Computing "great circle" Distances

between (latitude, longitude) coordinates

Due: Monday, September 19

In a world at a not too distant time, every citizen's location will be known by a set of coordinates such as latitude and longitude. For example, I type this from my office at: [41.9667526, -71.1870159]. The context for this assignment is that you are working for a new startup company which is manufacturing chips to embed in people's skin. Real-time monitoring of someone's coordinates and computing distances between two pairs of (Lat., Long.) sites is part of the app that you are prototyping.

41.9667526, -71.1870159

We will refer to websites for help with the equations for approximating the distances between two points on the earth's surface.

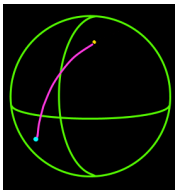
```
http://www.johndcook.com/python_longitude_latitude.html
http://www.johndcook.com/lat_long_details.html

http://www.miguelcasillas.com/?p=107
```

We will assume the earth is a perfect sphere and that the latitude is measured in degrees north of the equator; southern locations have negative latitude. Similarly, longitude is measured in degrees east of the Prime Meridian.

PURPOSE

To provide practice with: (0) translating algebraic and trigonometric formulas into computational equations, (1) calling mathematical library functions, (2) writing your own user-defined functions, and (3) using conditional control with the IF-ELSE statement.

You are to write a program that determines the distance in miles (m) *and* kilometers (km) between two sets of (latitude, longitude) coordinates and then using you computed distance (sometimes called the "great circle distance" to determine if you should (a) walk, (b) drive, or (c) fly to get from one point to the other.



METHOD

The problem is to be handled as follows: First, your program must ask the user at the keyboard to provide two (2) pairs of REAL (latitude, longitude) coordinates. Imagine that these values (in the future) come from (1) an embedded chip in your body and (2) a chip in another person's body. (*Yeah, **we** know where you are.*) Using the formula on the reference website, your program should next compute the distance between the two points and print the distance in both miles and kilometers. Lastly, your program should print an appropriate message indicating if how the people at the two points can travel to meet the other: (a) short distance, under **1** mile: **walk**; (b) **drivable** distance, at least 1 mile but no more than **500** miles; or (c) 500 or more miles they must **fly**. [Note: future work on this app will have to pay much more attention to *where* the two individuals are and if transportation is available, but for now, we will concentrate on the computational calculations … and of course, getting them correct!]

INPUT
All latitude and longitude (digital) coordinates are obtained from the keyboard. All input values are Real numbers. You should prompt the user appropriately. It is considered good practice to provide a good example of a correct input (see the sample output below). Some sample inputs are shown below:

```
"Wheaton College, Norton, MA, USA"            "Jay, ME, USA"
     41.9667526,   -71.1870159                     44.479866,   -70.196829

"near campus, Norton, MA, USA"                "Shanghai, China"
     41.962696,   -71.189858                       31.230393,   121.473704

"Attleboro, MA, USA"
     41.944487,   -71.284098
```

You can use this website to enter a location on earth and get the coordinate pair:
`http://itouchmap.com/latlong.html`

OUTPUT
Your output should include (1) a title; (2) a summary (echo) of the two coordinate pairs that were input; (3) the distance and units (miles and km) between the points; and (4) an appropriate message suggesting how you might travel from one point to another (see directions above). Part of a sample output is shown below (user **input** is in bold):

```
     ----------------------------------
     Enter name of the place: Wheaton College, Norton, MA, USA
     Enter latitude, longitude coordinates:  41.9667526,  -71.1870159
     ----------------------------------
     Enter name of the place: Shanghai, China
     Enter latitude, longitude coordinates:  31.230393,   121.473704
     ----------------------------------
     ------------------------------------
     Wheaton College, Norton, MA, USA:
      [41.966753, -71.187016]
         \
          \
           ------
                 \
                  \
       Shanghai, China:
        [31.230393, 121.473704]
     ------------------------------------
     Distance is:  7317.89 miles
     Distance is: 11776.97 km

     Too far to drive; you should FLY.

     Done.
```

REQUIREMENTS:

• The Starter Kit has given you two functions: `enterPlace()` and `echoTwoCoordinates()`.

• You must write two functions to convert between miles and km (and vice versa). The functions `convertMilesToKM()` and `convertKmToMiles()` should accept one argument (in one unit) and **`return`** the conversion.

• You must write a function `distanceBetweenPoints()` which will accept four arguments (two [latitude, longitude] pairs) and return the distance between these two locations. Obviously, this is where the mathematical work will be. See the reference websites on the first page of this specification to get *lots (ahem)* help here.

• You must call the functions above from `main()` to solve the problem.

PROGRAM DETAILS

Your program must have an initial comment section, giving your name, summary, and Date Last Modified subsections. A second comment section must include a description of the INPUT that the program uses (including where the input comes from, in this case, the keyboard, sometimes called standard input or stdin as well as the data types of all inputs), and a description of the program's OUTPUT. You MUST have these comments *neatly* placed in the comment sections as described in class. Yes, use a spell checker if needed; your boss cannot tolerate sloppy documentation.

• You MUST write a comment for EACH variable. Variables MUST have good names, for example: `# distanceMiles: holds the distance in miles`

• You MUST use "constants" where appropriate (use CAPITAL_LETTERS for constants). All numbers that will *never* change (e.g., the radius of the earth) should have names in CAPITAL_LETTERS. Set up all your constants at the bottom *before* you call `main()`.

• Note: a more "real world" scenario would collect the two coordinate pairs in real time, perhaps after one person pinged the other person. But here, you can just collect the two pairs one after another via the keyboard (input).

• All output is to be formatted to <u>as shown in the example output</u>.

GRADING

You will be graded by the following general rubric:

<u>Average</u>: You finish the program, your program produces the correct Output and you meet all the Requirements and Program Details.

<u>Above Average</u>: You write an additional function called `howShouldWeTravel()`. This function should accept one argument, the distance between the two points in miles. This function should do the `if-else` checking to print the appropriate message in the function (rather than at the end of the `main()` function).

<u>Superior Effort</u>: You alter the **`echoTwoCoordinates()`** function to print a more "realistic" picture of the relative locations of the two sites. You may assume a "flat" map. Your printout of the two locations should try to locate the two places in correct relative position, that is, west-east, north-south. You will clearly have to take advantage of some knowledge of (latitude, longitude) values. Be creative here, but of course we understand that you are limited to console (text-based) output (but so is an ATM screen and many mobile devices).