COMP 3522
Object Oriented Programming 2

Midterm Exam

Full Name	
Signature	
Student ID	
Set	

Instructions:

1. No phones. Turn off your phone and put it away. If your phone rings you will earn zero on this exam.
2. This is a closed-book exam. No cheat sheets. You may not use any notes, texts, manuals, etc., during the exam
3. No electronic devices. No calculators. No computers. No pocket organizers or translators. No watches. No music players. No headphones. 4. No talking. Speaking during the exam is forbidden. If you speak, you will earn zero.
5. Keep your eyes on your own paper. Looking at another exam, or purposely exposing your exam to the view of other candidates will be considered cheating and you will earn zero.
6. This midterm is 90 minutes long.
7. You must write your answers on this exam. You may write on the back.
8. No candidate will be admitted after 30 minutes.
9. No candidate will be permitted to leave during the first 30 minutes.
10. NO QUESTIONS. Candidates are NOT permitted to ask questions during the exam except in the case of supposed typos.
11. Good luck. You've got this.

Final grade: PART A _____ / 66 + PART B _____ / 39 = _____ / 105

Part A: Theory (Write your answers in the space provided)

1. (4) What is a copy constructor? When is it used? Be specific. Code snippets may be used as (an) example(s).

2. (3) What is a reference? How is it different from a pointer?

3. (4) In C++, we may pass arguments to functions by value, by pointer, or by reference. What is the difference? Why would we choose one over the other?

4. (4) Suppose we have a pointer to a string called success. What is the difference between &success and *success? When would we use each of these?

5. (4) What is the difference between static allocation and dynamic allocation? You may provide code snippets to help illustrate the difference.

6. (4) What is a memory leak? How do we prevent memory leaks in C++? Write a short function to demonstrate code that generates a memory leak, and then 'insert' and underline some code that 'plugs' the leak.

7. (6) With respect to the const keyword:

    a. What is the purpose of making an argument to a function const?

    b. What is the purpose of adding the const keyword to a function header?

8. (3) We rarely invoke an object's destructor directly. When is an object's destructor invoked for us (automatically)?

9. (4) There are three ways to initialize variables in C++. What are they, and which method do we prefer to use? Why?

10. (2) When do we use the scope operator :: and when do we use the dot operator . in C++?

11. (2) What is the difference between the class and the struct in C++?

12. (3) What is forward declaration and what is its purpose?

13. (3) How do we make a class abstract in C++?

14. (2) Does C++ have interfaces? Explain.

15. (6) In C++, multiple inheritance may result in a class diamond-shaped inheritance pattern. a. How does this happen?

b. Does this cause any ambiguities with respect to member functions? How can the ambiguities be solved?

c. What is the key responsibility of the most-derived class at the bottom of the diamond?

16. (3) When we implement the assignment operator, we use the “copy and swap” algorithm. How does the “copy and swap” algorithm work? Can you identify an easy way to introduce an error into a “copy and swap” function?

Part B: Code (Write your answers in the space provided)

20. (7) What is printed to standard output by this code? HINT: this prints 14 lines.

```
#include <iostream>

using namespace std;

class A
{
public:
    A() { cout << "Default constructor!" << endl; }
    ~A() { cout << "Destructor!" << endl; }
    A(const A&) { cout << "Copy constructor!" << endl;}
};

A foo(A input)
{
    cout << "Look ma, I'm fooing!" << endl;
    return input;
}

int main()
{
    A littleA;
    foo(littleA);
    A anotherA = littleA;
    A yetAnotherA = foo(littleA);
}
```

21. (3) Suppose we want to develop and implement a class called Fraction. A Fraction stores a numerator and a denominator, both as integers. Write a Fraction constructor that throws an exception if the user tries to instantiate a Fraction with a denominator of 0 (zero). Assume there are two private data members, both of type int, called numerator and denominator, defined in the Fraction class.

22. (3) Implement a copy constructor for the Fraction class described in question 21

23. (2) Implement a destructor for the Fraction class described in question 21.

24. (3) Write an insertion operator for the Fraction class. The insertion operator is a friend function to the Fraction class, and it has the following signature:

```
ostream& operator<<(ostream& os, const Fraction& fraction) {




}
```

25. (3) Write an extraction operator for the Fraction class. The extraction operator is a friend function to the Fraction class, and it has the following signature:

```
istream& operator>>(istream& is, Fraction& fraction) {




}
```

26. (10) I have a text document called Fractions.txt. It contains rows (lines) of numbers. Each line contains 2 integers separated by whitespace. I would like to open the file. I would like to use each row to create a Fraction object.

Use the first integer in the line as the numerator, and the second integer as the denominator. I would like to store these new Fraction objects in a std::stack<Fraction>.

Write a program that does this. You may assume the code from the Fraction questions (21, 22, 23) is available to you. You may assume that there are no zeros in the file.

27. (4) What is printed to standard output by this code? HINT: this prints 14 lines.

```
#include <iostream>
class A
{
     public:
     A() { std::cout << "A allocated" << std::endl; }
~A() { std::cout << "A deallocated" << std::endl; } };

class B : public A
{
     public:
     B() { std::cout << "B allocated" << std::endl; }
~B() { std::cout << "B deallocated" << std::endl; } };

class C : public B
{
     public:
     C() { std::cout << "C allocated" << std::endl; }
~C() { std::cout << "C deallocated" << std::endl; } };

class D : public B
{
     public:
     D() { std::cout << "D allocated" << std::endl; }
~D() { std::cout << "D deallocated" << std::endl; } };

class E : public C, public D
{
     public:
     E() { std::cout << "E allocated" << std::endl; }
~E() { std::cout << "E deallocated" << std::endl; } };

int main()
{
     E e;
}
```

28. (4) What is printed to standard output by this code? Hint: this prints 12 lines. Explain in 1 sentence why the output is different from the output in question 27.

```
#include <iostream>
class A
{
     public:
     A() { std::cout << "A allocated" << std::endl; }
~A() { std::cout << "A deallocated" << std::endl; } };

class B : virtual public A
{
     public:
     B() { std::cout << "B allocated" << std::endl; }
~B() { std::cout << "B deallocated" << std::endl; } };

class C : virtual public B
{
     public:
     C() { std::cout << "C allocated" << std::endl; }
~C() { std::cout << "C deallocated" << std::endl; } };

class D : public B
{
     public:
     D() { std::cout << "D allocated" << std::endl; }
~D() { std::cout << "D deallocated" << std::endl; } };

class E : public C, public D
{
     public:
     E() { std::cout << "E allocated" << std::endl; }
~E() { std::cout << "E deallocated" << std::endl; } };

int main()
{
     E e;
}
```

END OF EXAM