CS 215 – Study Questions 6
Overloaded operators and the make utility

1. In this lab, you overloaded operators for the WordData class. Given the following declaration of WordData objects:
   
   ```
   WordData A, B, C;
   ```
   
a. Write a segment of C++ code the uses the overloaded assignment ‘=’ operator for the WordData class.
b. Write a segment of C++ code the uses the overloaded equality ‘==’ operator for the WordData class.
c. Write a segment of C++ code the uses the overloaded less-than ‘<’ operator for the WordData class.
d. Write a segment of C++ code the uses the overloaded addition ‘+’ operator for the WordData class.
e. Write a segment of C++ code the uses the overloaded input ‘>>’ operator for the WordData class.
f. Write a segment of C++ code the uses the overloaded output ‘<<’ operator for the WordData class.

2. Why are the input ‘<<’ and output ‘>>’ operators declared as friends in the WordData class description?

3. Why does the assignment operator return a reference to a WordData object?

4. How does an assignment operator differ from a copy constructor?

5. The application program for Lab 6 contains a function called ReadFile:

   ```
   int ReadFile (ifstream & input, WordData * & Words)
   {
       Words = new WordData [MINWORDS];
       int size = MINWORDS;
       int count = 0;
       WordData aword;
       while (input >> aword)
       {
           if (count >= size)
           {
               WordData * temp = new WordData [size + BLOCKSIZE];
               for (int i = 0; i < size; i++)
                   temp[i] = Words[i];
               delete [] Words;
               Words = temp;
               size += BLOCKSIZE;
           }
           Words[count] = aword;
           count++;
       }
       return count;
   }
   ```
a. What is the data type for the parameter "Words"?
b. Why is the parameter "Words" passed by reference?
c. What is the purpose of the "if" statement inside of the "while" loop?
d. What is the purpose of the "for" loop inside of the "if" statement?
e. What is the purpose of the statement: Words = temp;
f. What is purpose of the value returned by this function?

6. In this lab, you created 3 source files: Lab06.h, Lab06.cpp, and Lab06app.cpp. What happens when you enter the compile command g++ Lab06.cpp? What happens when you enter the compile command g++ Lab06app.cpp?

7. In step 16 you used a makefile. What is a makefile? Why do we use makefiles?

8. The makefile you used for Lab 6 contains the following lines:

```
L06.out : Lab06.o Lab06app.o
g++ -o L06.out Lab06.o Lab06app.o

Lab06.o : Lab06.h Lab06.cpp
g++ -g -c Lab06.cpp

Lab06app.o : Lab06.h Lab06app.cpp
g++ -g -c Lab06app.cpp

clean :
    rm *.o L06.out
```

a. Which parts of this segment of a makefile are referred to as the "targets"?
b. Which parts are the "dependencies"?
c. Which part is the "rules"?
d. How are the "targets", dependencies" and the "rules" related?

9. In step 18, you replaced the word `class` in Lab06.h with the word `struct`. What happened when you compiled and executed the program?

10. In step 19 you commented out the words public: and private: from the struct. What happened when you compiled and executed the program?

11. In step 20, you replaced the word `struct` in Lab06.h with the word `class`. What happened when you compiled and executed the program?

12. In step 21, you uncommented the words public: and private: from the class to return to the original configuration. What happened when you compiled and executed the program?

13. What conclusions can you draw from steps 18, 19, 20, and 21?