Name: _____   SID: _____

COMP 2355, Winter 2013
Final Exam
March 14, 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.

Part 1   _____ [of 20]

Part 2   _____ [of 40]

Part 3   _____ [of 40]

Total   _____ [of 100]

Bonus   _____ [of 5]

1. Short answer question [Do 4 of 5]   **[5 points each]**

a) What is the difference between a class and a struct in C++?

b) What issues might occur when sharing binary data (e.g. structs written by fwrite/fread) across machines?

c) What must be checked first when writing an assignment operator (`operator=`)?

d) Why is an assignment operator needed?

e) What is the general principle used to determine whether it is suitable for one class to inherit from another?

2. Analysis questions [Do 4 of 5]:   **[10 points each / 40 points total]**

i. In the code sample below, indicate the scope of the data stored in variables a-e? That is, how long is the
    data associated with the variable retained before it is freed from the stack/heap?

```
int *function(int &d, int e)

{

    static int a = 0;

    a++;

    int *b = new int;

    *b = 0;

    while (d > 0 && !e)

    {

        for (int c = 0; c < myValue; c++)

            (*b) += c*e;

        d--;

    }

    return b;

}
```

ii. Fill in the blanks in the function below to iterate through the vector and print it:

```cpp
void PrintVector(std::vector<int> &vals)
{
        for (std::vector<int>::iterator i = _____;

            i != _____;

            _____)
        {
            std::cout << *i << std::endl;
        }

}
```
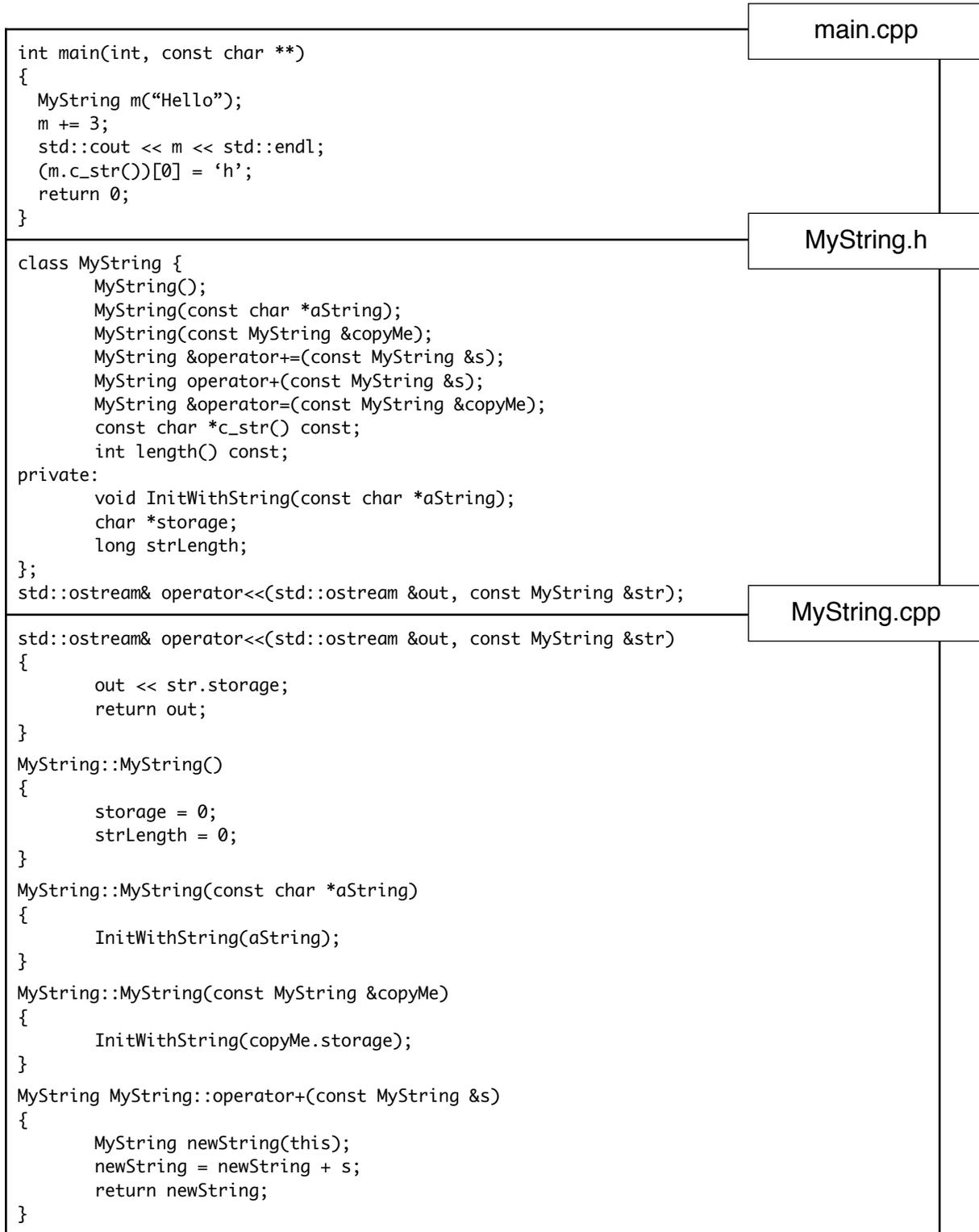
iii. Fill in the table by writing the following numbers/expressions in binary. Then, explain the result of the expression ((x&(x-1))^x).

