EECS 1710
Programming for Digital Media

Dr. Matthew Kyan
Dept. of Electrical Engineering & Computer Science
Fall 2018

YORK UNIVERSITÉ UNIVERSITY

EECS 1710
Programming for Digital Media

Lecture 1 :: Course Introduction

YORK UNIVERSITÉ UNIVERSITY

Welcome

- Motivations
- Administrative
- Syllabus
- Assessment
- Resources
- Course Themes
- Software & Tools

Contact

- Instructor

Dr. Matthew Kyan
Associate Professor
Dept. of Electrical Engineering & Computer Science
Office: LAS 3030 (office hours: Thursday 2-4pm)
Tel: 416-736-2100 x33965
Email: mkyan (at) cse (dot) yorku (dot) ca

- TA's

Mehrnaz Zhian
Email: mehrnaz (at) eecs (dot) yorku (dot) ca
Brian Wijeratne
Email: bwijerat (at) eecs (dot) yorku (dot) ca
Sean Delong
Email: seand (at) eecs (dot) yorku (dot) ca

YORK UNIVERSITÉ UNIVERSITY

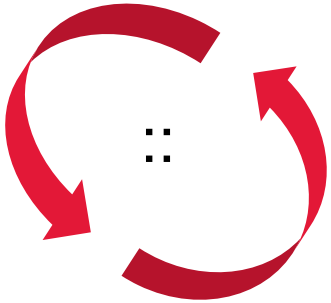
Digital Media @ YorkU

- Provides a foundation in the **computational basis** for the creation of digital media
  - imagery and sound, animation, simulation, 2D/3D environments
  - physical & tangible computing
  - human computer interaction
- Explores theoretical, artistic and experiential ideas
  - that lie behind an informed understanding of aesthetic & functional aspects of digital media
- Engages in the practice of creating digital media works
  - that explore the ways in which culture is produced and can be produced through technology
  - broader socio-cultural effects and the theory and research concerning responses to and use of digital media.

