# Lab Tutorial for Week 3 Session 2: Template for Stack

## Contents

1. Tutorial w3s2a: Function Template
2. Tutorial w3s2b: Class Template
3. Tutorial w3s2c: Template for Stack

## Tutorial w3s2a: Function Template

### Tasks:

1. Create a new file (CTRL-N).
2. Solve the following programming problem. Suppose you need to write a function that determines the larger of two items. Both items can be integers, floating-point numbers, characters, or strings.
3. Create a function to compute the larger of two integers
4. Create a function to compute the larger of two floating point numbers (decimals)
5. Create a function to compute the larger of two character data
6. Create a function to compute the larger of two string data
7. Save each function above with the main function in a separate file, for example larger\_of\_int.cpp
8. You can start from the main function code:

#include <iostream>

#include <string>

using namespace std;

// your code here

int main() {

string data1, data2;

cout << "Enter first data " << endl;

cin >> data1;

cout << "Enter second data " << endl;

cin >> data2;

cout << "The larger of " << data1 << " and " << data2 << " is " << larger(data1, data2) << endl;

return 0;

}

// your code here

1. Complete the program.
2. Save each of the source file, compile and run it.
3. Continue with the following programming problem.
4. Create a function template to compute the larger of two data of any data type (int, double, char, string, etc)
5. Save the function above with the main function in a separate file, for example larger\_template.cpp
6. You can use the same main function as in the step 3 above.
7. Complete the program.
8. Save the source file, compile and run it.

### Questions:

1. What new C++ programming keywords that you learned from this exercise?
2. What is function template?
3. Why is it useful?

## Tutorial w3s2b: Class Template

### Tasks:

1. Create a new Project in Dev-C++ (File, New, Project)
2. Solve the following programming problems. Suppose we need to write a class that has two member data of type integer and a method to computer which member data is larger. We can interpret the class Larger as a fraction having the first member data as the numerator and the second member data as the denominator.

#include <iostream>

#include <string>

using namespace std;

class Larger {

private:

int first;

int second;

public:

Larger();

Larger(int f, int s);

int getFirst();

int getSecond();

void setFirst(int f);

void setSecond(int s);

int isLarger();

};

// your code here

int main() {

Larger l;

int f, s;

cout << "Enter first data " << endl;

cin >> f;

l.setFirst(f);

cout << "Enter second data " << endl;

cin >> s;

l.setSecond(s);

cout << "The larger of " << f << " and " << s << " is " << l.isLarger() << endl;

return 0;

}

1. Change the data type of the data members to different type, for example string, then rewrite the class.
2. Do the same with integer and character data type for the member data.
3. You might have noticed that the logic of the methods are exactly the same. This is not efficient coding. Why should we need to write each class separately? Class template save us from writing such duplicate code.
4. Create the class template for the above Large class which is independent of any particular data type.

#include <iostream>

#include <string>

using namespace std;

template <class T>

class Larger {

private:

T first;

T second;

public:

Larger();

Larger(T f, T s);

T getFirst();

T getSecond();

void setFirst(T f);

void setSecond(T s);

T isLarger();

};

// your code here

int main() {

Larger<int> l;

string f, s;

cout << "Enter first data " << endl;

cin >> f;

l.setFirst(f);

cout << "Enter second data " << endl;

cin >> s;

l.setSecond(s);

cout << "The larger of " << f << " and " << s << " is " << l.isLarger() << endl;

return 0;

}

### Questions:

1. What new C++ programming keywords that you learned from this exercise?
2. What is class template?
3. Why is it useful?

## Tutorial w3s2c: Template for Stack

### Tasks:

1. Now, having practiced on how to create function template and class template, please convert your Stack code in the tutorial session 1 into template

template <class T>

class Stack

{

private:

T st[MAX];

T top=-1;

public:

void push(T val);

T pop();

T peek();

void display();

};

1. Save the completed source file and make sure it’s compiled.
2. Do the same test as the previous tutorial, but now you can use different data types without rewriting the Stack code at all

### Questions:

1. If you’re asked to separate your Stack code into INTERFACE, IMPLEMENTATION and CLIENT, how would you do that?

---end of Tutorial Week 3 Session 2---