Program Purpose

Using Visual Studio 2017 create a CLR console application. Create a program plan and then convert it into C++ statements. Practice debugging, declaring variables, type casting, formatting output, if statements and input /output to / from the console window, file I/O, functions, loops, and arrays.

Use for your reference

1. Gaddis' book, in-class exercises and your class notes.
2. This assignment sheet & the grade sheet for this lab already printed out.
3. USB Flash drive(s) or other storage media.

LATE PROGRAMS will be not be accepted. Please be sure to upload compressed solution and Word document grade sheet to Canvas by due date/time.

Group Instructions

This program will be a GROUP project. Each group will need at least 3 students in it. I will form your groups by randomly selecting students from Canvas class list.

Each group member will be responsible for writing the solution to the program using stubs and drivers for the functions other members are responsible for and submitting that solution for themselves. In addition, the group will be responsible for putting together all the individual parts and turning that in as a combined group solution. See below for explanation of each individual group member's responsibilities.

Every member will: write main() and getShipData(). Create stubs for functions written by other members as needed. Stubs should display the headers of each report section ONLY. Place documentation header on each function/stub. For the functions you are writing use drivers to test your code.

Member 1: in addition to main() and getShipData() write showShipCategories() and getCategory()
Member 2: in addition to main() and getShipData() write showShipCapacity() and getAveCapacity()
Member 3: in addition to main() and getShipData() write showShipSpeed() and getHighestSpeed()

Mandatory Portion Instructions

The Royal Caribbean cruise line would like a report on all their ships. Given the ship name and other data, write a C++ program to print a report showing the information described below.

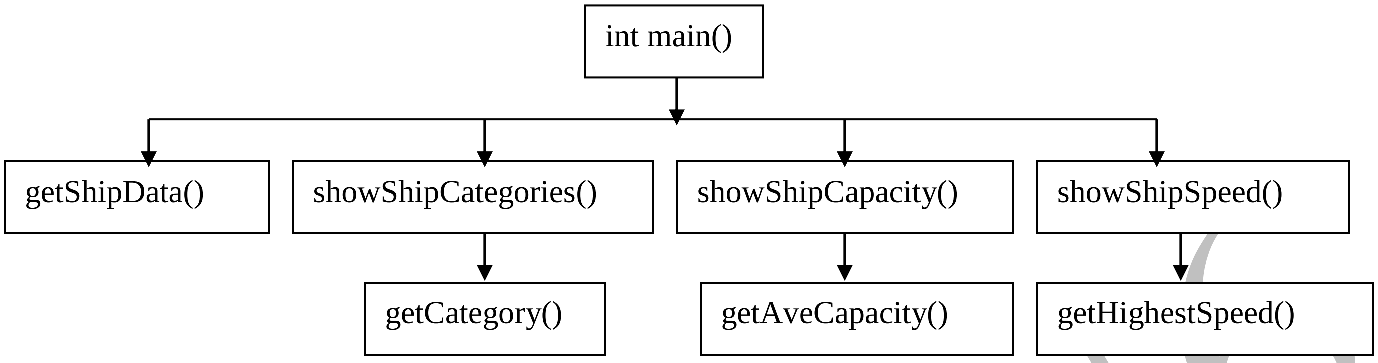
Input: Information about the fleet is stored in a file called **fleet.dat** and available for download on Canvas. Each line in the file contains ship name, capacity, gross tonnage, and top cruising speed in that order. The file may contain any number of ships. Read data while not end of file, i.e., `while(!infile.eof()){}`. Do not count the lines of data!!!

You will need several arrays to store all the ship data. You should reserve space for 30 ships in your arrays. Sample line from the data file is shown below. Assume data file will NOT have more than 30 lines of data **but** it could have less.

```
Freedom   3634    160000     21
```

Your main() should call the functions described below. Each function should have a function documentation header.
Hierarchy of the program:

Program 6 [35 points] DUE: On Canvas



1. Write a function, getShipData(), to read in the ship data from the file **fleet.dat** and store it in four arrays, one for the ship names, one for the capacity, one for the tonnage and one for the speed. Put the data for the first ship in element 0 in each array, the data for the second ship in element 1 of each array, etc. Return the actual number of ships read to the caller. Pass all arrays as parameters to this function. Close the data file.

   NOTE: All further processing of the data will be done with the data stored in the arrays, not the data file.

2. Write a function, showShipCategories(), to show the ship data stored in the arrays in the console window. Include a title for the report and a line with column headings for the ship type/category, ship name, capacity, gross tonnage and top cruising speed. For each ship print the ship type/category (see table in part 3), ship name, capacity, tonnage and speed. Call function in step 3 to assign ship category to a ship.

3. Write a function, getCategory(), that accepts a SINGLE ship's gross tonnage and returns the category name according to the table below. This function DOES NOT have a loop in it!!!

Gross Tonnage	Ship type/category
< 48,600	Empress Family
< 74,000	Sovereign Family
< 90,100	Vision Family
< 138,000	Radiance Family
< 155,000	Voyager Family
< 160,666	Freedom Family
> 160.666	Oasis Family

3. Write a function, getAveCapacity(), to find and return the average ship capacity for all ships. Ship data is passed via parameters.

