print_stack(s); // printing the initial stack.
    cout<<"Enter the value up to which the stack will pop: ";
    cin>>x;
    cout<<"The stack after delete: \n";
    while(!s.empty()) // till stack 's' is empty.
    {
        if(!(s.top()<=x)) // pushing value in stack 'b' if not less or equal to x.
        {
            b.push(s.top());
        }
        s.pop();
    }
```

```
while (!b.empty()) // pushing the elements in stack 's', those are
{                          more than x.
    s.push(b.top());
    b.pop();
}

print_stack(s);

return 0;
}

void print_stack(stack<int> a) // Function to print stack.
{  stack<int> b=a;
   while (!b.empty())
   {  cout<< b.top()<< "\n";
      b.pop;
   }
}
```

NO.4

```
#include<iostream>
#include<stack>
#include<queue>
Using namespace std;
void print_stack(stack<int>);
int main()
{
    int inp, x;
    queue<int> b;
    stack<int> s;
  for(int i=0; i<10; i++) // For taking 10 elements in stack 's'.
  { cout<<"Enter a stack value: ";
     cin>>inp;
     s.push(inp);
  }
 cout<<"The initial stack: \n";
 print_stack(s); //printing the initial stack.
cout<<"Enter a value up to which the stack will pop: ";
cin>> x;
cout<<"The stack after delete: \n";
While(!s.empty()) // Pushing the values in queue 'b' which
{ if(!(s.top()<=x)                  are greater than x.
   {
     b.push(s.top());
   }
   s.pop();
}
```

```
while(!b.empty()) // Pushing the values in stack after deleting
{                 //          values less than or equal x.
    s.push(b.front());
    b.pop();
}

while (!s.empty()) // Pushing the values in queue again to make
{                  //          the stack in correct order.
    b.push(s.top());
    s.pop();
}

while(!b.empty()) // Pushing the values in correct order.
{
    s.push(b.front());
    b.pop();
}
print_stack(s);
return 0;
}
void print_stack(stack<int> a) // Function to print stack.
{
    stack<int> b = a;
    while(!b.empty())
    {
        cout<< b.top() << "\n";
        b.pop();
    }
}
```