CSCI 3110 (Barbosa S18)
Project 3: On the High Seas

Due: **Wed Feb 21 by 11:59 PM** – may be turned in until Feb 28 by 11:59 PM with reduced points (per Project guidance found in the course syllabus).

Assignment ID: **proj3**
File(s) to be submitted: **proj3.zip**

Objectives: 1) Use inheritance to derive new classes; 2) Understand abstract base classes; 3) Overload base class functions; 4) Override base class virtual functions; 5) Define pure virtual functions; 6) Understand static and dynamic function call binding; 7) Use custom destructors to free dynamically allocated memory

Project description:
In this assignment you will create a class representing a ship, and create two additional classes that inherit from the *Ship* class, a *CargoShip* class and a *CruiseShip* class. The project is geared to give you practical experience with characteristics and C++ implementation details of inheritance and polymorphism.

Requirements:

1. Your program must be split into 7 files. There will be 3 classes (each with separate interface and implementation files), and a driver file. The requirements for these are specified below:

a) The *Ship* class – This is an **abstract** class

- Files must be named ***ship.h*** and ***ship.cpp***
- Class must be named ***Ship***
- Must contain #include guards for SHIP_H
- Will have these protected members
  - i. A string containing the ship's name
  - ii. A double representing the amount of fuel on board (in tons)
- Must have these public members
  - i. A two parameter constructor that takes the ship's name and fuel load (in that order)
  - ii. A void function, ***fuel***, that outputs the ship's name and its fuel load (see sample output)
  - iii. A **virtual** void function, ***sail***, that prints a generic message indicating that the ship is underway
  - iv. A **pure-virtual** void function, ***load***, that loads the ship with tonnage (cargo) or passengers (cruise)

b) The *CruiseShip* class – This is a ***derived*** class that inherits from the *Ship* class as public

- Files must be named ***cruiseship.h*** and ***cruiseship.cpp***
- Class must be named ***CruiseShip***
- Must contain #include guards for CRUISESHIP_H
- Will have these private members
  - i. Three doubles that indicate the percentage of passengers in Luxury, Upper Deck, and Lower Deck cabins respectively. These values should sum to 1.
  - ii. Three integers that indicate the number of passengers in Luxury, Upper Deck, and Lower Deck cabins respectively.

- Must have these public members
  i. A five parameter constructor that takes the ship’s name, fuel load, and the percentages of passengers in Luxury, Upper Deck, and Lower Deck (in that order) – The percentages should sum to 1 – This constructor must pass parameters to the constructor in the base class
  ii. A void function, ***sail***, that **overrides** the virtual function in the base class, and prints a message indicating that the cruise ship is underway, with a breakdown of the number of passengers in each cabin level (see sample output)
  iii. A void function, ***load***, that accepts an integer representing the number of passengers as a parameter, and distributes those passengers among the three cabin levels (Luxury, Upper Deck, Lower Deck) per the percentages specified

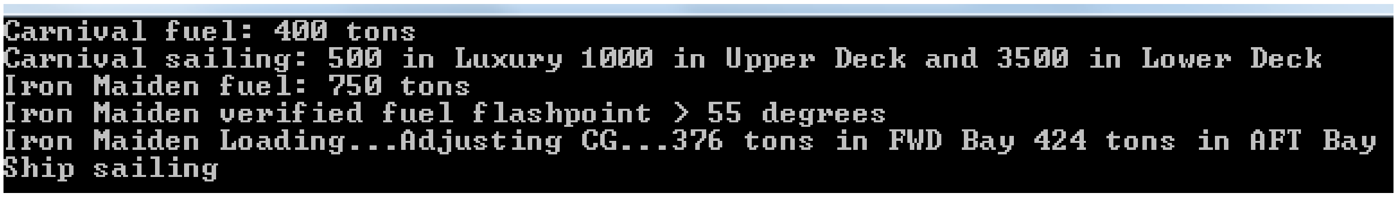
c) The *CargoShip* class – This is a ***derived*** class that inherits from the *Ship* class as public

- Files must be named ***cargoship.h*** and ***cargoship.cpp***
- Class must be named ***CargoShip***
- Must contain #include guards for CARGOSHIP_H
- Will have these private members
  i. Two double pointer variables representing tonnage in the forward cargo bay and the aft cargo bay
  ii. An integer that indicates the maximum cargo capacity for the ship (in tons).
- Must have these public members
  i. A three parameter constructor that takes the ship’s name, fuel load, and cargo capacity (in that order) – This constructor should dynamically allocate the memory to represent the cargo bays and initialize them to zero , and must pass parameters to the constructor in the base class
  ii. A destructor that frees the memory allocated for the cargo bays
  iii. A void function, ***fuel***, that **overloads** the function in the base class, and has an integer parameter that indicates the minimum flashpoint for fuel used in cargo ships – This function outputs that the flashpoint has been verified to be above the minimum, in addition to the fuel quantity message output by the base class (see sample output)
  iv. A void function, ***load***, that accepts an integer representing the tons of cargo to be loaded on the ship – you must check to ensure that only an amount up to the capacity of the ship is loaded – This function should place 47% of the cargo in the forward bay and the remainder in the aft bay, and should output a message that indicates that loading and center of gravity (CG) adjustment have taken place and show the cargo tonnage distribution between the forward and aft cargo bays (see sample output)

c) A driver, or client, file

- Must be named ***proj3.cpp***
- Must have two functions as shown below – *main* must be defined first
  - *sailShip* – This function simulates the ship getting underway with a single function call
    - It has a single parameter: a reference to a ***Ship*** object
    - It invokes the sail function through this object
  - *main* – This function performs the following steps
    - Instantiates a *CruiseShip* object with initialization of your choosing, outputs the amount of fuel on the ship, loads the ship with a passenger count of your choosing, and invokes the sailShip function with the *CruiseShip* object as an argument
    - Instantiates a *CargoShip* object with initialization of your choosing, outputs fuel information (amount and confirmed minimum flashpoint), loads a cargo tonnage amount of your choosing, and invokes the sailShip function with the *CargoShip* object as an argument

2. The format of your output should match the output in this specification. This means the verbiage, line spacing, character spacing (do not use tabs – only a single space between items). An example screenshot is shown below:



```
Carnival fuel: 400 tons
Carnival sailing: 500 in Luxury 1000 in Upper Deck and 3500 in Lower Deck
Iron Maiden fuel: 750 tons
Iron Maiden verified fuel flashpoint > 55 degrees
Iron Maiden Loading...Adjusting CG...376 tons in FWD Bay 424 tons in AFT Bay
Ship sailing
```

3. Test your program - Use different initializations.

4. Code comments - Add the following comments to your code:

- A section at the top of the source file(s) with the following identifying information:

    Your Name
    CSCI 3110-00X (your section #)
    Project #X
    Due: mm/dd/yy

- Below your name add comments in each file that gives an overview of the program or class.
- Place a one or two line comment above each function that summarizes the workings of the function.