CIS 22B Lab 4 Itty Bitty Airfreight (IBA)

#### Lab 4

#### 500 Points

Topics:

Overloaded operators

Files

-----------------------------------------------------------------------------------------------------------------------------------------

The anti AI constraints:

Any C++ terms, constructs and/or methodologies covered in CIS 22A are available for your use.

You may only use additional C++ constructs or terms covered in lecture or zyBook (ebook) reading

assignments found in In modules 1, 2, 3, 4, 5, 6, 7 and 8. Deviation from these constraints lose points.

**Lab 4.1**

Utilizing the code from Lab 3, add an overloaded operator **==** which will be used to compare

two objects to see if they are equal. For our purposes, two objects are equal if their abbreviation

and uldid are the same. You’ll need to make your overloaded operator a friend of the Cargo class.

Create a unit1 object and load it with this data:

uld – Pallet

abbreviation – PAG

uldid – PAG32597IB

aircraft - 737

weight – 3321 **Pounds**

destination – SEA

Make a **copy** of unit1 called unit2 utilizing a copy constructor.

Create a default unit3.

Test with code like this in **main**:

if (unit1 **==** unit2)

cout << "\n unit1 is the same as unit2 \n";

else

cout << " \nunit1 is not the same as unit2 \n";

if (unit2 **==** unit3)

cout << " \nunit2 is the same as unit3 \n";

else

cout << " \nunit2 is not the same as unit3\n";

**Lab 4.2**

Using the code from lab 4.1, add the ability to read from a file.

Modify the input function:

\* Use the Cargo output function to print the Cargo object.

\* At the bottom of the input function, create a Cargo object named

**temp** using the constructor that takes six parameters.

\* Call the input function from the main function with no arguments.

Remove the rest of the code from main.

3. **Create a file and use it for input.**  You will be using the input file many times.

\* Create a file named **cardata4.txt** (or use the one provided) in Canvas.

**Pallet PAG PAG45982IB 737 4978 OAK**

**Container AYF AYF23409AA 737 2209 LAS**

**Combo Flat CFE CFE89023DL 737 5932 DFW**

\* All weights are in pounds, don’t worry about kilograms. You may

comment out or remove that code since user interaction with the

program will be limited to file name input.

\* In the input function, declare an object of type ifstream named

inputFile, which we will use to read from the file. Ask the user

for the name of the input file, and store it in a C++ string.

\* At the beginning of the code for the input function, open the

file. If the open fails, send a message to stderr and exit the

program. We will extend error checking in lab 5.

\* In all the reads within the input function, remove the user

prompt and read from the inputFile object, rather than reading

from the stdin object.

\* Use a loop to read each line from the file. To do this use a

while loop including all the reading in the input function, as

well as the building and output of the unit.

**Hint: you can do this with the following while statements**

while(getline(inputFile, cargostring))

{

istringstream cargoISS(cargostring);

cargoISS >> type1;

if ((type1.compare(comp1)) == 0)

{

cargoISS >> type2 >> abrv >> uld >> plane >> weight >> dest;

type1 = type1 + " " + type2;

if (!((type2.compare(comp2)) == 0))

{

cout << type1 << " bad unit name\n";

exit (99);

}

}

else

{

cargoISS >> abrv >> uld >> plane >> weight >> dest;

}

The code above is taken from the lab4protowcombo.cpp file in Canvas. This

code uses input stream processing and parsing to simplify data reading from a

file.

Put an eye catcher before the beginning of each function, class, and the global area:

// class name function name comment(if any)

Submit your .cpp files (both 4.1 and 4.2) via Canvas. Late labs lose points.