

Remember "Reviewing C++" has many more examples and topics.

Tutorial 14 **Templates**

Understanding templates

Templates are useful in the sense that they allow multiple types to be used instead of set types. In other words, a simple function could be written as follows:

```
double max( double a, double b ){
    if( a > b ) return a;
    else return b;
}
```

The above function will return the max of two double precision numbers.

But say you didn't want to only find the max of two double precision numbers. Say you wanted to be able to find the max of any number (long, float etc...). Without the use of a template, you could not do this. Here is what the new max function would look like:

```
template< class T >
T max( T a, T b ){
    if( a > b ) return a;
    else return b;
}
```

The template is defined above where class can be any type either pre defined or user defined. These include int, float, short, long, etc... The letter or name following the keyword class does not always have to be a capital T but it should be a useful name or letter.

The above functions allows us to find the max of anything. The T is the new data type. If we did not do this, we would have to rewrite the max function for every conceivable type in C++ (thankfully there is the template).

A template works for a class as well. In a similar fashion:

```
template< class T >
class Name{

private:
    T a, b, c; //data of type T accordingly.
public:
    //rest of code below
};
```

Remember "Reviewing C++" has many more examples and topics.

Now this class could be used with any type. Here is a short program demonstrating the template of the max function above:

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

template< class T >
T max(T a, T b){
    if(a > b) return a;

    return b;
}

int main(){

    cout << max(7.03 , 7.1) << endl;
        cout << max(9, 11) << endl;
    cout << max(.01, .001) << endl;

    return 0;
}
```

Just note that you cannot template the main function! Do not try it as it will confuse the execution of the program.

Here is an example using a template class called Point:

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

template< class T >
class Point{

private:
    T x, y;

public:
    Point(){ }
    Point(T a, T b){
        x = a;
        y = b;
    }

    T getX(){ return x; }
    T getY(){ return y; }
};

int main(){

    Point<int> pt(3,5);
    return 0;
}
```