YORK UNIVERSITÉ UNIVERSITY

Digital Media @ YorkU

Research Creation :: Science/Engineering R&D

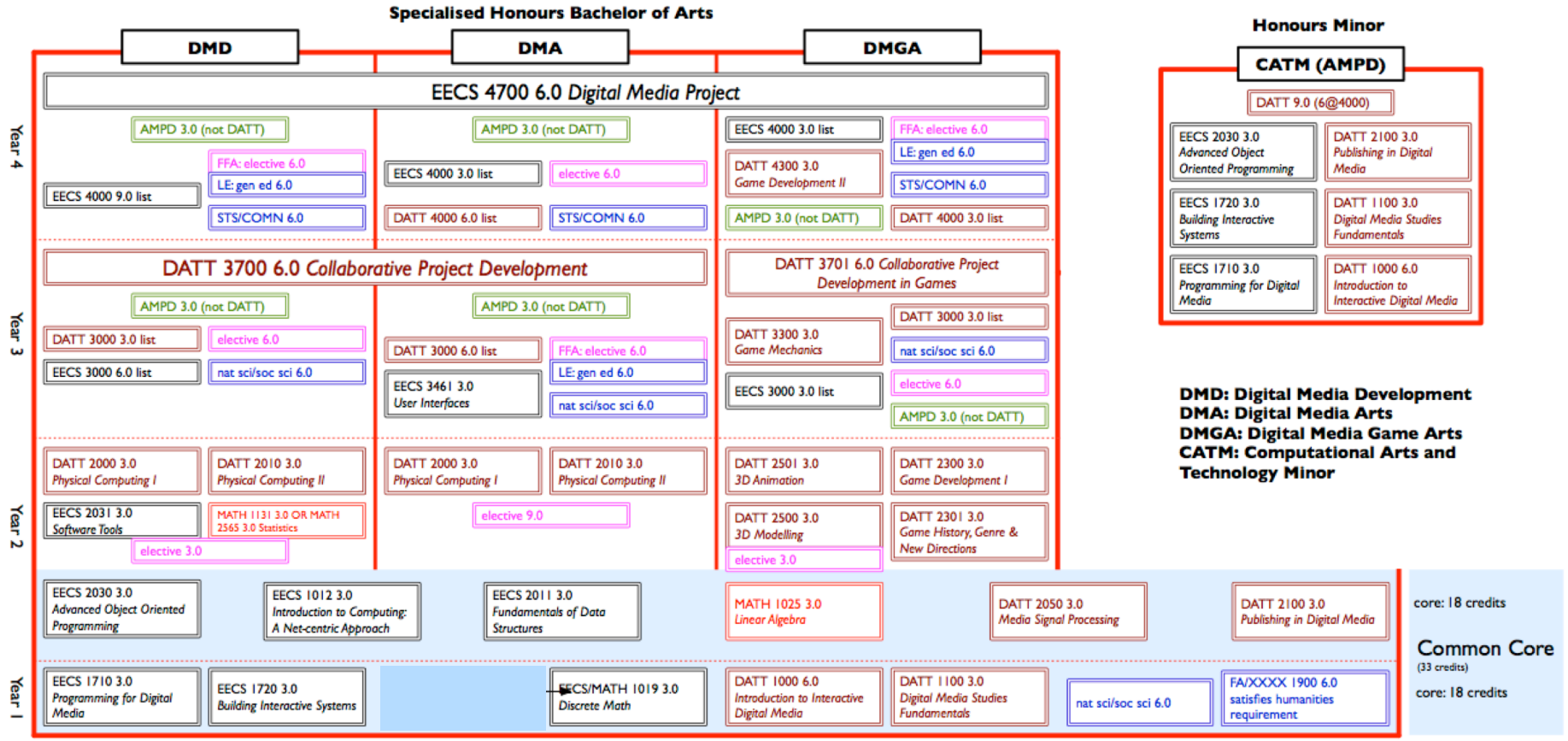


‘creative coding’

goal: create something
‘expressive’ rather than
‘functional’

YORK UNIVERSITÉ UNIVERSITY

Pathways



YORK UNIVERSITÉ UNIVERSITY

What is Computer Science about?

- Study of *process*:
  - **how** we do things, how we specify **what** we do, and how we specify the **stuff** we are processing
  - The study of “recipes” (programs/algorithms)
  - Recipes can simulate real world environments/events, or
  - Recipes can interface directly with real world environments/events

What are Computer Scientists concerned with?

- How recipes are **written** (efficiently expressed and efficient execution/use of resources)
- How a recipe **works** (core steps) as opposed to how it is written (algorithms)

- How units of information are **represented** and used in a recipe (data structures)
- How to **store** and work with very large amounts of information (databases)

- How to design recipes that **think/act intelligently**? (artificial intelligence/machine learning)
- Explore how people **engage** with recipes? (human computer interface/interaction - HCI)

- Use of recipes to **compose** (e.g. graphics/music)
- Use of recipes to **analyze**

- What happens when lots of recipes **communicate**/interact with one another? (networking)

YORK UNIVERSITÉ UNIVERSITY

- Recipes have aspects which are **flexible/variable**, …

  (e.g. amount of ingredients)

- … and aspects which are quite **fixed/restrictive**

  (e.g. type of some ingredients, order of steps, etc)

ultimately they serve as a template for a *process* (in which certain ingredients are transformed into a result)

What do computers understand?

- What do computers understand?
  - Numbers – in fact, even less.. just high/low (on/off) voltages

- What is the concept of an “encoding”?
  - Uses high/low to “encode” things
    - how many things can be encoded with a single “wire”?
    - Multiple “wires”, encode more things (numbers, symbols, etc)

Imagine a “wire”
in 1 of 2 states:


has a voltage (ON)


no voltage (OFF)

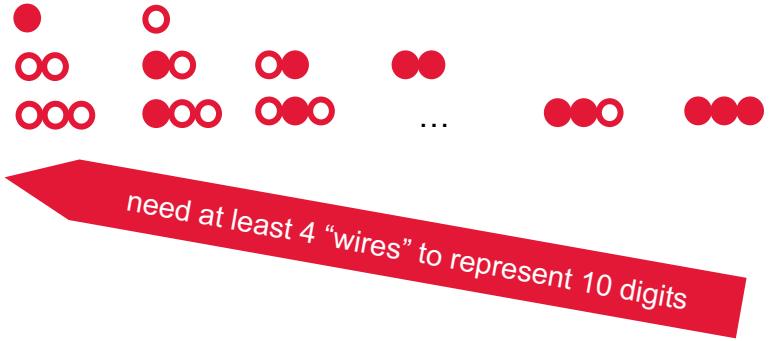
Each state can represent a (symbolize) a different thing:
Therefore 2 things can be represented (e.g. 2 digits)?

