**Assignment - 41 A Job Ready Bootcamp in C++, DSA and IOT**

**Stack**

1. Create a stack of int type, push 5 elements in it and print it on the console screen.

Sol – 1.

#include<bits/stdc++.h>

using namespace std;

void printstack(stack<int>s)

{

if(s.empty())

return;

int x=s.top();

s.pop();

cout<<x<<" ";

printstack(s);

s.push(x);

}

int main()

{

stack<int>s;

s.push(1);

s.push(2);

s.push(3);

s.push(4);

s.push(5);

printstack(s);

return 0;

}

1. Create a stack of int type, and find the top most element in a stack.

Sol – 2.

#include<bits/stdc++.h>

using namespace std;

int main()

{

stack<int>s;

s.push(1);

s.push(2);

s.push(3);

s.push(4);

s.push(5);

cout<<"Top : "<<s.top();

return 0;

}

3. Create a stack, and implement main operations like push(), pop(), peek(), empty()

and size().

Sol – 3.

#include<iostream>

using namespace std;

class stack

{

private :

int a[100];

int top;

public :

stack()

{

top=-1;

}

~stack()

{

top=-1;

}

void push(int val)

{

if(top==99)

{

cout<<"OVERFLOW!"<<endl;

}

else{

a[++top]=val;

}

}

int getsize()

{

return top+1;

}

void pop()

{

if(top==-1)

{

cout<<"UNDERFLOW!"<<endl;

}

else

{

cout<<a[top--]<<" deleted"<<endl;

}

}

bool isstackempty()

{

if(top==-1)

return true;

}

int peek()

{

if(top==-1)

{

cout<<"UNDERFLOW!"<<endl;

return 0;

}

else

{

return a[top];

}

}

};

int main()

{

stack s;

s.push(5);

s.push(4);

s.push(3);

s.push(2);

s.push(1);

cout<<"Top : "<<s.peek()<<endl;

s.pop();

s.pop();

cout<<"Size : "<<s.getsize()<<endl;

return 0;

}

4. Reverse the Words of a String using Stack.

Example:

Input: str = “I Love To Code”

Output: Code To Love I

Sol – 4.

#include<bits/stdc++.h>

using namespace std;

void reverse(string s)

{

string temp="";

stack<string>stc;

for(int i=0;i<s.length();i++)

{

if(s[i]==' ')

{

stc.push(temp);

temp="";

}

else

{

temp=temp+s[i];

}

}

stc.push(temp);

while(!stc.empty())

{

cout<<stc.top()<<" ";

stc.pop();

}

cout<<endl;

}

int main()

{

string s="I Love To Code";

reverse(s);

return 0;

}

5. Create stack1 of int type, and create another stack of the same type with name

stack2 and copy all the elements of stack1 into stack2 in the same order.

Sol – 5.

#include<bits/stdc++.h>

using namespace std;

int main()

{

stack<int>s1;

s1.push(1);

s1.push(2);

s1.push(3);

s1.push(4);

s1.push(5);

deque<int>dq;

while(!s1.empty())

{

dq.push\_front(s1.top());

s1.pop();

}

stack<int>s2;

while(!dq.empty())

{

s2.push(dq.front());

dq.pop\_front();

}

while(!s2.empty())

{

cout<<s2.top()<<endl;

s2.pop();

}

return 0;

}

OR

#include<bits/stdc++.h>

using namespace std;

int main()

{

stack<int>s1;

for(int i=0;i<5;i++)

{

s1.push(i+1);

}

stack<int>s2(s1);

cout<<"Stack 2 : "<<endl;

while(!s2.empty())

{

cout<<s2.top()<<endl;

s2.pop();

}

return 0;

}

6. Reverse a string using a stack.

Example:

Input: str = "Reverse me"

Output: em esreveR

Sol – 6.

