// Name: Luxi Liang

// Seneca Student ID: 165936188

// Seneca email: lliang44@myseneca.ca

// Date of completion: 02/09/2020

OOP345 Lab04 Home Reflect

In this lab I learned how to work with compositions and aggregations. I had to go back and review how to write a move constructor and copy constructor in this lab.

I had difficulties with deallocating memory and using the deconstructor because originally I had used a shallow copy. However, when using composition, I needed to implement a

deep copy constructor so that my program wouldn't abort when the object went out of scope.

1) the difference between the implementations of a composition and an aggregation:

Composition is where two classes have a "" relationshhas-aip which results in both being dependant on the other.

Once one of the entities reaches the end, the other does too.

Implementation: we allocate a class within another class. When the "parent" class ends/deconstructs, the "child" class

will disappear as well because without its "parent" class, the child class doesn't exist.

An Aggregation doesn't manage the creation/destruction of objects that it uses. Either entity is able to be independant of one another. Therefore,

when one class ends, the other class is still able to continue.

Implementation: we allocate a class outside of another class and then passes a pointer to the "parent" class. Since the

two classes are independant of each other, even when we deallocate/deconstruct the "parent" class, the "child" class object

still exists and is able to continue functioning.

2) the difference between the implementations of move and copy functions in a composition and an aggregation.

Composition: copy and move function will be responsible for allocating and destroying the resources (destroying the resource only works for the move)

Aggregation: copy and move function are not responsible for allocating and destroying the resources. It will only copy/move the pointer.

\*\*\*Quiz 5 Reflect \*\*\*

Wrong Answers:

19) Code 6.0, the output of line 7 is:

Right:39

Mine: 5

24) The output of line 8 iss:

Right: 2

Mine: 5

Sub-object: where type points to instances of the class itself.

Class Luxi {

Luxi\* name; }

Instance Defintion: class-key Identifier {

Sub-object declaration

Member function declarations

} identifier

Ie. Class Luxi {

Unsigned num;

Char a [41];

} Luxi, \*pLuxi;