CS 1110/1111 – Encryption Chase – Loops and Shifts

Today's lecture is a bit different. Once you get settled, please get into groups of three or four. At least one person needs to have a laptop. Download the assigned A, B, C, or D path file from the course notes. The file contains the contents of today's lecture, but it's encrypted in three parts. Each part is encrypted individually using a different key.

Your first job is to decrypt the first part of the lecture by using your Caesar code (due this evening) and a loop (as discussed in class on Monday). The Caesar cipher code from today's homework assumes you have twelve letter and the Caesar shift value of 3. For this exercise, you'll want to be able to handle all of the characters in a line of text, no matter how long that is, and use a different shift value than 3. The shift value to use can be determined by solving the puzzle on the board.

Once you decode the first part, it will give you instructions regarding how to decode the next part. Do what is says.

Hints on Coding

Information on the *for* loop can be found in Section 4.5 of your text and in last class's lecture notes; here's the quick version:

```
for (int i = 0; i < maximum; ++i) { // (i and maximum can be any variable you like)
        // add code to be repeated here
}
```

The part of the *for* loop inside the parentheses works somewhat like the similar part of an *if* statement, except there are more parts to deal with:

- *int i = 0* – This sets up the counter. In this case, *i* is going to keep up with which time through the loop you are in. You start in the 0^{th} iteration (kinda like how the first character in a string is in the 0^{th} position…).
- *i < maximum* – This part of the loop sets up the condition for when you should stop looping. Right now, this says to keep looping as long as the value of *i* is less than that of *maximum*… (kinda like how you would want the Caesar cipher to keep going until you hit the end of the string… represented by the length of that string)
- *++i* – This part of the loop tells Java how much to increment *i* each go around. Here, this increments *i* by 1 each time (that's what ++ does). But you could also put in *i = i + 2* (or *i += 2*) if you wanted to only do even numbers.
- Inside the *for* loop, you can use *i* at any time to get the current iteration number (0, 1, 2, 3, 4, 5, etc. all the way to just before maximum).

To complete your first task and decrypt the first part of today's lecture, modify your Caesar cipher code to be a loop, thus allowing it to decode messages of any length. Consider this: where should *i* start in the first part of the loop? What is the maximum you want it to go (hint: type the name of your string variable, hit the period key, and cycle through the options… you'll find one that does exactly what you want)? How much should *i* increment each go around? What do you do with *i* to get a character from the string? What do you do with that character?