CSCI 2275 – Programming and Data structures

Instructor: Hoenigman

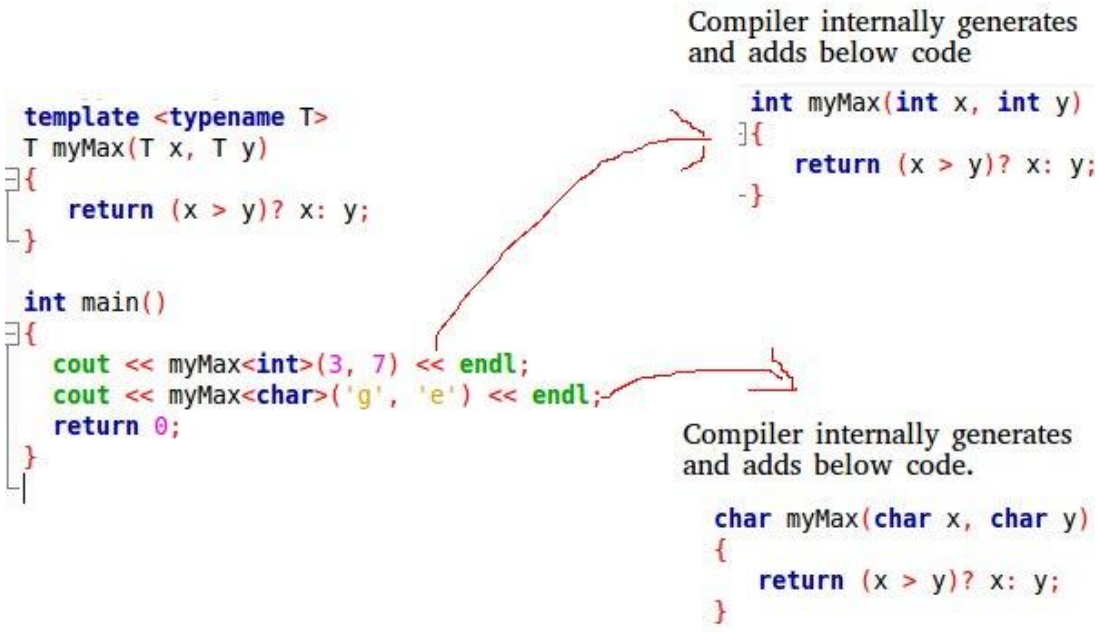
Recitation 9

Learning Objectives:

1. Templates
2. STL
3. Try-catch exception
4. Debugger

1. Template functions and classes

   The simple idea is to pass data type as a parameter so that we don't need to write same code for different data types. For example a software company may need sort() for different data types. Rather than writing and maintaining the multiple codes, we can write one sort() and pass data type as a parameter.



You can replace the typename by class. Templates will work with any kind of datatype – even user defined. But it will be your responsibility to make sure that whatever is required by the template class/function is implemented.

You can have a template class/ function to have specialized implementation for specific datatypes.
Reference : https://www.geeksforgeeks.org/template-specialization-c/

You can pass in multiple classes/typenames as parameters to a function/class.

2. STL – Standard Template Library
   As discussed earlier programmers are smart (and lazy). There are standard and optimized implementations of a lot of required data structures, sorting functions and important algorithms already.

   The important point to use STL is read and understand the predefined functions. Most of the time you will not see the implementation of these functions. Documentations and official references are your biggest friends:

   Some of the important STL implementations:
   1. Vectors
   2. Lists (Doubly Linked List)
   3. Forward Lists
   4. Sets (Unique collections of the data)
   5. Map (Key value pairs of data)
   6. Stacks
   7. Queues
   8. Priority Queue
   9. Sorting Templates
   10. Binary Search

3. Try-catch exceptions

Most of the times while writing huge code the basic error are not enough. And having complicated if else conditions can just make your code harder to maintain.

Try – catch exceptions can help you clear the code better.

Whatever you have in try will be executed. If something goes wrong, you can “throw” an exception and “catch” after. Now the important think to remember is your function will catch only what you program it to do.

Eg:

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

int main()
{
    try  {
       throw 10;
    }
    catch (char *excp)  {
        cout << "Caught " << excp;
    }
    catch (...)  {
        cout << "Default Exception\n";
    }
    return 0;
}
```

References : http://www.cplusplus.com/reference/exception/exception/

4. Debugger

Download the files provided. terrible dynamic size array unsorted.h and terrible dynamic size array unsorted.cpp has declaration and definition of a array data structure and its functionality. This code has bugs. Try to find out bugs using debugger.