Team Project Specification

Due: 11:59 P.M. Tuesday, April 26, 2016

The project is to design and write a C++14/FLTK computer game program with a graphical user interface. The game is based on “maximal dispersal on a sphere,” which is a simplified model of where to place outer-shell electrons or various kinds of satellites (*e.g.*, GPS, spy, TV broadcast, telephone, Internet, etc.) or ground stations (satellite receivers, radio telescopes, approaching-asteroid detectors, etc.) around the earth most effectively and efficiently.

Your team should decide on a storyline for your game (for example, the player has to start a new company to broadcast Aggie news around the world, or the player is adding electrons to an atom to give it a more negative charge). Choose a clever name for your project, like “The World-Wide Whoop” or “Eyes for Spies” or “Charge!” (but keep it clean ).

The object of the game is to place a certain number of satellites (or electrons or ground stations) on the surface of a sphere so that the two closest satellites are as far apart as possible. The farther apart the closest two satellites are, the higher the score, as described below.

The program must:

1. Start with an attractive splash screen showing (at least) the name of the game, the team number, and the team members' names. Feel free to add team members' pictures, play music, etc.
2. Explain the storyline and how to play the game.
3. Ask for the player's initials and display the top 5 scores read in from the disk (see item 6 below) followed by the player's initials. The top scores file starts out empty.
4. Ask for a difficulty level from 2 to 8; this is the number of satellites (or whatever). Each satellite is a circle of a different color. Display the satellites in random positions on a planar map of the earth; use either Mercator or polar projection. (See discussion below about “wraparound.”) Note: The Honors project goes up to 18 and displays in 3-D; see the “Honors Project” section below.
5. Allow the player to select a satellite and move it north, south, east, or west from its present position by a given step size, which gradually decreases as the game progresses. The step size starts out at 20 degrees of latitude or longitude, then after 10 moves changes to 15 degrees, after another 10 moves to 10 degrees, then 5 degrees, then 1 degree of latitude or longitude. You may decide to have four buttons labelled N, S, E, and W, or choose some other way to indicate which direction to move the selected satellite. Remember to redraw the screen!
6. After each move, calculate the “minor-arc great-circle distance” **in miles** from every satellite to its nearest neighbor, then take the minimum of those distances and multiply by the number of satellites to get the the player's score, and display it next to the player's initials. (Note: To simplify the calculations, even if you chose satellites or electrons, place the markers on the surface of the earth and assume a radius of 4,000 miles.) After 50 moves the game is over, so sort the list of 6 scores and write the top 5 out to disk with initials. Then the next time the game is played that file will be read in and displayed in step 3 above. Ask the player if they want to play another game or quit.

WRAPAROUND

Since the earth is a sphere, if you keep moving east (increasing the longitude) you will eventually “wrap around” to the west edge of the map (assuming Mercator projection). For example, if you start at +15 degrees longitude and increment by 100 degrees twice, you will move to +115 and then to -145 degrees. A similar thing happens crossing the other edges. Be careful moving to the exact edge of the map, since +180 degrees longitude (the right-hand edge of the map) is the same as -180 (the left-hand edge of the map). Even worse, the North Pole is actually the entire top edge of the map! In the case of polar projections, the equator on both halves is the same, so a satellite on the equator appears on both halves.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

EXTRA ITEMS

Have a “Give me a hint” button that will show the best satellite to move next.

Timer to end the game when time is up. Allow 30 seconds times the number of satellites, *e.g.*, 7 satellites gives 210 seconds. Display a digital clock with the time remaining. Hint: Check the online FLTK documentation for Fl::add\_timeout.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

This is a team project, with three or four students on a team. The instructor will assign the teams. (Note: If there are any problems with your team assignment, please talk to your TA promptly.)

Divide up the code, with each student on a team of 3 doing two of the six items on page 1. If you have a fourth team member, that person must do one of the extra items.

A team of 3 will receive 5 points extra credit for doing one "Extra Item" or 10 points for doing two. A team of 4 will receive 5 points extra credit for doing a second "Extra Item."

You must use at least two C++11 or C++14 features, such as auto and range-based for. (See the reference pages on <http://cplusplus.com/> to see if a feature is C++11 or C++14.) You must write at least two classes of your own, with separate header and implementation files. Follow good style, and limit each function to no more than 24 lines (one terminal window). Each team member is expected to have a rough idea of how all the code works, and should be able to explain in detail how their own part of the code works.

Be creative in deciding how to meet these specifications in an attractive and user-friendly way, but get the basic functionality working before you try to make it too fancy, or you may run out of time!

All user input and output must be through the GUI, not the console window. However, you may use the console window for printing debugging messages for the developers (your team).

Your program should compile and run without change on the Visual C++ 2015 environment in the lab or build.cse.tamu.edu with g++ -std=c++14 and X windows.

Your program must be submitted both to CSNET and also on a CD or DVD. The project report (described below) should be submitted on paper to your TA, along with your CD or DVD. You only need to submit one report and CD or DVD and CSNET file per team. Write a report according to the outline below. All team members will receive the same project grade, unless some team member does not do his/her part (see report outline below).

**Important! You must demonstrate your project to your TA or it will not be graded!**

Note about teamwork: Immediately exchange contact information with your teammates and schedule times to meet and work on the project. If your teammates are in the same lab section, you have the two hours per week during lab, but in any case you need to schedule meeting times outside of lab. As meeting scheduling can be difficult, use this lab time wisely! Attendance will be taken during lab, so that complaints of "We could never find a time to meet" will not be taken seriously.

REPORT OUTLINE

The project report must be printed on a laser printer. The report should include the following sections:

1. Team information (team name, members' names, who did what, did each member

do a fair share of the work)

2. Statement of the problem, significance, etc.

3. Restrictions and limitations

4. Explanation of your approach (analysis to choose a strategy for programming the project, how you coded it, etc.)

5. Sample run (screen shots)

6. Results and analysis

7. Conclusions - What did you show? What did you learn?

8. Future research (how your program could be improved or extended)

9. Instructions on how to run your program

10. Listing of the COMMENTED program

11. Bibliography - references used, if any

HONORS PROJECT

The Honors project includes the basic 6 items above, with the following changes:

1. The maximum number of satellites is 18.
2. Use a 3-D wireframe globe projected onto the plane (the display screen). Find some way to identify which satellites are on the “near side” hemisphere (closest to you) and which satellites are on the “far side” hemisphere, such as making a satellite smaller and/or darker in color when it is on the back side.

Divide up the code, with each student on a team of 3 doing two of the six items on page 1. If you have a fourth team member, that person must do one of the extra items.

A team of 3 will receive 5 points extra credit for doing one "Extra Item" or 10 points for doing two. A team of 4 will receive 5 points extra credit for doing a second "Extra Item."