CSC 335: Project 0 – Getting to know NACHOS
Part 1 due Friday, April 5, 2024 at 3pm
Part 2 due Monday, April 8, 2024 at 5pm
Part 3 due Friday, April 12, 2024 at start of class

Objectives:

- To successfully install NACHOS and get it to run its default program
- To familiarize yourself with some of the NACHOS classes and be able to *reverse engineer* how it works
- To create a well-tested doubly-linked list (to be used in future projects)

Part 1: Install NACHOS

Time to get acquainted with NACHOS, the toy OS that we'll be building this term. Download it from Nexus, unzip it, and configure it in your Java IDE of choice. Consult the NACHOS documentation in the General tab on Nexus for how to do this:

- If using IntelliJ or VSCode, use the dedicated instructions.
- If using Eclipse, consult section 2.6 of the *Nachos 5.0j Tutorial*.
- If using command-line, then I assume you're using Linux. Consult section 2.2.2 of the *Nachos 5.0j Tutorial*.

No matter what development environment you use, make sure you are comfortable using the debugger for code tracing. You'll thank me later.

Once installed, run the code. If it worked, you should see output showing two threads going through a loop, bouncing back and forth between each other:

```
nachos 5.0j initializing... config interrupt timer user-check grader
*** thread 0 looped 0 times
*** thread 1 looped 0 times
*** thread 0 looped 1 times
*** thread 1 looped 1 times
*** thread 0 looped 2 times
*** thread 1 looped 2 times
*** thread 0 looped 3 times
*** thread 1 looped 3 times
*** thread 0 looped 4 times
*** thread 1 looped 4 times
Machine halting!
```

Congrats! You just ran your first NACHOS program.

Part 2: A doubly-linked list

To demonstrate the issues that an OS must deal with involving shared data structures, you're going to implement a doubly-linked list ADT in Java, which we'll then use in the next couple of projects.

Download the **DLList.java** starter code from Nexus and place it into the *threads* directory in NACHOS. Make sure the file is in the `nachos.threads` package.

Remember that each node in a doubly-linked list has two pointers – one to the next node and one to the previous node in the list. The `next` pointer of the last node in the list points to null, as does the `prev` pointer of the first node.

This ADT is not meant to be a generalized doubly-linked list, but a limited one with a few specific behaviors. Here are the details:

1. Each node will contain two data items: a *key* (an int) and a *data* item (an Object).
2. The list should always be kept in sorted order by *key*.
3. You'll notice something in the code that you may not have seen before in Java: a **private inner class** for the node. This lets DLList objects access the private DLLElement (node) instance variables as if they were public. So in DLList, you can say `first.next` and that would still compile.
4. You are NOT allowed to import any other classes or packages. You are taking care of all the pointer manipulation yourself.

Write, debug, and test all methods in the starter code until they work. You need to create a separate class (also in nachos.threads) to test your code. I recommend JUnit tests, but you're not required to use them. You are required to submit your DLList class to Gradescope for additional testing.

Part 3: Tracing NACHOS

A lot of your time will be spent figuring out what the existing NACHOS code is doing, so we'll start practicing that now by figuring out exactly how the default output shown on the previous page is getting printed.

There is no coding for you to do in this part. Instead, go to Gradescope and answer the questions listed there. This may seem easy, but it's actually the most difficult part of this project. You'll be pouring over a lot of different classes in NACHOS to see what they do. No matter what IDE you're using, you'll want to be able to do a global search across all the different NACHOS directories in order to deduce what's being called from where. The debugger will also be a good tool. And, of course, the Java API.

Turning it in

Part 1: Take a screenshot of your installed NACHOS producing the default output. Upload it to Nexus.
Part 2: Upload a pdf copy of your DLList.java file to Nexus.
Part 3: Submit your answers on Gradescope before the deadline.

Grading

This project is worth 50 points. Here's the breakdown:
Part 1: 6 points
Part 2: 18 points (all from Gradescope)
Part 3: 26 points (again, from Gradescope)

Gentle Reminder

Programming projects are *individual* projects. I encourage you to talk to others about the general nature of the project and ideas about how to pursue it. However, the technical work, the writing, and the inspiration behind these must be substantially your own. You must cite anyone else who contributes in any way to the project by adding appropriate comments to the code. Similarly, if you include information that you have gleaned from other sources, you must cite them as references. Looking at, and/or copying, other people's code is inappropriate, and will be considered an honor code violation.