CS23820 Assessed Assignment 2017-2018 "Counting the Cetacean Mammals of Cardigan Bay".

David Price and Neal Snooke

November 9, 2017

1 Monitoring Dolphins and Porpoise in the sea area between Strumble Head and Braich-y-Pwll

You have been contracted to produce a software system to assist marine biologists who are interested in monitoring the cetacean mammals of Cardigan Bay. While several species of cetacean mammals are present in the bay, this project focusses on Harbour Porpoise and Bottlenose Dolphins.

The observers have been equipped with modern accurate equipment. The equipment allows an observer to give a bearing and a distance to the sighted cetacean mammal and state whether a Harbour Porpoise or a Bottlenose Dolphin has been sighted. The equipment (and the observers) are not perfect and so some errors will exist in the information they report.

The sea area is depicted on the Eastern section of the Admiralty Chart number 1410 titled "Saint George's Channel". Some observers are on shore but others are located on boats at sea.

NOTE: While much of the information supplied in this assignment specification is closely related to real issues of monitoring marine mammal populations, various simplifications have been made to reduce the complexity of the problem.

The software system must consist of two programs, one written in C and one in C++, that interact via files. We require that your programs are compliant with recognised versions of ANSI C or ISO C++ as described below. Our "official environment" in which all your user programs must run, and in which we will test them, is that provided by CLion as running on the Linux installation in our labs.

2 Your Missions

The following sections describe the functionality you must implement. You will need to consider a number of details such as the exact format that has been specified for the data files, how to represent locations (latitude and longitude) and how to perform some related calculations. These technical details are included in the referenced Appendices.

2.1 The ANSI C Program

The first program, which must be written in ANSI C, requires that you analyse a set of **sightings** of cetacean mammals and calculate the implied locations (latitude/longitude) of each mammal.

2.1.1 Locate the mammals

For this first section you can ignore the boundaries of "our sea area" and process the data for all the Dolphins and Porpoise.

You are required to have the following features in your program.

1. The program must prompt for the name of a file which contains details of all the observer locations that are active in our sea area.

   Your program must open the file (see Appendix B.1), read in all the data and store it in a suitable set of data structures and variables.

2. The program must prompt for the name of a file which contains the sightings of cetacean mammals.

   Your program must open the file (see Appendix B.2), read in all the data and store it in a suitable set of data structures and variables.

3. Your program must now calculate the locations of all the cetacean mammals as sighted by the observers. Refer to Appendix A for details about locations and how the calculation is to be performed.

4. The program must prompt for the name of a file in which it should save the cetacean mammal locations.

5. Your program must write the locations of the cetacean mammals and their types to the specified file. The file must have the entry for one cetacean mammal on each line in the format:

   `T lat long`

   where `T` is replaced by `D` (for Bottlenose Dolphin) or `P` (for Harbour Porpoise) as appropriate and `lat` and `long` are real numbers giving the latitude and longitude in degrees.

   An example file created by your program for just two mammals might be:

   ```
   D 52.750254 -4.300645
   P 52.100213 -4.700281
   ```

2.1.2 Locate the mammals that lie within our sea area

For this section you must extend your code so as to ignore reports of any Dolphins or Porpoise that are outside the boundaries of "our sea area" as specified at the end of Appendix A.1.

The file containing cetacean mammal locations that you create will thus sometimes be smaller than that created by your solution to 2.1.1 for some of our data sets.

2.2 The C++ Program

The second program, which must be written in ISO C++ 11 or higher, requires that you analyse the calculated locations of cetacean mammals and decide how many mammals have really been sighted. That is, you decide whether some sightings actually refer to the same mammal and if so, calculate **corrected** locations. The program must then analyse the corrected locations of the cetacean mammals and decide how many separate pods of mammals are present in the bay.

The program must prompt for the name of a file which contains the initial calculated locations of the cetacean mammals. The format of that file will be that created by your C program and as described as the end of section 2.1.1 above.

2.2.1 True locations of the cetacean mammals

For this feature you must analyse the locations of the cetacean mammals that have been sighted and decide which reports refer to the same animal.

Two (or more) sighting reports are considered to refer to the same mammal if the calculated locations are within 0.02 of a nautical mile (approximately 40 metres) of each other.

You must set the "true" location of the cetacean mammal to the numerical average of the multiple sightings.

This C++ program must read in the file of cetacean mammal locations produced by the C program and from that calculate the set of true locations of each mammal.

