Administrative Issues

EECS1021:
Object Oriented Programming:
from Sensors to Actuators
Winter 2019

CHEN-WEI WANG

Instructor

- How may you call me?

  JACKIE (most preferred)

  "Professor Jackie", "Professor", "Professor Wang", "Sir", "Hey", "Hi", "Hello"
- Office: Lassonde Building 2043
- Office hours: ***3pm – 5pm*** on ***Wednesdays*** and ***Fridays***.

  Or by appointments.
- Advice on performing well in the course? Speak to me *early* !

Class Protocol

- No talking, no mobile – *distracting*, *disrespectful* to everyone
- If you feel like talking or using mobile, please ***leave***.
- In class: core concepts, examples, *your engagement*

Writing E-Mails to Your Instructor

- Think of me as your ***colleague*** who is happy to help you learn.
- You want to write to your colleagues with ***courtesy***.
- This sounds ***very rude*** (and may be delayed, if not ignored):

```
On the link you sent us for our mark
my mark for lab0 did not appear on it
and i submitted lab0 during my lab session
--
sent from my iPhone
```

- This sounds ***much nicer***:

```
Hello Jackie, the link you sent didn't work.
I did submit my lab0. Could you please look into this?
Thanks! Jim
--
sent from my iPhone
```

Course Information

- Course moddle page:
  `https://moodle.info.yorku.ca/`

  Log in and look for Section Z of EECS1021

  There is a course forum. Post your questions!!

  Never share solutions to labs on the forum!!!
- For personal, course-related inquiries:
  eecs1021.w19@gmail.com
- Check your emails regularly!

Not enrolled yet? Switch lab session?

- Please speak to the undergraduate office at ***LAS 1012M***.
- If you haven't enrolled yet, it's your ***sole responsibility*** of following the course as if you were enrolled.

  $\Rightarrow$ It's *not acceptable* to ask for extensions on missed labs/quizzes/tests after you are enrolled.
- If you haven't been enrolled but want to be added to the course moodle, contact: eecs1021.w19@gmail.com with your:
  - Student Number
  - LastName, FirstName
  - Passport York Login ID

Required Study Materials

- Lecture materials (recordings, iPad notes, slides, example codes) will be posted on my website for you to *re-iterate concepts and examples*:

  `https://www.eecs.yorku.ca/~jackie/teaching/lectures/#EECS1021_W19`

- Given that we only have 2 hours of lecture per week, when there are *slides that I cannot finish in class*, you will be required to complete them in this tutorial series:

  `https://www.youtube.com/playlist?list=PL5dxAmCmjv_4ULUz5UodwJjtaJ8YJ__bz`

  I will restrict this to < 1 hour (a week) ☺

- I will crate tutorial videos to *help you complete lab exercises*:

  `https://www.youtube.com/playlist?list=PL5dxAmCmjv_5NRNPG3OiWZWAqmvCjiLfG`

Grading Scheme

2-Part Lab 0 Submissions (1.5% each)	3%
6 Lab Submissions (3% each)	18%
6 in-lab Quizzes (2% each)	12%
In-Lab Programming Test #1	5%
In-Lab Programming Test #2	10%
In-Lab Programming Test #3	10%
Final Exam	42%

Each lab submission (except for Lab 0) consists of two parts:

- 2% for Eclipse programming (e.g., console application)
- 1% for hardware configuration (e.g., Phidget board)

Lab (Grading, Quiz, and Test) Schedule

Agenda for Scheduled Lab Session

- Each scheduled lab session is for 3 hours.
- The agenda to be run depends on whether or not there is a scheduled lab test.
- You ***must*** attend the quiz/test/grading session of your ***registered lab session***.

Agenda for Scheduled Lab Session (1)

Case 1: When There Is a Scheduled Lab Test

- You will be seated according to a *seating plan*.
- The test starts promptly *at the start* of the scheduled lab session.
- Typically a lab test takes ***80 minutes***.
  - *Preparation guide* will be given to you about a week prior to the test.
  - *No data sheet* will be allowed.
  - You **are** required to use Eclipse and write Java programs with *valid syntax* .