#include<bits/stdc++.h>

using namespace std;

void reverse(string str)

{

stack<char>s1;

for(int i=0;i<str.length();i++)

{

s1.push(str[i]);

}

cout<<"Reverse : "<<endl;

while(!s1.empty())

{

cout<<s1.top();

s1.pop();

}

}

int main()

{

reverse("Reverse me");

return 0;

}

7. Create a stack of int type and sort it.

Sol – 7.

#include<bits/stdc++.h>

using namespace std;

stack<int> sortfun(stack<int> &mainstack)

{

stack<int> tempstack;

while(!mainstack.empty())

{

int temp=mainstack.top();

mainstack.pop();

while(!tempstack.empty()&&tempstack.top()>temp)

{

mainstack.push(tempstack.top());

tempstack.pop();

}

tempstack.push(temp);

}

return tempstack;

}

int main()

{

stack<int>mainstack;

mainstack.push(31);

mainstack.push(18);

mainstack.push(56);

mainstack.push(24);

mainstack.push(66);

mainstack.push(6);

stack<int>tempstack=sortfun(mainstack);

while(!tempstack.empty())

{

mainstack.push(tempstack.top());

tempstack.pop();

}

cout<<"Sorted number in increasing order are :"<<endl;

while(!mainstack.empty())

{

cout<<mainstack.top()<<" ";

mainstack.pop();

}

return 0;

}

8. Create a stack of int type and sort it in descending order.

Sol – 8.

#include<bits/stdc++.h>

using namespace std;

stack<int> sortfun(stack<int> &mainstack)

{

stack<int> tempstack;

while(!mainstack.empty())

{

int temp=mainstack.top();

mainstack.pop();

while(!tempstack.empty()&&tempstack.top()>temp)

{

mainstack.push(tempstack.top());

tempstack.pop();

}

tempstack.push(temp);

}

return tempstack;

}

int main()

{

stack<int>mainstack;

mainstack.push(31);

mainstack.push(18);

mainstack.push(56);

mainstack.push(24);

mainstack.push(66);

mainstack.push(6);

stack<int>tempstack=sortfun(mainstack);

vector<int>v;

while(!tempstack.empty())

{

int temp = tempstack.top();

v.push\_back(temp);

tempstack.pop();

}

cout<<"Sorted number in decreasing order are :"<<endl;

for(auto i=v.begin();i!=v.end();i++)

cout<<\*i<<endl;

return 0;

}

9. Create back button functionality using stack.

Sol – 9.

#include<bits/stdc++.h>

using namespace std;

string current\_url="";

stack<string>forward\_st;

stack<string>backward\_st;

void visit\_new\_url(string url)

{

if(current\_url!="")

backward\_st.push(current\_url);

current\_url=url;

}

void forward()

{

if(forward\_st.empty()||current\_url==forward\_st.top())

{

cout<<"Not Available\n";

return;

}

else

{

backward\_st.push(current\_url);

current\_url=forward\_st.top();

forward\_st.pop();

}

}

void backward()

{

if(backward\_st.empty()||current\_url==backward\_st.top())

{

cout<<"Not Available\n";

return;

}

else

{

forward\_st.push(current\_url);

current\_url=backward\_st.top();

backward\_st.pop();

}

}

10. Given an array, print the Next Greater Element (NGE) for every element using stack.

Example:

Input: arr[] = [ 4 , 5 , 2 , 25 ]

Output: 4 –> 5

5 –> 25

2 –> 25

25 –> -1

Sol – 10.

#include<bits/stdc++.h>

using namespace std;

void printNGE(int arr[],int n)

{

stack<int>s;

s.push(arr[0]);

for(int i=1;i<n;i++)

{

if(s.empty())

{

s.push(arr[i]);

continue;

}

while(s.empty()==false&&s.top()<arr[i])

{

cout<<s.top()<<"-->"<<arr[i]<<endl;

s.pop();

}

s.push(arr[i]);

}

while(s.empty()==false)

{

cout<<s.top()<<"-->"<<-1<<endl;

s.pop();

}

}

int main()

{

int arr[]={4,8,2,13,9,11};

int size=sizeof(arr)/sizeof(arr[0]);

printNGE(arr,size);

return 0;

}