4. Write a function, showShipCapacity(), to show the name and capacity for all ships with a capacity greater than the average. Insert a few blank lines to separate this part of the report from the previously printed data. Then print a title ("Ships with Above Average Capacity "), underneath the title print the average capacity found by the previous function ("Average Capacity = 999999.9") followed by a line with column headings for the ship name and capacity. Then list the name and capacities for each ship with capacity above the average.

5. Write a function, getHighestSpeed(), to find and return the array **index** for a ship with the highest cruising speed.

6. Write a function, showShipSpeed(), that will display the name(s) and cruising speed of ship(s) with the highest cruising speed.

<u>* READ* THIS: Naming of projects:</u>

Individual member solution: Last_Program6_MemberX – where X is your member number (1,2,3, etc.)

Combined solution should be named: Combined_LastMember1_LastMember2_LastMember3

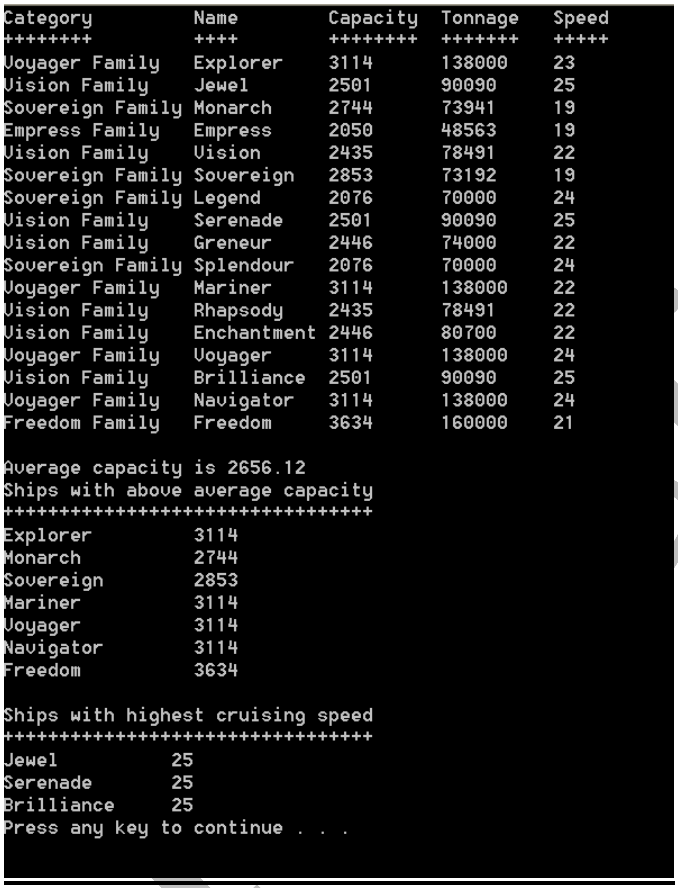
<u>Example:</u>
Group members:

Brown – member 1, Gray – member 2, Black – member 3

Brown_Program6_Member1
Gray_Program6_Member2
Black_Program6_Member3

Combined_Brown_Gray_Black

Sample program output (May be different from your data file)



```
Category         Name        Capacity  Tonnage   Speed
++++++++         ++++        ++++++++  +++++++   +++++
Voyager Family   Explorer    3114      138000    23
Vision Family    Jewel       2501      90090     25
Sovereign Family Monarch     2744      73941     19
Empress Family   Empress     2050      48563     19
Vision Family    Vision      2435      78491     22
Sovereign Family Sovereign   2853      73192     19
Sovereign Family Legend      2076      70000     24
Vision Family    Serenade    2501      90090     25
Vision Family    Greneur     2446      74000     22
Sovereign Family Splendour   2076      70000     24
Voyager Family   Mariner     3114      138000    22
Vision Family    Rhapsody    2435      78491     22
Vision Family    Enchantment 2446      80700     22
Voyager Family   Voyager     3114      138000    24
Vision Family    Brilliance  2501      90090     25
Voyager Family   Navigator   3114      138000    24
Freedom Family   Freedom     3634      160000    21

Average capacity is 2656.12
Ships with above average capacity
+++++++++++++++++++++++++++++++++
Explorer         3114
Monarch          2744
Sovereign        2853
Mariner          3114
Voyager          3114
Navigator        3114
Freedom          3634

Ships with highest cruising speed
+++++++++++++++++++++++++++++++++
Jewel          25
Serenade       25
Brilliance     25
Press any key to continue . . .
```

Program Documentation & Style:

1. Declare all variables and constants that your program uses at the beginning of your program.
2. Your program should include two types of comments:
   Header Comments at the top including lines with:
   - Your name, course name, and class time
   - Program assignment number, program file name (pgm5.cpp) and due date
   - A sentence or two explaining the purpose of the program
   - A description of the input data needed by the program when you run it

- A description of the processing (calculations) done by the program
- A description of the results (output) produced by the program

b. In-line comments: There should be an in-line comment for each main step in your program. In general, this means a comment with each group of C++ statements that handles the input, the processing and the output steps of your program.

3. Use meaningful identifier names
4. Include clear prompts for the user about entering the data.
5. Include clear descriptions of the results when you display them.

What to turn in?

1. Log into Canvas, locate this assignment, and upload the compressed project folders.
2. Upload the grade sheet (Word document) after you have edited it to provide requested information.
3. You're done.