CSE 687 Midterm Exam

Notes: You are free to use the internet, (including Google), class notes, textbook etc. You are forbidden from using another individual to help you answer questions.
Copying code or text directly from the internet is also **forbidden**, but you may read any source and then answer questions in your own words.

Name: Evan Smith

Question 1) [8 Points]

a) What are the stages of C++ compilation? (You can just list them.)
Preprocessing, Compilation, Assembly, Linking

b) Explain what happens during the preprocessor stage of compilation.
The compiler copies the #include lines with the full content of those references, allowing the following code to reference it.

c) What relationship does (b) have for the reason we need to use #ifndef statements in our header files?
The #ifndef or #pragma once statements keep the preprocessor from duplicating code that is #included in multiple files.

d) What implications does the way pre-compilation work have on Template classes?
Without using the *extern* keyword, template classes will be compiled every time they are encountered. With the keyword, they will only be compiled once (as possible).

Question 2) [12 Points]

a) What is the difference between implicit and explicit dynamic library linking?

Implicit linking is done by the compiler and relies on a static library that must exist before compile-time, which is then loaded during application startup. Explicit linking is done during runtime, and the program itself manually loads the required parts of the DLL into its process memory.

b) What are the tradeoffs between the two?

Implicit is cleaner (in my opinion) by reducing boilerplate code. It also means that a program will not compile if the DLLs referenced are incorrect or incomplete. With dynamic linking, the load time of the program is faster, since it doesn't need to load in the DLLs ahead of time, but it can fail during runtime if required elements of the DLL does not exist. Additionally, you must know the names of the functions, classes, etc already in order to use explicit linking, while implicit linking provides the benefit of more helpful IntelliSense in most IDEs.

c) When would you choose explicit over implicit DLLs?

If you were very concerned about application start up time, explicit loading might be worthwhile. Additionally, explicit linking allows for dynamically selecting which libraries to load at runtime, which is not possible with implicit linking. This also means that errors loading DLLs explicitly can be caught and handled within the program, rather than preventing the application from starting at all with an implicit linking system.

Question 3) [12 Points]

a) What performance problem can come about from using class templates and function templates?

It can cause long compile-times. Run-time should not be affected, since the compiled version is type-specific.

b) Can a templated class have pure virtual member function? If yes come up with an example, if no state why.

yes:

```
template <typename T> class TemplatedWithPureVirtual
{
public:
    virtual void Func(T data) = 0;
};
```

c) Can a non-templated class have a templated member function? If yes come up with an example, if no state why.

yes:

```
class NonTemplatedClassWithTemplatedFunction
{
public:
    template <typename T> T Func(T data)
    {
        return data;
    }
};
```

d) Can a non-templated class have a templated pure virtual member function? If yes come up with an example, if no state why.

No, because the templated function will only be created when it is invoked with a specific type. This means that the vtable for these functions would have to be mutable, since we only know how many versions of the function will exist at runtime.

Question 4) [28 Points]

Write a templated class Matrix<T> with the following methods:

a) Constructor takes the initial x, y number of elements in the matrix.
b) Provides methods for getting/setting a single entry.
c) Provides copy constructor.
d) Provides the ability to take another matrix add them together if the dimension match.

```
#include <vector>
using std::vector;

template <typename T> class Matrix
{
public:
   Matrix(int x, int y) : _x(x), _y(y)
   {
      //initialize contents to default value
      for (int i = 0; i < _x; i++)
      {
         //generate a column
         vector<T> column;
         for (int j = 0; j < _y; j++)
         {
            column.push_back(T());
         }

         //add it to the matrix
         _matrixContents.push_back(column);
      }
   }
   Matrix(Matrix &inpMatrix)
   {
      _x = inpMatrix.GetXDim();
      _y = inpMatrix.GetYDim();

      //initialize contents to given matrix's value
      for (int i = 0; i < _x; i++)
      {
         //generate a column
         vector<T> column;
         for (int j = 0; j < _y; j++)
         {
            column.push_back(inpMatrix.GetEntry(i,j));
         }

         //add it to the matrix
```

```
            _matrixContents.push_back(column);
        }
    }
    T GetEntry(int x, int y)
    {
        //validate coordinates are within matrix
        if (x >= 0 && y >= 0 && x < _x && y < _y)
        {
            return _matrixContents[x][y];
        }
        else
        {
            //I choose to return the default T to fail silently
            return T();
        }
    }
    void SetEntry(int x, int y, T value)
    {
        //validate coordinates are within matrix
        if (x < _x && y < _y)
        {
            _matrixContents[x][y] = value;
        }
    }
    int GetXDim()
    {
        return _x;
    }
    int GetYDim()
    {
        return _y;
    }
private:
    int _x, _y;
    vector<vector<T>> _matrixContents;
};
```