	x = 6	x = 8	x = 9
x			
x-1			
x&(x-1)			
(x&(x-1))^x			

iv. Find 5 errors in the code fragments below. (Assume proper includes and that all code not shown is also defined.)

main.cpp

```cpp
int main(int, const char **)
{
  MyString m("Hello");
  m += 3;
  std::cout << m << std::endl;
  (m.c_str())[0] = 'h';
  return 0;
}
```

MyString.h

```cpp
class MyString {
        MyString();
        MyString(const char *aString);
        MyString(const MyString &copyMe);
        MyString &operator+=(const MyString &s);
        MyString operator+(const MyString &s);
        MyString &operator=(const MyString &copyMe);
        const char *c_str() const;
        int length() const;
private:
        void InitWithString(const char *aString);
        char *storage;
        long strLength;
};
std::ostream& operator<<(std::ostream &out, const MyString &str);
```

MyString.cpp

```cpp
std::ostream& operator<<(std::ostream &out, const MyString &str)
{
        out << str.storage;
        return out;
}
MyString::MyString()
{
        storage = 0;
        strLength = 0;
}
MyString::MyString(const char *aString)
{
        InitWithString(aString);
}
MyString::MyString(const MyString &copyMe)
{
        InitWithString(copyMe.storage);
}
MyString MyString::operator+(const MyString &s)
{
        MyString newString(this);
        newString = newString + s;
        return newString;
}
```

v. Fill in the sample calls for each function; indicate the value of the variable at the end.

```
class myClass {
public:
      myClass()
      {
            x = &y;
      }
      int *x, y;
};

void MyFunc1(int *value)
{
      value[0] = 3
}

void MyFunct2(int &value)
{
      value = 4;
}

int main(int, const char **)
{
      myClass a;
      myClass *b = new myClass;

      // pass in the x member variable of a
      MyFunc1(                                );

      // pass in the y member variable of b
      MyFunc1(                                );

      // pass in the y member variable of a
      MyFunct2(                                );

      // pass in the x member variable of b
      MyFunct2(                                );

      // What is the value of the member variable y in b here?


      return 0;
}
```

3. Synthesis questions [Do 4 of 5]   **[10 points each]**

i. Write a function `ClearBitRange` which will take three parameters, a `uint64_t` and two `ints` representing the start and size of a range. The function should clear all the bits in the range to 0. (The bits should be changed in the calling function after the function returns.)

ii. The threaded code below does not use locks except in the DLList. Show how locks can be added to make the code correct. Describe how to fix the code to make it efficient as well.

```
pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;

// assume that some code is putting work
// into the queue for the threads
DLList q;

const int numWorkItems = 98237523;
int itemsHandled = 0;

void *Worker(void *)
{
    threadWork myWork;
    while (itemsHandled < numWorkItems)
    {
        if (q.SafeGetBack(myWork))
        {
            // some work is done here
            // ...

            itemsHandled++;
        }
    }
    return 0;
}
```

iii. Write the operator+ code for the following class. Assume that all other functions have been defined already.

```
template <typename T>
class MyList {
public:
      MyList();
      ~MyList();
      MyList &operator+=(const MyList<T> &s);

      void AddBack(T val);
      T GetFront();
      void RemoveFront();
private:
      struct Node {
            T val;
            Node *next;
      };
      Node *front;
};

template <typename T>
MyList<T> &MyList<T>::operator+=(const MyList<T> &s)
{



}
```

iv. Given the following base class, write the full derived class definition for a carrot. (If you want more space, feel free to use the back of the page.)

```
class Vegetable
{
public:
    Vegetable(uint8_t *image);
    virtual ~Vegetable();
    virtual void GetColor(float &red, float &green, float &blue) = 0;
private:
    uint8_t *imageData;
};
```

Carrot.h: Carrot.cpp

v. Write a function which converts a `uint32_t` into a character string. For instance, MakeString(415) would return the c-string "415".

```
char *MakeString(uint32_t value)
{



}
```

4. Bonus Question (extra credit)   **[5 points]**

Overloading `operator[]` allows the array notation to be used with your class. Assignment is made possible by returning a reference an item in the data structure. How is this operator overloaded for bit arrays, when we cannot return a reference to a 1-bit type?