CS1337 – Fall 2016

Assignment #3 – Class & Object

October 19th, 2016
Due Date: November 6th, 2016 – 11:59pm

Specifications: You are going to write the definitions of **two** classes (and their member functions, of course) and a main function to test them. As you have learned in class, you will have a pair of files (a header and an accompanying implementation file) for each class. Your main function will test and use the classes you've created. That means, in the main function you will declare objects of your two classes and demonstrate that they "work" and are what you think they are. Thus, you would declare an object and then output it to the screen to see that indeed it has the attributes you built into its constructor(s). Then use the member functions and output again to see that something has indeed changed. Test all functions (including constructors).

The classes to create are described in details below:

The Student class:

member variables:

- a string (either type) variable for the student's name.
- student location (in 2-dimensions representation). You can either keep two ints for x- and y-coordinates or make a struct for a point....it's up to you.
- an integer variable for the object's intelligence quotient (iq).

member functions: (all public)

- a default **constructor** to which you can assign student's name to "Bob" with an IQ of a randomly generated value between 50 and 70. The constructor function will also call the **studentPos()** function (to generate a random position).
- Another constructor to which you can pass a string, a position, and an IQ value.
- a **studentPos()** function that will give the object a location value for which both x- and y-coords will be chosen randomly between and include 0 and 24.
- appropriate **setIQ()** and **getIQ()** functions for writing and reading iq value. Be careful how you define them.
- an overloaded **insertion operator** that will output a classmate in the form: <name> has iq <iq_val> and is at <x-loc, y-loc>.
  For example: Tom has iq 45 and is at (4, 7).
- a **studentMove()** function that returns nothing but it will randomly change the location of the object by either decrementing or incrementing its x- or y-coord with an equal chance for any possibility. Be sure that the move function doesn't place the object out of bounds!

The Campus class:

member variables:

- a 2-Dimensional array of char. Make both dimensions MAX = 25 in the definition of the class. This will represent the maximum size of any campus you declare. Of course, some campus may be smaller than that (e.g. 10*10, 15*15, etc). So....
- an integer campus_size to represent the dimension (width) of the actual usable size of the campus. This also corresponds to how much of the 25 x 25 max yard your particular campus will use.
- an integer building_size to represent the dimensions (both) of the school building on campus. This corresponds to how much of the campus space that the building will use. (See info about placement of building below.)

member functions:

- a constructor that has two arguments (inputs) where the input parameters are the dimensions of the campus and the building. The first dimension will be the campus size (It must be less than or equal to MAX). The second is the building size of a school building that sits on campus with one corner at (0, 0). Again, the dimension must be (strictly) less than the size of the campus, otherwise the building would take up the whole campus and students would have no place to play. The constructor should then call a **private** function called **buildCampus().**
- the **buildCampus()** function, called only by the constructor, will do the following:
  1. Place an 'S' in every cell of the school building (cells[0 – buildingwidth-1][0--buildingheight-1] to establish where the building is.
  2. Place a 'D' (for door) in the cell of the school building opposite the (0, 0) cell (The diagonal corner).
  3. Place a space everywhere else.
  4. Place a 'T' (stands for trash) in 10% (rounded down) of the blank spaces at random in the yard (outside of the building).
- an overloaded **insertion()** operator that will output the entire schoolyard and building in a nicely formatted way so you can see what it all looks like.
- **getCampusDimension()** functions for campus size
- **getBuildingDimension()** function for building size

The main Program:

Your main function, is supposed to be designed for testing each class and its member functions. In your main function, create instances of each of the above described classes and test their functionality. Thus, your program will declare objects of the above types using all the constructors you have made, and then will test the member functions of these objects. You should output the objects to show that the functions have indeed worked.