Question 5) [25 Points]

a) What are the two main performance-related advantages of using threads?

i) Parallel and concurrent processing

ii) can spread work over multiple processors

b) Spurious Wake

i) What is a spurious wake?

When a thread wakes because its wake condition was met, but by the time that it spins up that condition is no longer met.

ii) What can cause them?

Race conditions where other threads alter the initial thread's wake condition.

iii) How do we protect against them in our code?

We should code defensively to validate that the wake condition is actually met when the thread finishes waking. We can also try to avoid relying on conditions that can be frequently toggled by other threads, although this is often impossible.

c) There is a program where N threads have been assigned ids. There exists a function *Foo* which the threads must call in the order that their ids have been assigned. Threads have to wait until all other threads have called foo before than can continue onwards (they should wait in ExitFoo until all other threads have finished).

```
void ThreadFunction(int id,  … ) {
      …
      EnterFoo(...);
      Foo();
      ExitFoo(...);
      …
}
```

Implement the EnterFoo() and ExitFoo() methods using mutex, condition variables and any other variables you see fit. Threads will be created during ThreadFunction, but you may specify additional arguments for ThreadFunction, EnterFoo and ExitFoo methods, if necessary.

```
void ThreadFunction(int id, std::condition_variable cv, std::mutex mutex, std::mutex
finishMutex)
{
    //assuming N threads have been made starting with id = N
    std::unique_lock<std::mutex> lock(mutex);
    //int nextThreadToExecute is set to the first index
    //and stored on the containing class
    EnterFoo(id, cv, lock);
    //Foo();
    ExitFoo(cv, lock, finishMutex);
}

void EnterFoo(int id, std::condition_variable cv, std::unique_lock<std::mutex> lock)
{
    cv.wait(lock, [id] {return id == nextThreadToExecute});
}

void ExitFoo(std::condition_variable cv, std::unique_lock<std::mutex> lock, std::mutex
finishMutex)
{
    //get next thread or trigger exit
    nextThreadToExecute++;

    if (nextThreadToExecute > N)
    {
        //all threads have completed
        //bool allThreadsDone defined on containing class
        allThreadsDone = true;
        lock.unlock();
        cv.notify_all();
    }
    else
    {
        //wait until all threads are done
        std::unique_lock<std::mutex> finishLock(finishMutex);
        cv.wait(finishLock, [] { return allThreadsDone; });
    }
}
```

Question 6) [15 Points]

a) What are the main differences between a lambda expression and a C-style function pointer?

A c-style function pointer doesn't have the concept of capture specification like lambdas do.

b) What are the main differences between a lambda expression and a functor?

Functors are names and are therefore cleaner to reuse. Lambdas are nameless functors, essentially. Functors also have a specific type, while lamdas do not.

c) What is the problem in the following snippet of code?

```
class Adder {
public:
    Adder(int x) {
        x_ = x;
    }
    int Add(int y) {
        return x_ + y;
    }
private:
    int x_;
};

std::function<int(int)> Function (int x) {
    Adder adder(x);
    return  [&](int x) { return adder.Add(x);};
}
```

The Adder that is instantiated in the Function(int x) call falls out of scope when the return block is hit. This means that a caller of the returned function will operate with an uninitialized Adder and will add their argument to random data that is in _x, giving a useless result.