CIS 22B Lab 2 Itty Bitty Airfreight (IBA)

#### 500 Points

Topics:

heap

class

struct to class

constructors

destructors

mutators (setters)

accessors (getters)

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

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 and 5. Deviation from these constraints lose points.

Welcome to Itty Bitty Airfreight. We are a small, one airplane, local, airfreight company. We specialize in freight charters, special handling and quick turnaround shipments. If you need something delivered in the West, we are your solution. We do the smaller, local airports as well as the major airports.

What we need from you, the programmer is the following:

An object oriented, Unit Load Delivery (ULD), setup and tracking program. A Unit Load is one unit, either Container or Pallet to be loaded onto our Boeing 737. Here’s your data to set up and manage

Unit: Container or Pallet

Abbreviation: AYF, AYK, AAA, AYY type Containers and PAG, PMC, PLA type Pallets (a pallet is a flat platform for holding strapped down cargo).

Unit ID: Container or Pallet type + five digits + airline code; our ID code is IB, e.g. AYF12345IB

Aircraft type: Ours is a 737; we are planning on adding more planes in the future, of course.

Weight: The weight, in pounds, of the loaded container or pallet.

Destination: A three alpha character IATA string, e.g. MFR (Medford, OR), or PDX (Portland, OR)

### Lab 2 struct to class

Utilizing the code in lab 1, convert your struct to a class with **private** data. Use new to allocate space on the heap for your object. You will need to provide complete object oriented code to support your program. That means a class (see Cargo class below), a default constructor taking no arguments, a constructor taking six arguments, mutators (6 setters), accessors (6 getters), a destructor plus input and output functions.

In main, create a load object on the heap using the default constructor. Output the contents of the default object, then delete it.

In main, create another load object on the heap utilizing the default constructor, and then pass it

to input for data setup. Pass by reference is indicated here.

In the input function, **utilize the mutators (setters)** to set up your data:

uld – Container

abbreviation - AYF

uldid – AYF97326IB

aircraft - 737

weight - 1710

destination – SMF

Use **the accessors (getters)** in your output function to print out your object contents.

Destroy your unit object using delete

Hints:

Pass your object from main by reference. This allows your input and output functions to work much

more easily. You will need this methodology in later labs.

You can easily construct your uldid from the abbreviation plus the five digit plus two character airline owner, abbrev + five digit code + airline owner

Your output should look like this (your output data is different, of course)

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

Unit load type: Pallet

Unit load abbreviation: PMG

Unit identifier: PMG12345IB

Aircraft type: 737

Unit weight: 1257 pounds

Destination code: SJC

Cargo destructor called

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

class Cargo

{

private:

string uldtype;

string abbrev;

string uldid;

int aircraft;

double weight;

string destination;

public:

///Default constructor prototype

Cargo();

///Full constructor prototype

Cargo(const string uldtype, const string abbrev, const string uldid, const int aircraft, const int weight, const string destination);

///Destructor prototype

~Cargo();

///Mutator (setters) prototypes

void setuldtype(string);

void setabbrev(string);

void setuldid(string);

void setaircraft(int);

void setweight(int);

void setdestination(string);

///Accessor (getters) prototypes

string getuldtype() const;

string getabbrev() const;

string getuldid() const;

int getaircraft() const;

double getweight() const;

string getdestination() const;

input(Cargo &);

output(Cargo &);

};

Submit your .cpp via Canvas, remember, late labs lose points.