2.2.2 Identifying the pods of cetacean mammals - 'the Pod Mission'

Cetacean mammals often swim in groups referred to as pods. For the purposes of our program, a cetacean mammal is considered to be in a pod if it is within 0.1 of a nautical mile of any other cetacean mammal of the same type within a pod.

For this feature you must calculate how many pods of cetacean mammals are active in our area at the time of the sightings. For each pod you must display how many animals are within the pod and the location of each of the mammals. You must present your results in a suitable tabular form.

2.2.3 Code that we Supply

We are providing you with a C++ class that represents locations and calculates the shortest distance between two locations. You are **required** to use this class in your code.

The implementation of the distance calculation is based on the calculation as described on the web page `http://en.wikipedia.org/wiki/Great-circle_distance`.

The documented header and implementation files (named `Location.h` and `Location.cpp`) are available for download from the Blackboard course web site. The header file is also in Appendix C.

3 The Material You Must Submit

Your solution to this assignment must be submitted on AberLearn Blackboard before 2:00pm on Friday 15th December 2017. There will be appropriate links in the Assignments part of the course materials for module CS23820.

You must submit a `zip` file containing the following files:

- **A README file** that describes what is what. This is a plain text file (not a MS Word document, not a pdf document). In particular this file must give the names of the various files you hand in and what they correspond to. This must also describe how to build your code, including a mention of standard libraries/packages and the versions you used.
- **The source code of your two programs**. You must submit these as two CLion project exports. Make sure that all the necessary files are included. Do not include standard libraries.
- **A screencast**, showing:
  1. a clean building (Run ⇒ Clean followed by Build) of the code of your two programs, with all errors (hopefully none) and warnings (hopefully also none — remember in almost every case warnings are really caused by bugs!);
  2. the running of your code, clearly showing the output.

  The screencast must include voice to explain what is going on. Before compiling your code for the screencast, make sure you clean it (Run => Clean) so that the compilation does do something. Also, make sure you choose the correct compilation options so that warnings are given. Alternatively, this can be one screencast per program. The screencast or casts must run for no more than 10 minutes in total.
- **A document** (maximum two pages long) that explains what you have done and justifies your design.

4 Assessment Criteria

The most appropriate "assessment criteria" are those in Appendix AA of the student handbook, namely those for "Assessment Criteria for Development".

The usual requirements for coding projects apply, namely that programs should be well commented with comments that add real value and do not just, in essence, duplicate code. Programs should have good layout and must use meaningful names for variables, classes and other identifiers.

Finally, part of the evaluation is done on the compilation and whether your code compiles without warnings. Make sure you compile your code with the appropriate options to tell the compiler to display warnings and to compile against the required standard.

More specifically, the detailed marking scheme for this particular assignment is given in Table 1. The aspects in Table 1 will apply equally to both the C and C++ programs.

Table 1: Assessed aspects of your work and their value

What	Value
Code layout	5%
Identifier names	5%
Comments	5%
Readability	5%
Documentation	10%
Program quality and design	15%
Program compilation	10%
Program success	30%
Extended feature	15%
Total	100%

- The C program's extended feature is dealing with our sea area as described in section 2.1.2.
- The C++ program's extended feature is the 'pod mission' as described in section 2.2.2.

NOTE: Solutions which do not tackle the sea area (section 2.1.2) or the pod mission (section 2.2.2) could potentially be awarded a mark of up to 85% if they cannot be criticized in any manner and are accompanied by all other materials indicated in Section 3. Solutions must also tackle the sea area and/or the pod mission for marks above 85%.

Overall, this assignment is worth 40% of the total marks available for this module.

4.1 Implementation Requirements

You are expected to make good use of the facilities of the ANSI C and C++ programming languages.

Your C program **must** make use of **arrays** and/or **linked data structures** at appropriate places. Your C program **must** make good use of **functions** to modularise your code in a sensible way.

Your C++ program **must** be Object Oriented and use the C++ approach to design and implementation. C (rather than C++) programs submitted as your solution to Section 2.2 will not score well, even if they compile and implement the functionality perfectly.

4.1.1 Comments and layout

We expect you to sensibly comment your programs, making sure that all comments add real value and do not just, in essence, duplicate code. The programs must have good layout and must use meaningful names for variables, structures and other identifiers.

