## Program Purpose

For this program you are asked to use C++ inheritance to create a base class and two derived classes. You will also be working with constructors and redefining/overloading class member functions.

## Mandatory Instructions

1. Design a *Ship* class that has the following members:
   - A member variable for the name of the ship (string)
   - A member variable for the year the ship was built (int)
   - A default constructor that sets the name to the empty string and the year to zero
   - An overloaded constructor that accepts values for the name and year and assigns them to the data members
   - Accessors and mutators for the name and year
   - A virtual display function **displayShip** that displays the ship's name and the year it was built

   Store the declaration for this class in a file called **prog7ship.h**. Implement functions in .cpp file of same name.

2. Design a *CruiseShip* class that is derived from the *Ship* class with:
   - A member variable for the cruise departure date (string)
   - A member variable for the maximum number of passengers (int)
   - A member variable for the number of passengers signed up for the cruise (int)
   - A default constructor that invokes the *Ship* default constructor to initialize the inherited data members and sets the *CruiseShip* fields to zero or the empty string.
   - A second constructor that accepts a cruise ship's name, year, departure date, maximum number of passengers and number signed up. Pass the first two arguments to the *Ship* constructor and use the last three to set the *CruiseShip's* data members.
   - A member function to calculate the percentage filled for the cruise (number signed up / maximum number of passengers)
   - A **displayShip** function that overrides the **displayShip** function in the base class. It should display the ship's name, year, departure date, maximum number of passengers, passengers signed up for the cruise and the percentage filled for the cruise.

   Store the declaration for the *CruiseShip* class in the file **prog7cruiseship.h**. Implement functions in .cpp file of same name.

3. Design a *FerryShip* class that is derived from the *Ship* class with:
   - A member variable for the maximum number of passengers (int)
   - A member variable for the maximum number of cars (int)
   - A default constructor that invokes the *Ship* default constructor to initialize the inherited data members and sets number of passengers and cars to zero.
   - A second constructor that accepts a ferry ship's name, year, people and car capacity. Pass the first two arguments to the *Ship* constructor and use the last two to set the *FerryShip's* data members.
   - A **displayShip** function that overrides the **displayShip** function in the base class. It should first call the base class's function (to display the ship's name and year) and then display the ship's data members.

   Store the declaration for the *FerryShip* class in the file **prog7ferryship.h**. Implement functions in .cpp file of same name.

4. Write a client program called **prog7client.cpp** to demonstrate the three ship classes.

   - Declare a *Ship* object initializing the data members with "Flying Dutchman" and "1795".

   - Declare a *CruiseShip* object with arguments for all data members using the values "Titanic", 1912, 4/10/12, 2200 and 1324.

   - Declare a *FerryShip* object initializing the data members with "Barberi" 1981, 6,000 and 0.

   - Call the **displayShip** function to display the information about each of these objects.

```
--------------------------------------
Ship:                   Flying Dutchman
Year:                   1795

--------------------------------------
Ship:                   Titanic
Year:                   1912

CRUISE SHIP
Departing:              4/10/12
Max # of passengers:    2200
# Passengers sailing:   1324
Percent full:           60.18%
--------------------------------------
Ship:                   Barberi
Year:                   1981

FARRY SHIP
Max # of passengers:    6000
Max # of cars:          0
--------------------------------------

Press any key to continue . . .
```

5. When your program is working correctly, define an array of pointers to *Ship* objects to store pointers of up to fifteen *Ship*, *CruiseShip* or *FerryShip* objects.

   Then read in the data from the file **prog7.txt**. For each set of ship data it contains a code to indicate whether the data that follows is for a "S"hip, "C"ruise ship or "F"erry ship.

   - If the code is "S" then the code is followed by the ship's name and year.

   - If the code is "C" then the ship's name, year, date of departure, maximum number of passengers and number of passengers signed up for the cruise are also included in the data file.

   - If the code is "F" then the code is followed by the ship's name, year, people and car capacity.

   Use the **new** operator to create either a *Ship*, *CruiseShip* or *FerryShip* object, storing the pointer to the object in the next element of the array.

6. In your header file for the *Ship* class, make your **displayShip** function virtual. Then write a loop to display the data for all of the data read in and stored in the array.

**Program Documentation & Style**

sure your name, class and the due date are included at the beginning of each file. The standard header comments with the sections labeled *Purpose*, *Input*, *Processing*, *Output* should be included in the client program file. Brief comments describing the class member functions should be in the class specification file.

## Program Documentation

Provide documentation header for your program and for each function other than main().

```
// Program        7
// Description:   Products program
// Programmer:    Your name
// Class:         CS 2020, Spring/Fall 20xx
```

```
// Function:      Function name
// Description:   Function purpose
// Programmer:    Your name
// Class:         CS 2020, Spring/Fall 20xx
// Parameters:    list and describe
// Returns:       describe
```

## What to turn in?

Make sure you have completed your program according to the specifications given.
Make a printout of your program by typing in the following commands at the prompt:

```
$ photo  prog7.log
$ ls  –l
$
$
$ cat  prog7ship.h
$ cat prog7ship.cpp
$ cat  prog7cruiseship.h
$ cat prog7cruiseship.cpp
$ cat  prog7ferryship.h
$ cat prog7ferryship.cpp
$ cat  prog7client.cpp
$ g++ prog7client.cpp   prog7ship.cpp   prog7cruiseship.cpp   prog7ferryship.cpp
$ .\a.out
$ Ctrl+D
$ exit
```