Agenda for Scheduled Lab Session (2.1)

LASSONDE SCHOOL OF ENGINEERING

Case 2: When There Is not a Scheduled Lab Test

- There will be ***a Quiz*** (based on lectures and labs).
- Arrive promptly *at the start* of your scheduled lab session.

10 minutes : Check your YU card and sign up for the quiz.

20 minutes : Take the quiz (no data sheet, no Eclipse).

5 minutes : Take a break.

75 minutes : Complete and get programming exercises graded.

- Programming exercises will be **assigned to you in advance**.
- You must **work on your own** for this programming part of the lab.

10 minutes : Take a break and check out the hardware kit.

50 minutes : Complete hardware experiments.

- Instructions will be **given to you in advance**.
- You must **work as a team of 4** for this hardware part of the lab.

Agenda for Scheduled Lab Session (2.2)

Example: LAB 06, starting at 5:30pm on Fridays, in WSC108.

Case 1: When There Is a Scheduled Lab Test

- The test runs from 5:30pm to 6:50pm.

Case 2: When There Is not a Scheduled Lab Test

- You are required to arrive promptly at 5:30pm.
  - **5:30 – 5:40** : Check your YU card and sign up for the quiz.
  - **5:40 – 6:00** : Take the quiz.
  - **6:00 – 6:05** : Take a break.
  - **6:05 – 7:20** : Complete and get programming exercises graded.
  - **7:20 – 7:30** : Take a break and check out the hardware kit.
  - **7:30 – 8:20** : Complete hardware experiments.

Why this Course? (1)

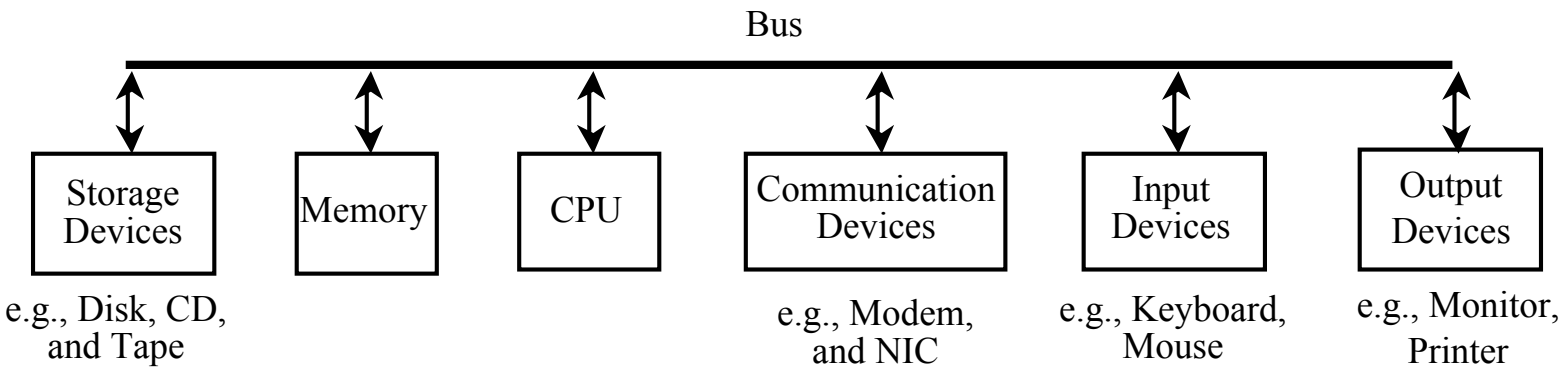
- It is a *pre-requisite* to:
  - ***EECS2030***: Advanced Object Oriented Programming
  - ***EECS2011***: Fundamentals of Data Structure

[the “job interview course”]

Why this Course? (2)

- *Computational thinking (CT)* is a fundamental skill for **everyone**, not just for computer scientists.
  - Reference: Wing, J.M., 2006. *Computational thinking*. Communications of the ACM, 49(3), pp.33 – 35.
  - Thinking like a computer scientist means **more than being able to program** a computer. It requires **thinking at multiple levels of abstraction**.
    - ***Level of Java Code***: How Programs Behave at Runtime
    - ***Above the Level of Code***: *Logical rationale* behind some *functioning*/*malfunctioning* code.
- Being able to think *abstractly* without seeing changes on a physical device is an important skill you are expected to acquire when graduating.
  - Think of programming interviews at Google: Given problems described in English, solve it on a whiteboard.

