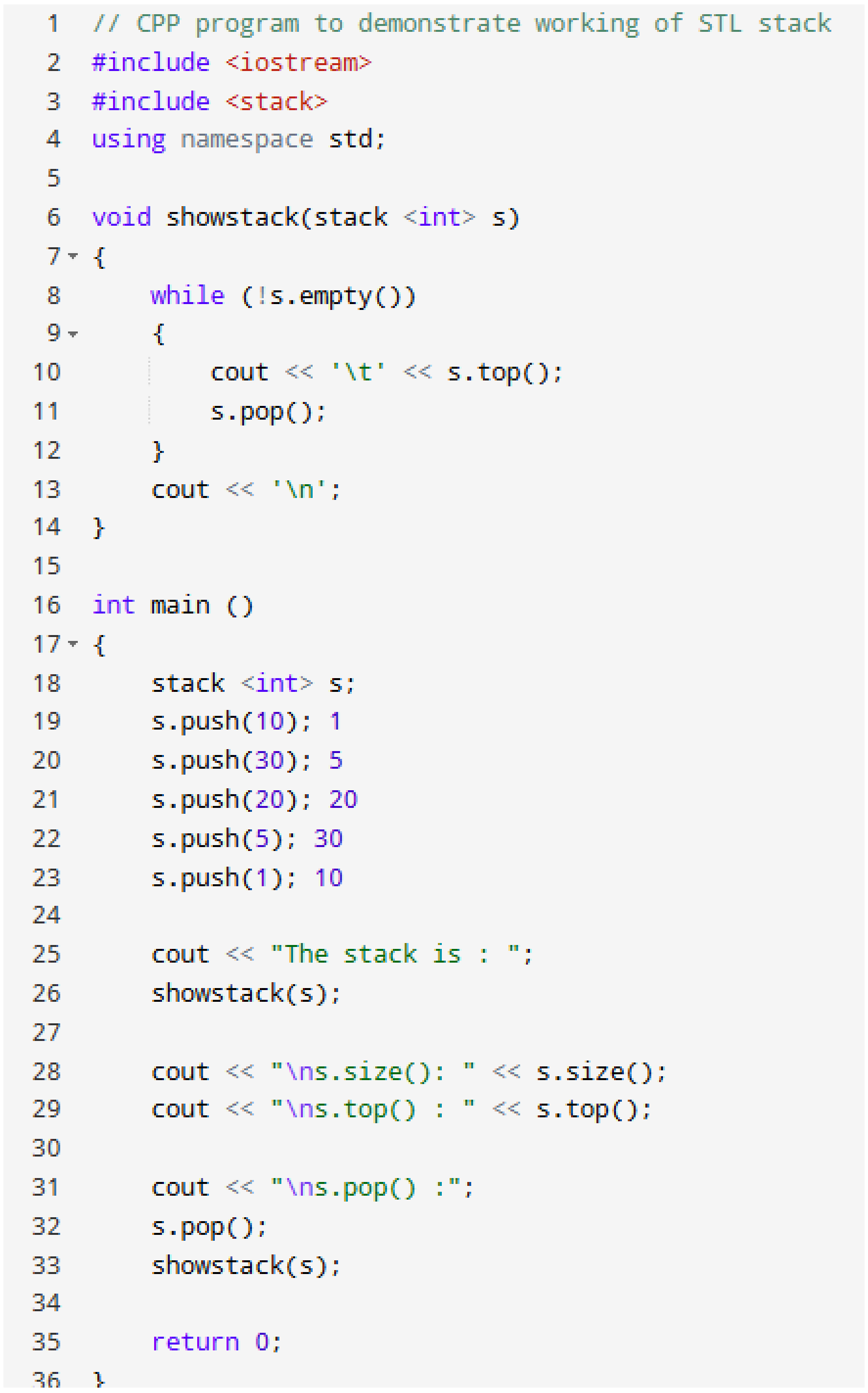
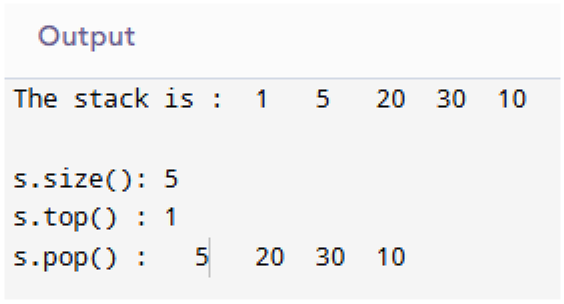
Seatwork 4.1	
Seatwork 4.1 : Stacks	
Course Code: CPE010	**Program:** Computer Engineering
Course Title: Data Structures and Algorithms	**Date Performed:** 08 - 12 - 25
Section: CPE21S4	**Date Submitted:** 08 - 12 - 25
Name(s): Santiago, David Owen A.	**Instructor:** Engr. Jimlord Quejado

6. Output



```cpp
// CPP program to demonstrate working of STL stack
#include <iostream>
#include <stack>
using namespace std;

void showstack(stack <int> s)
{
    while (!s.empty())
    {
        cout << '\t' << s.top();
        s.pop();
    }
    cout << '\n';
}

int main ()
{
    stack <int> s;
    s.push(10); 1
    s.push(30); 5
    s.push(20); 20
    s.push(5); 30
    s.push(1); 10

    cout << "The stack is : ";
    showstack(s);

    cout << "\ns.size(): " << s.size();
    cout << "\ns.top() : " << s.top();

    cout << "\ns.pop() :";
    s.pop();
    showstack(s);

    return 0;
}
```



Output

```
The stack is :  1   5   20  30  10

s.size(): 5
s.top() : 1
s.pop() :   5   20  30  10
```

7. Supplementary Activity

8. Conclusion

In this activity, we utilized the basic implementation of a stack in c++. A void function is created that continuously prints each element of a stack through a while loop. It does this by first checking if the stack is empty (using the !s.empty condition) then, if satisfied, prints the top element. Afterwards, it pops the top element to move onto the element below it. In the main function, each element is inserted using the push function, then the void function is called after the stack is created. It then prints the size of the stack as well as the current top element (after popping the top element: 1).

9. Assessment Rubric