4.1.2 Choices you are free to make

You need to read in the data files and store the information within suitable data structures, which you have designed, within your programs. Your programs must use multiple functions, located in multiple files as you choose. You must for example, use the widely adopted C++ file conventions for declaring and defining classes. As part of our assessment, we will evaluate the quality of the choices you have made.

4.1.3 Testing and Data Processing Requirements

As well as any testing you conduct to debug and validate your programs, you **MUST** also analyse each of our sets of data files.

We have uploaded six pairs of data files to Blackboard. They cover a representative range of scenarios including small and large numbers of mammals, some have animals in pods, some do not, some include sightings of mammals from outside our sea area.

A Locations and Distances

A.1 Units used in our Data and for Measurements and Calculations

In accordance with common maritime practice, the "units" used in our data are degrees of latitude and longitude for locations, degrees from true north (clockwise) for directions, and nautical miles for distances.

Observer and Cetacean Mammal Positions

Positions of observers and cetacean mammals will be provided or calculated as a latitude and a longitude in degrees and expressed as real numbers.

Latitude: will be given (or must be calculated) as positive values in the Northern hemisphere and negative values if a location is south of the equator. Thus, latitude goes from 0.000 at the equator to 90.000 at the North Pole and to -90.000 at the South Pole.

Longitude: will be given (or must be calculated) as positive values if the location is to the East of the Greenwich meridian and negative values if the location is to the West of the Greenwich meridian. Thus, longitude will go from 0.000 at Greenwich to negative values as you travel due West and -180.000 is the far side of the world. As you head due East from Greenwich, values increase positively and 180.000 is also the far side of the world, and indeed exactly the same place as -180.000.

Example Locations: Here are two example locations.

1. A latitude of 52.500 with a longitude of -5.500 is in the middle of St. Georges's Channel (i.e., between Wales and Ireland)
2. A latitude of 51.767 with a longitude of 1.283 is the approximate location of the Gunfleet light about 10 miles off Frinton-on-Sea on the UK East coast just North of the entrance to the Thames estuary in the southern North Sea.

Bearings (Directions)

The bearing of a mammal sighting will be given in "degrees true". The direction reports which direction the observer is looking when they sight the cetacean mammal. Thus, a bearing of 45.000 means the observer is looking towards the North East, a bearing of 180.000 means due South and 270.000 means due West.

Ranges (Distances)

The range of a sighting (distance from observer) will be given in nautical miles and expressed as real numbers. [Note: A nautical mile corresponds to the distance equivalent to moving north or south by one minute of latitude, i.e. by one $\frac{1}{60}$ of a degree.]

Our Sea Area

We are only "interested" in what cetacean mammals are doing if they are within "our" sea area (i.e. the sea area between Strumble Head and Braich-y-Pwll). In other words if they are in the sea area covered by the Eastern part of the Admiralty Chart "Saint George's Channel". The precise co-ordinates of the limits of our area are given below.

- If sighting reports indicate that a mammal is further South than a latitude of 52.00, or further North than 52.833, you must ignore that report.
- If sighting reports indicate that a mammal is further East than a longitude of -4.000, or further West than -5.500, you must ignore that report.

A.2 Calculating a Cetacean Mammal's Position

You need to calculate the reported location of cetacean mammals based on the sightings that are received.

You are required to write a function that calculates the location of a cetacean mammal. The function must take an observer location, a bearing and a range as parameters and it must produce a calculated location as its return value.

Here's the "mathematics" you need to be able to calculate that position.

In what follows:

OLAT represents an observer's own latitude in degrees

OLATR represents an observer's own latitude converted into radians

OLONG represents an observer's own longitude in degrees

BG represents the bearing from the observer to the cetacean mammal measured from True North in degrees

BGR represents the bearing of the cetacean mammal from True North converted into radians

RANGE represents the range from the observer to the cetacean mammal measured in nautical miles

CMLAT represents a cetacean mammal's calculated latitude in degrees

CMLONG represents a cetacean mammal's calculated longitude in degrees

To convert an angle in degrees to an angle in radians you have to multiply by PI and divide by 180.0 The value for PI can be accessed as a constant called M_PI which is defined in the math.h standard header file.

That is,

(angle in radians) = (angle in degrees) * M_PI / 180.0;

The references to sin and cos in the equations given below are references to the mathematical functions sin and cos.

Pre-compiled versions of these exist in the C standard maths library and their signatures are again defined in the math.h standard header file.

```
CMLAT = OLAT + (RANGE * cos(BGR)) / 60.0;

CMLONG = OLONG + (RANGE * sin(BGR) / cos(OLATR)) / 60.0;
```

The accuracy of above mathematics depends on a few simplifying assumptions such as the earth being round and the locations not being too near the North or South poles.