What Is Course About? (1)



A computer includes both:

- *Hardware*
  - visible, physical, tangible (peripheral) devices
  - *repeatedly* and efficiently executes given instructions
- *Software*
  - invisible, abstract, intangible task-control instructions
  - reflects programmers' *intelligence*

Does the notion of ***stupid computer*** really make sense?

What Is Course About? (2)

- What computers read is difficult for humans, and vice versa.
  - Computers are good at processing *machine language* (0s and 1s).
  - Human beings are good at *abstract thinking* for problem solving.
- *Assembly language* is a big step forward for humans to specify steps of primitive instructions (e.g., memory loads/stores, arithmetic operations, etc.).

  Say `$t0`, `$t1`, `$t2`, `$n`, `$i` are addresses; `$n` stores value *N*:

```
lw      $t0, $n           # fetch N, store in $t0
mult    $t0, $t0, $t0     # store N*N in $t0
lw      $t1, $n           # fetch N, store in $t1
mult    $t1, $t1, 3       # store 3*N in $t1
add     $t2, $t0, $t1     # store N*N + 3*N in $t2
sw      $t2, $i           # store N*N + 3*N in $i
```

  - *Level of abstraction* of the assembly is still *too low* for humans.
  - The above is equivalent to a line of Java code: `i = N*N + 3*N`
  - You will have fun with programming in assembly in EECS2021!

What Is Course About? (3)

- *High-level programming language* (e.g., Java) is even closer to our natural way of thinking (i.e., closer to “writing an essay”).

```
1  Scanner keyboard = new Scanner(System.in);
2  int weight = keyboard.nextInt();
3  int height = keyboard.nextInt();
4  int bmi = weight / (height * height);
5  System.out.println("BMI (Body Mass Index) is: " + bmi);
```

- You will study fundamentals for *Computational Thinking*:
  - assignments
  - conditionals
  - loops
  - 1D and 2D arrays
  - classes and objects
  - attributes and methods

Is This an Easy Course?

This may ***not*** be an easy course.

- You need to work HARD and STEADILY in order to perform well.
- Hardware experiment (e.g., Phidget board) is only meant to be a way to have you engaged.
- Acquiring the *programming* and *problem-solving* skills is the key to success in this course.

But this will ***be*** a course for you to acquire solid computational thinking and programming skills.

Study Tips

Each lecture will be recorded entirely:

- *Not meant to be a replacement for classes!*
- Focus reaching *maximum comprehension*.
- *Ask questions!*
- Take (even incomplete) notes, which will help when re-iterating lectures.

General Tips about Studying in a University

- To do well, ***inspiration*** is more important than ***perspiration***.
- Hard work does not necessarily guarantee success, but no success is possible without ***hard work***

  $\Rightarrow$
  - Don't be too satisfied just by the fact that you work hard.
  - Make sure you work hard both on ***mastering "ground stuffs"*** and, more importantly, on ***staying on top of what's being taught***.
  - Be *adventurous* about going beyond lectures (e.g., CodingBat).
  - Be *curious* about why things work the way they do.
  - Always *reflect* yourself on *how things are connected*.

Academic Integrity

The moral code or ethical policy of academia:

- avoidance of cheating or plagiarism;
- maintenance of academic standards;
- honesty and rigor in research and academic publishing.

Pay careful attention to *all* occasions where the submitted work is to be graded and receive credits (i.e., labs, quizzes, assignments, tests, exams).

It is *absolutely not* acceptable if, in any of these occasions, you:

- share your (programming or written) solutions with others;
- copy and paste solutions from elsewhere and claim that they are yours.

Need Accommodation for Tests/Exams?

- Please approach me (email, in person) as soon as possible, so we can make proper arrangements for you.
- We will work out a way for you to gain the most out of this course!

Index (1)

Index (2)