CS202 Assignment #2

Purpose:
Learn to implement a C++ class, including constructors and destructors.

Introduction:
YouTube video Introducing the assignment Video Briefing on AS 2

Supplied:
The main.cpp program is provided as the driver for the assignment. The class definition is provided along with the calls necessary for the program to run. You may only modify the class definition if you choose to implement your own algorithm to parse the data. In this case, you may remove the optional functions and add your own. Do not modify the main(). Students may use reuse any portion of **their** code, or the provided code from the Assignment 1 for this assignment. The supplied database file used is the same as from assignment 1.

Expected Submissions:
main.cpp with the students functions incorporated. The file should be submitted online via the class Canvas website. **No late submissions will be accepted.**

Problem:
This program will implement the same functionality as Assignment 1, but as a C++ class. The class definition is provided and should not be modified except as specified. Refer to the Assignment 1 write up for program functionality.

AISClass
- makeStringUpper(string s) : string
- getNextField(string &line, int &index, string &subString): bool
- distanceTraveled(int first, int last) : void
- findLastOccurrance(string mmsi) : int
- findFirstOccurrance(string) : int
- stringConvert(string) : double
- addUniqueString(string value) : void
- saveField(int fieldNumber, string subString, AISType &tempItem) : void
- openInputFile(ifstream &) : bool
- readFile(ifstream &) : void
- AISData : vector<AISType>
- uniqueMmsi : vector<string>
- recordCount : int
- printRecord(int) : void
- searchForVesselByName (string vesselName) : int
+ AISClass()
+ ~AISClass()
+ readDataBase() : void
+ newSearch( ) : void

Detailed Function Descriptions
void readFile(ifstream & inFile)
inFile – already open file to be read

This function performs similarly to the version from assignment 1, but stores the data in the class variable vector *AISData*, and the count is saved in member *recordCount*. The same algorithm may be used to parse the file,or students may implement their own algorithm.

Optional
bool getNextField(string &line, int &index, string &subString)
string &line – the line of data read from the file that needs to be parsed
int &index – the current starting position of the parsing. The first time this function is called for a new line, index should be set to zero. The function should update the index before returning, so that on the next call it will look at the next field.
string &subString – the parsed string

return value – *true* if more data is available, *false* if the whole string has been parsed.

This function is the same as in Assignment 1, but is a member of the class now. This function may be omitted if students develop their own algorithm to parse the input data file.

bool openInputFile(ifstream& inFile)
ifstream& infile – file stream variable for the file to be opened.

This function is the same as assignment 1 but a member of the class now.

void printRecord(int recordNumber)
int recordNumber The element number of the record to print from *AISData*.

The function is similar to the version from Assignment 1, but is passed the element number instead of the record itself. All fields for the record are printed. See example output for formatting.

int searchForVesselByName(string vesselName)
string vesselName – the string to be searched for in the vessel names

return value – the number of records found that match the passed string, *vesselName*.

This function is very similar to Assignment 1 but is a member of the class now. The function searches the member vector, *AISData* for possible matches, and adds vessels found to the member vector *uniqueMmsi* by calling *addUniqueString().*

string makeStringUpper(string s)
string s – the string to be converted to upper case.
return value – upper case version of passed string.

This function converts the passed string to upper case and returns it. The library function *toupper()* may be called by this function. This is the same as Assignment 1 but is now a member of the class.

double stringConvert(string s)
string s – the string to be converted into a double

return value – the double converted from the string

This function converts a string to it's corresponding double value. Implementation of the function must use string stream. All other implementations are not allowed and will have significant point deductions. This is the same as assignment 1 but now is a member of the class.

int findLastOccurrance(string mmsi)
string mmsi - the mmsi value being searched for

return value – the index of the last record in the vector that contains the *mmsi* value passed in.

This function is very similar to the Assignment 1 function, but searches the member vector *AISData* for the matching vessel.

int findFirstOccurrance(string mmsi)
string mmsi - the mmsi value being searched for

return value – the index of the first record in the vector that contains the *mmsi* value passed in. This function is very similar to the Assignment 1 function, but access the member vector *AISData* for the matching vessel.

void addUniqueString(string value)
string value – the string to be added

This function is very similar to the version from Assignment 1, but adds the string to the member vector *uniqueMmsi*.

Optional
void saveField(int fieldNumber, string subString, AISType &tempItem)
int fieldNumber – the number of the field, starting at zero
string subString – the value to be saved in the field, may require conversion to double inside the function.
AISType &tempItem- the record to which the field will be added

This function is the same as Assignment 1. If students choose to implement their own algorithms to parse the data, this function may be omitted.

void distanceTraveled(int first, int last)
int first – index of starting location record
int last – index of ending location record

This function is similar to the version from Assignment 1, but access the member vector *AISData* to retrieve the values used for the distance calculation. The function also output the distance calculated.

