Stack Class

```
#include <iostream>
using namespace std;

template<class T>
class Stack
{
private:
    T *st;
    int size;
    int top;
public:
    Stack(){size=10;top=-1;st=new T[size];}
    Stack(int size){this->size=size;top=-1;st=new
T[this->size];}

    void push(T x);
    T pop();
    T peek(int index);
    int stacktop();
    int isEmpty();
    int isFull();
    void Display();
};

template<class T>
void Stack<T>::push(T x)
{
    if(isFull())
        cout<<"Stack Overflow"<<endl;
    else
    {
        top++;
        st[top]=x;
    }
}
```

```
template<class T>
T Stack<T>::pop()
{
    T x=-1;
    if(isEmpty())
        cout<<"Stack underlfow"<<endl;
    else
    {
        x=st[top];
        top--;
    }
    return x;
}

template<class T>
T Stack<T>::peek(int index)
{
    T x=-1;
    if(top-index+1<0)
        cout<<"Invalid Index"<<endl;
    else
        x=st[top-index+1];

    return x;
}

template<class T>
int Stack<T>::stacktop()
{
    if(isEmpty())
        return -1;
    return st[top];
}

template<class T>
int Stack<T>::isFull()
{
    return top==size-1;
}
```

```
template<class T>
int Stack<T>::isEmpty()
{
    return top==-1;
}

template<class T>
void Stack<T>::Display()
{
    for(int i=top;i>=0;i--)
        cout<<st[i]<<" ";
    cout<<endl;
}

int main()
{
    Stack<char> stk(5);
    stk.push('a');
    stk.push('b');
    stk.push('c');
    stk.push('d');
    stk.push('e');
    stk.push('f');

    stk.Display();

    cout<<stk.peek(1)<<endl;


    return 0;
}
```