How many wires needed to represent 10 digits (0,1,2,…,9)?

YORK UNIVERSITÉ UNIVERSITY

An encoding is a way of storing information

- We can store information in such encodings!
  - 10 digits requires 10 combinations of on/off
    - 1 wire = 2 combinations
    - 2 wires = 4 combinations
    - 3 wires = 8 combinations
    - 4 wires = 16 combinations



  - on/off voltages (*bits*) are the most basic unit of information understood by a computer
  - numbers can be used to compute & store new numbers:
    - 2 + 4
    - 13 * 5 + (8 – 2)/3

YORK UNIVERSITÉ UNIVERSITY

We can do more than just compute numbers

- Numbers can also be tested against other numbers (conditional logic)
    - e.g. are numbers the same/different, is one smaller/larger than another, etc)
    - Logic tests form a basis for making **decisions**, which can be used to execute different options, resulting in different outcomes

- Encodings can be stored internally for the purpose of a calculation, or can be used to drive (actuate) physical equipment (e.g. screens, motors, lights, etc.)
    - Manipulating encodings is a very exact mechanism..
    - But can be very, very fast

Media Computation :: Why digitize media?

- Computers are good at encoding *discrete* parcels of information
  - Images
    - made up of tiny dots (pixels), sometimes millions of pixels per image
    - each pixel holds a set of red, green and blue (r,g,b) components
    - each component can be encoded by a set of bits!
  - Audio
    - Made up of tiny snapshots (samples) in time, usually 44,100+ samples per second!!
    - Each sample holds a set of bits to represent the amplitude of the sound wave

- Human perception blurs these discrete parcels of information together
  - to form a “continuous” experience of images and sounds!
  - think about a flick book animation
    - a series of pictures each slightly different from the last, in a small notepad
    - when you flick the pages, you see a moving image

Media Computation :: why study computation?

- Computers allow us to easily:
  - digitize/store, but most importantly, deconstruct, manipulate, & reconstruct media
- Digitizing also makes it easier to communicate:
  - Securely send media over large distances (e.g. streaming parcels over the air or internet)
- However, in our discipline (digital media):
  - it allows us to be **more** than just content authors
  - allows us to get “under-the-hood” and understand at a fundamental level, how to construct/deconstruct and refashion media at the source!
- BONUS:
  - studying computers is **empowering** almost a necessary literacy in today’s world (useful for any industry)

YORK UNIVERSITÉ UNIVERSITY

Central themes explored (EECS intro courses)

- EECS 1710 & 1720
  - Programming from ***Client*s** point of view
  - 1710 – basic constructs and concepts
  - 1720 – assembly of more complex interactive programs
- EECS 1012
  - Introduction to web based (browser) programming
  - More emphasis on networking
- EECS 2030
  - Programming from ***Implementers*** point of view
  - (more challenging)

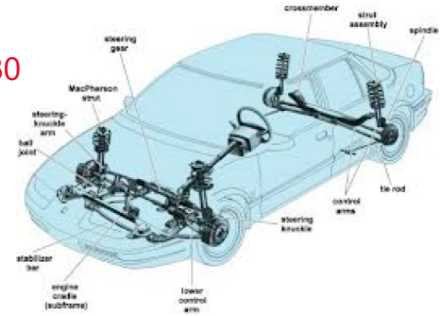
Client vs Implementer (analogy)

- CAR ≈ Program

EECS17xx view

program (recipe)

EECS2030 view



CLIENT

This view creates programs by assembling or 'using' different (implemented) pieces, working with each via their available interface.

IMPLEMENTOR

This view creates program internals from scratch (or assembly), then defines an interface that can serve the needs of possible clients (hiding inner workings from client).

