Name: _____ SID: _____

COMP-2355, Winter 2012
Final Exam
March 13, 2012

READ ALL INSTRUCTIONS BEFORE TAKING THIS EXAM!

*This midterm is worth 25% of your course grade. Be careful to **read all instructions** for the exam before you begin. A small number of points will be deducted for not following instructions correctly.*

The exam should be written in black or blue ink. All answers should be written inside the exam booklet. No other materials are allowed during the exam. If you are unsure what a question is asking, write your assumptions. If there are more general clarifications, ask the proctor of the exam.

Write your name and SID on every page of the exam.

This exam should be entirely your own work. Students caught cheating may receive a 0 on the midterm and/or be subject to academic discipline. Additionally, it is your responsibility to make sure that other students cannot copy your work.

Answer Question 1 and nine of the remaining ten questions. Place an X through the question you do not want to be graded.

Question 1 _____ [of 10]
Question 2 _____ [of 10]
Question 3 _____ [of 10]
Question 4 _____ [of 10]
Question 5 _____ [of 10]
Question 6 _____ [of 10]
Question 7 _____ [of 10]
Question 8 _____ [of 10]
Question 9 _____ [of 10]
Question 10 _____ [of 10]
Question 11 _____ [of 10]
Total _____ [of 100]

Name: _____ SID: _____

1. General questions [2 points each]:

a) What is the difference between a class and a struct?

b) What are two ways to create new types?

c) What happens to an object allocated on the stack after a function returns?

d) What happens to an object allocated on the heap after a function returns?

e) What is the size of an `int`?

Name: _____ SID: _____

Sample Code

This exam will use the following classes in multiple questions.
Refer to them as necessary during the exam.

MyClass.h:

```
class MyClass {
public:
	MyClass();
	MyClass(int value);
	int GetValue();
	void SetValue(int value);
	int FunctionOfValue();
	void Print();
private:
	int mMyValue;
};
```

As a hint, the operator prototypes from Lab 10 were:

```
Sentence &operator+=(const Sentence &s);
std::ostream& operator<<(std::ostream &out, const Sentence &f);
```

2. Templates: Re-write the MyClass found on page 3 to use templates. Provide definitions for each of the functions in the class.

3. Operator overloading: Replace the Print function in MyClass with an operator for using streams/cout for printing the class. Show the updated class declaration (without an implementation) and any additional functions (with implementations).

4. STL: Write a complete “main.cpp” file which will create a vector of three MyClass objects by adding them, one at a time, to the end of a vector. Then loop through the vector and compute the sum of the FunctionOfValue() call for each object.

main.cpp:

5. Overloading: Define a new class YourClass which inherits from MyClass and provides an alternate definition for the FunctionOfValue() function. Show any changes to MyClass, as well as the YourClass .h and .cpp files

6. Operators: Define a new operator function inside MyClass such that two MyClass objects can be added together using +=. Provide the changes to both the .h and .cpp files.

7. STL: Write a function which takes a reference to a map containing MyClass and iterates through the class printing each object and the key that was used to store it. (Assume that Question 3 has been completely answered.) (Hint: the type of the iterator is something like: `std::map<  >::iterator`.)

8. Write a complete "main.cpp" file which will write a MyClass object to a file using both file streams and fwrite. (Assume that Question 3 has been completely answered.) (Hint: The second argument to fwrite is the size of the object being written; the third argument is the number of items being written.)

9. Provide the complete .cpp definitions for the following class:

Password.h:

```
class Password {
public:
	Password(int length, int *pass);
	Password(int length); // default password should be all 0's
	~Password();
	bool TestPassword(int *test); // return true if passwords match
	bool SetPassword(int *oldPass, int *newPass); // assume passwords are the same length
private:
	int passLength;
	int *password;
};
```

Password.cpp:

Name: _____ SID: _____

10. Threads: (a) Add one or more mutex locks where necessary and briefly explain why. (b) Will this code run faster with threads? (c) Could it be made to run faster?

```
pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;

int main (int argc, const char * argv[])
{
     MyClass c;
     for (int x = 0; x < numThreads; x++)
             pthread_create(&myThreads[x], 0, Worker, (void*)&c);
     for (int x = 0; x < numThreads; x++)
             pthread_join(myThreads[x], 0);
     return 0;
}


void *Worker(void *obj)
{
     int result = 0;


     MyClass *mc = (MyClass*)obj;


     for (int x = 0; x < 1000; x++)
     {


             mc->SetValue(x);


             result += mc->FunctionOfValue();


     }


     return 0;
}
```

Name: _____ SID: _____

11. The function tolower(char c) returns the lowercase equivalent of a character. Write a function to compare two C strings (char *) and return true if they match in a case insensitive manner:

```
bool CaseInsensitiveMatch(const char *first, const char *second)
{

}
```