Functions New in Assignment 2
void newSearch( )
This function handles a large part of what was handled by main in Assignment 1. After the file has been open and read in, all user interaction occurs in this function. The user is prompting for the vessel name to be searched for, exiting if ‘q’ is entered, calling *searchForVesselByName( )*, asking the user if they want to see the first record for the vessel, calling *printRecord()* if required, asking if they want to calculate the distance, and calling *distanceTraveled().*

Notes:

- This function loops until the user enters ‘q’ at the prompt for the vessel name to search for.
- The member vector *uniqueMmsi* should be cleared at the beginning of a new search.

AISClass()
The default constructor for the class should initialize member *recordCount* to zero.

~AISClass()
The default constructor should clear both member vectors, *AISData* and *uniqueMmsi*. The message “Memory returned for vectors” should be output.

void readDataBase()

This function performs a portion of what was done in *main()* in Assignment 1. It calls *openInputFile()*, and exits the program if a *false* is returned from the open function. If *true* is returned, then *readFile()* is called to read the data into the member *AISData*, lastly the number of records that were read is output.

Example Output

```
dolly@Snoopy:AS2$ g++ main_solution.cpp -Wall -pedantic
dolly@Snoopy:AS2$ ./a.out
Enter input File Name/(q-quit): AISData.csv
File opened correctly
----------------------
1800000--- End of file reached ---Items read: 1899355
1899355 records read
Enter vessel name: bill
1543 vessel records containing name "bill", Unique vessels: 2
*******************************************
Would you like to display the first record of each vessel? [y/n] y
************************
MMSI:         367007060
Base Date Time: 2019-01-01T00:00:00
Lattitude: 30.2898 Longitude: -91.2234
SOG: 0.2
COG: -196.3
Heading: 511
Vessel Name: BILL S
imo: IMO7030963
Call Sign: WDC3383
Vessal Type: 31
Status: 15
Length: 30
Width: 8
Draft: 3.6
Cargo: 0
Transceiver Class: B


************************
MMSI:         366751770
Base Date Time: 2019-01-01T11:27:05
Lattitude: 49.9244 Longitude: -125.12
SOG: 6.5
COG: 131.1
Heading: 132
Vessel Name: BILLIE H
imo: IMO8964719
Call Sign: WCY4992
Vessal Type: 31
Status: 0
Length: 27
Width: 8
Draft: 4.7
Cargo: 32
Transceiver Class: B


Would you like to find the distance traveled for a vessel? [y/n] y
MMSI for vessel: 366751770
***************
Vessel: "BILLIE H" MMSI: 366751770  Trip Starting time: 2019-01-01T11:27:05
Distance traveled from (49.9244, -125.12) to (49.1369, -123.534) 89.5485 Miles

Enter vessel name: maersk
15706 vessel records containing name "maersk", Unique vessels: 29
```

```
*******************************************
Would you like to display the first record of each vessel? [y/n] n
Would you like to find the distance traveled for a vessel? [y/n] n
Enter vessel name: mearsk
0 vessel records containing name "mearsk", Unique vessels: 0
*******************************************
Enter vessel name: atomic
363 vessel records containing name "atomic", Unique vessels: 1
*******************************************
Would you like to display the first record of each vessel? [y/n] y
************************
MMSI:         319071600
Base Date Time: 2019-01-01T00:04:11
Lattitude: 26.7489 Longitude: -80.0499
SOG: 0
COG: 186.5
Heading: 182
Vessel Name: ATOMIC
imo: IMO1009807
Call Sign: ZGEG8
Vessal Type: 37
Status: 5
Length: 45
Width: 9
Draft: 2.8
Cargo:
Transceiver Class: A


Would you like to find the distance traveled for a vessel? [y/n] y
MMSI for vessel: 319071600
***************
Vessel: "ATOMIC" MMSI: 319071600  Trip Starting time: 2019-01-01T00:04:11
Distance traveled from (26.7489, -80.0499) to (26.7489, -80.0499) 0.00151337 Miles

Enter vessel name: nothing
0 vessel records containing name "nothing", Unique vessels: 0
*******************************************
Enter vessel name: q
Memory returned for vectors
dolly@Snoopy:AS2$

dolly@Snoopy:AS2$ ./a.out
Enter input File Name/(q-quit): q
Error opening file
Exiting....
dolly@Snoopy:AS2$
```