YORK UNIVERSITÉ UNIVERSITY

Course Description (extended)

- Introduction to program design and implementation focusing on digital media elements, and focusing on *CLIENT* perspective
  - sound, images, and animation; algorithms, simple data structures, control structures, and debugging techniques
- Lays the conceptual foundation for the development and implementation of Digital Media artefacts

Course Details

Course Title:	EECS 1710: Programming for Digital Media
Term:	Fall 2018
Lectures:	Tuesday, Thursday: 10:00am – 11:20am Location: LAS C
Laboratories:	Thursday: 11:30pm – 1:30pm Location(s): LAS 1006/1004 (supervised labs begin Week 2: Sept. 13-19)
Term Dates:	Sep 5, 2018 – Dec 4, 2018 Study Day (Dec 5, 2018) Exam Period (Dec 6 – Dec 21, 2018)
Last Day to Add:	Sep 18, 2018 (without permission); Oct 2, 2018 (with)
Last Day to Drop:	Nov 9, 2018 (no grade); Nov 10-Dec 4 ('W' on transcript)
Course Website:	Hosted on Moodle (https://moodle.yorku.ca/)

YORK UNIVERSITÉ UNIVERSITY

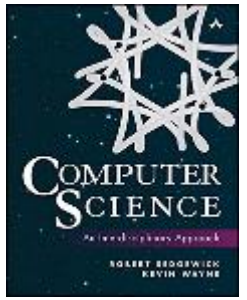
Course Text

- **NO OFFICIAL TEXTBOOK !!**

- Useful Reference Texts
  - M. Guzdial and B. Eriscon, "*Introduction to Computing & Programming with JAVA - A Multimedia Approach*", ISBN 0-13-149698-0
  - J. Lewis and W. Loftus, "*Java Software Solutions*", 9th Ed., ISBN-13: 9780134462028; Pearson, 2018.
  - R. Sedgewick and K. Wayne, "*Computer Science – An Interdisciplinary Approach*"; ISBN-13: 978-0-13-407642-3; Addison-Wesley, 2017.

- Useful Reference Texts & tutorial links (will be posted on website)
- Selected Readings
- Website (Moodle)

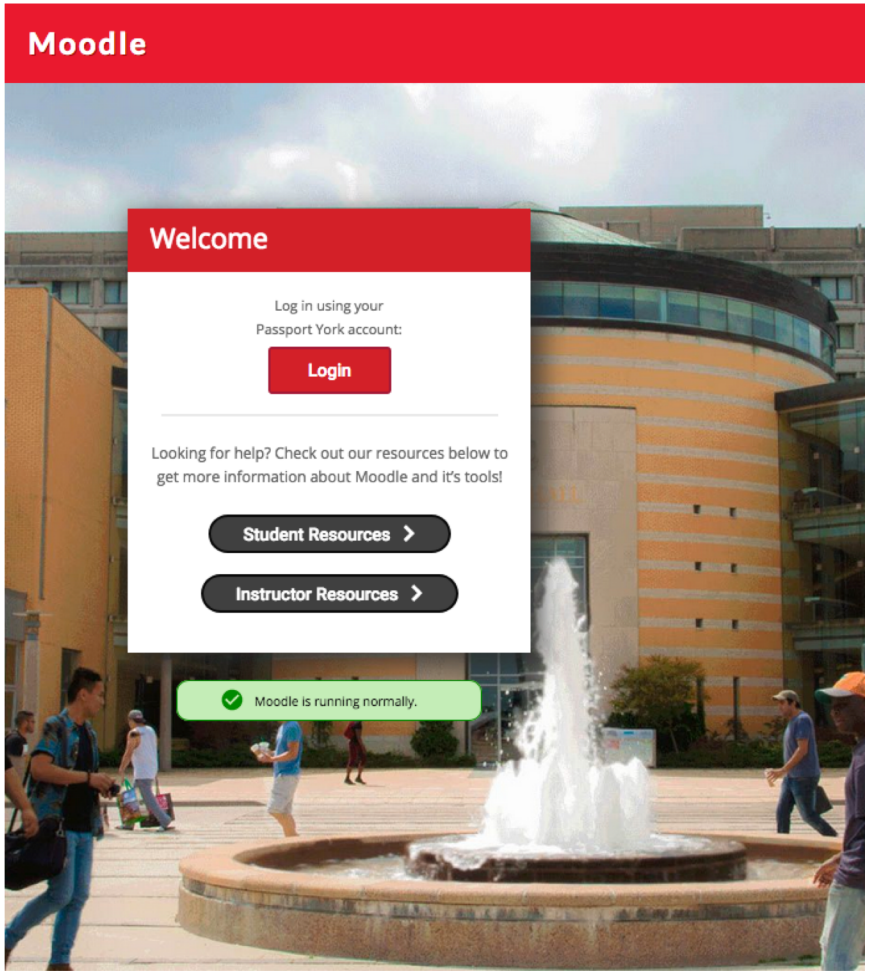
Can rent on amazon for the semester ($23)