A.3 Calculating Distances Between Locations

For the C++ program you need to calculate the distances between the locations of cetacean mammals.

We have provided a class that calculates the distance between any two locations, that is detailed in Section 2.2.3 of this worksheet.

B The Data Files Provided

We provide a set of data files to describe the location of the observers and the sightings they make of cetacean mammals.

B.1 Observer Locations

The Observer Location files contain a set of lines of information representing the location of our observers at a single point in time.

Each data file contains a fixed format introduction followed by a set of entries, one per observer. Your program **MUST NOT** assume any fixed specific number of observers but must process the file until end-of-file is encountered.

The first line of the file contains the date and the time using a 24 hour clock. The line contains six integers, separated by spaces, which are day month year hour minute second.

The remainder of each file contains lines of information, each line representing a single observer and their location. You may assume that all the observer ID values are all unique.

The values supplied on each line are separated by spaces and the values and their types are in the following order:

an "ID" for the observer as an alphanumeric string without any embedded gaps;

the latitude of the observer in degrees as described above;

the longitude of the observer in degrees as described above;

Here is the contents of a (small) example file for the 5th November 2014 at 14:53:00.

```
05 11 2014 14 53 00
AB01 52.408 -4.217
XY23 51.750 -4.300
PQ29 52.100 -6.000
NY23 52.000 -5.900
```

B.2 Cetacean Sightings

The Cetacean Sightings files contain a set of lines of information representing the cetaceans that have been sighted by the observers.

Each data file contains a set of entries, one per sighting. Your program **MUST NOT** assume any fixed specific number of sightings but must process the file until end-of-file is encountered.

The file contains lines of information, each line representing a single sighting, the observer's ID, the cetacean type, the bearing to the cetacean mammal and its range.

Note: an observer might sight multiple cetacean mammals and multiple observers will often sight the same cetacean mammal.

The values supplied on each line are separated by spaces and the values and their types are in the following order:

an "ID" for the observer as an alphanumeric string without any embedded gaps;

the type of cetacean mammal that has been seen as a single letter (`P` for Harbour Porpoise, `D` for Bottlenose Dolphin);

the bearing from the observer to the cetacean mammal measured from True North in degrees as described above;

the range from the observer to the cetacean mammal measured in nautical miles as described above;

Here is the contents of a (small) example sightings file.

```
AB01 P 270.0 4.10
AB01 D 275.0 4.50
XY23 D 320.0 6.79
XY23 D 340.0 7.50
PQ29 P 230.0 6.50
NY23 D 359.0 8.30
```

You will notice that some observers have sighted more than one cetacean mammal.

B.3 Alternative Data Files

The data we provide must not be “hard coded” into your program in any way.

It must be possible for us to provide new data files and then use your program to process the data purely by us providing the names of the new files.

Indeed, we may choose to deliberately have some extra data files, of exactly the same format, which we choose to use when we evaluate your program, which quite deliberately are **NOT** available to you during development!

While any files we would use would be of the same format to those described above, we would have different numbers of observers, cetacean mammals, etc.

C Header file for the `Location` object

```
/*
 * Created by Neal Snooke on 24/10/2017.
 */
#ifndef CODE_LOCATION_H
#define CODE_LOCATION_H

#define EARTH_RADIUS 3440.07
//#define DEBUG

/*
 * class to represent a location
 */
class Location {

private:
    double lat;
    double lng;

public:
    /**
     * @param lat
     * @param lng
     */
    Location(double lat, double lng);

    /**
     * change the coordinates of this location
     * @param lat
     * @param lng
     */
    void setLocation(double lat, double lng);

    /**
     * @return latitude of this location
     */
    double getLatitude() {return lat;}

    /**
     * @return longitude of this location
     */
    double getLongitude() { return lng;}

    /**
     * calculate the great circle distance between this location and another
     * The code is based on the calculation as described on the web page
     * http://en.wikipedia.org/wiki/Great-circle_distance
     *
     * @return distance
     */
    double distance(Location);

    /**
     * stream insertion operator overload to allow formatted stream output
     * @return the stream
     */
    friend std::basic_ostream<char>& operator<<(
            std::basic_ostream<char>&, const Location&);
};

#endif //CODE_LOCATION_H
```