10 LABS x 2 hours 15 minutes

- Lab 1: OOP Reviews & Arrays
- Lab 2: Simple sorting
- Lab 3: Stacks & Queues
- Lab 4: Linked List
- Lab 5: Recursion
- Lab 6: Trees
- Lab 7: Hash Tables
- Lab 8: Graph
- Lab 9: Exam
- Lab 10: Project Presentation

There are 8 practical labs (30%):

- Select 3 random submissions to mark
- If you miss a lab or a submission: that lab will be selected to mark

Lab 9 will be a practical exam (35%)

- You can use your laptop to code
- You are only allow to use the following IDE:
    - NetBeans
    - VS Code
    - BlueJ
    - IntelliJ
- You must DISCONNECT your laptop from the Internet

Lab 10 is the project presentation (35%)

Deadline to submit your work on Blackboard: 3 days from the lab day

- i.e., Lab day is Monday => deadline is Wednesday (mid-night)

Assignments submission guide

- Create the folder with a name like: ***StudentID_Name_Lab#***, (e.g. ***01245_VCThanh_Lab1***) to contain your assignment with subfolders:
    - Problem_01 (sometimes Problem_i or Problem_Array)
    - Problem_02 (sometimes Problem_ii or Problem_Queue)
    - etc.
- Compress (.zip) and Submit the whole folder with the same name (i.e., ***01245_ VCThanh_Lab1.zip***) to Blackboard
- Students **not** following this rule **will get their marks deducted**

4. Lab 4: Linked List

4.1. Objectives

- Linked list implementation and operations
- Linked list efficiency
- Abstract Data Types (ADT)
- Stack and Queue as linked lists
- Sorted lists
- Doubly linked lists

4.2. LinkList2App.java

- add a method insertAfter to insert after a particular item in the list

4.3. LinkStackApp.java

- write an application to reverse a list using a stack

4.4. LinkQueueApp.java

- Put different classes in separate files.
- Create a new remove() method that removes item N after N calls to the method.
- Simulate a queue of customers each one served for a random amount of time.
- Add a size() method and investigate how simulation is affected by the time needed to serve a customer and the rate at which customers join the queue.

4.5. Problem II: Josephus Problem

The Josephus Problem (https://en.wikipedia.org/wiki/Josephus_problem) is a famous mathematical puzzle that goes back to ancient times. There are many stories to go with the puzzle. One is that Josephus was one of a group of Jews who were about to be captured by the Romans. Rather than be enslaved, they chose to commit suicide. They arranged themselves in a circle and, starting at a certain person, started counting off around the circle. Every nth person had to leave the circle and commit suicide. Josephus decided he didn't want to die, so he arranged the rules so he would be the last person left. If there were (say) 41 people, and he was the 16^{th} person from the start of the circle, what number should he tell them to use for counting off? The problem is made much more complicated because the circle shrinks as the counting continues.

The problem—given the number of people, starting point, direction, and number to be skipped—is to choose the position in the initial circle to avoid execution.

Your task:

- Create an application that uses a circular linked list (like that in Programming Project 5.3) to model this problem.
- The application must ask for user inputs:
    - the number of people in the circle
    - the number used for counting off
    - the number of the person where counting starts (usually 1).

- The application must print out the output: the list of people being eliminated in order (assuming you go around clockwise).

See Figure 1 below for a visualization of an example. There are 41 soldiers numbered 1 through 41. They start counting at 1 and with a step size of three. The last two soldiers remaining are #16 and #31. ***Note:*** *if the counting continues with those two soldiers, then the last one remaining is #31 only.* Figure 2 shows the example input and output that your application should have.

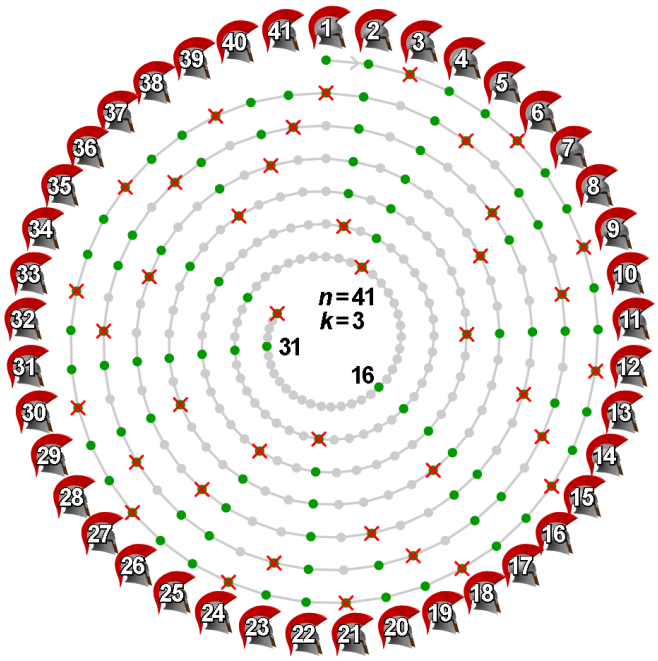

Figure 1: Claude Gaspar Bachet de Méziriac's interpretation of the Josephus problem with 41 soldiers and a step size of 3, showing that places 16 and 31 are last to be killed – time progresses inwards along the spiral, green dots denoting live soldiers, grey dead soldiers, and crosses killings.

```
============================================================

Enter the number of people in the circle: 41
Enter the number used for counting off: 3
Enter the number of the person where counting starts: 1
Elimination order:
3 6 9 12 15 18 21 24 27 30 33 36 39 1 5 10 14 19 23 28 32 37 41 7 13 20
26 34 40 8 17 29 38 11 25 2 22 4 35 16
Last person standing: 31
```

Figure 2: Example input and output