A little more current

YORK UNIVERSITÉ UNIVERSITY

Moodle(s) ?



Course is here!

http://moodle.info.yorku.ca/

Labs

- Practice, make, practice, make, learn … (DATT 1000)
- Practice, break, practice, break, learn … (EECS 1710)

- Lab 0 – not supervised (week 1, on your own)
- Lab 1 – week 2

…

9 labs in total (including Lab 0)
Labs 1-8 will be formally submitted (15%)
Lab 1 (1%); Lab 2-8 (2% each)

Course Schedule

Week	Topics	Dates	Activity
1	Course Introduction	Sep 6 – Sep 12	Lab 0 (on own)
	JAVA basics		
2	JAVA basics	Sep 13 – Sep 19	Lab 1 exercises
3	Methods & Objects	Sep 20 – Sep 26	Lab 2 exercises
4	Methods & Objects	Sep 27 – Oct 3	Lab 3 exercises
	Arrays & Loops	Oct 4	Lab 4 exercises
5	FALL READING DAYS	Oct 8 – Oct 11	
6	Arrays & Loops	Oct 18 – Oct 24	MIDTERM (20%)
			Lab 5 exercises
7	Decision Making	Oct 25 – Oct 31	Lab Test #1 (15%)
8	Decision Making	Nov 1 – Nov 7	Lab 6 exercises
9	Working with Graphics	Nov 8 – Nov 14	Lab 7 exercises
10	Working with Graphics	Nov 15 – Nov 21	Lab 8 exercises
11	Working with Sound	Nov 22 – Nov 28	Lab Test #2 (15%)
12	Working with Sound	Nov 29 – Dec 3	
	Course Review	Dec 3	
	FALL STUDY DAY	Dec 5	
	Exam Period	Dec 6 – Dec 21	FINAL (TBD)

YORK UNIVERSITÉ UNIVERSITY

Course Evaluation

Item	Weight (% of final grade)	Due Date
Labs	15	(1 week after scheduled lab session)
Lab Test 1	15	Oct 25 – Oct 31 (in scheduled lab session)
Lab Test 2	15	Nov 22 – Nov 28 (in scheduled lab session)
Midterm Exam	20	Oct 16 (in class)
Final Exam	35	TBD
TOTAL	**100**	

Yet to be scheduled (will be announced on course website)

YORK UNIVERSITÉ UNIVERSITY

Academic Integrity

What is Plagiarism?

Plagiarism is representing someone else's ideas, writing or other intellectual property as your own, and is another form of academic dishonesty.

Any use of the work of others, whether published, unpublished or posted electronically (e.g., on web sites), attributed or anonymous, must include proper acknowledgement.

You can find full definitions of plagiarism and other forms of conduct that are regarded as serious academic offences in York's Senate Policy on Academic Honesty.

Common Types of Plagiarism

In doubt? ASK!!

Plagiarism can take many forms. Some of the most common types of plagiarism include[1]:

- Downloading or buying research papers (Downloading a free paper from a web site or paying to download a paper and submitting it as your own work)
- Copying and Pasting (copying and pasting portions of text from online journal articles or websites without proper citation)
- Copying or submitting someone else's work (copying a paper/lab report/formula/design/computer code/music/choreography/assignment etc. and submitting it as your own work)

[1] Harris, R. A. (2002). *The plagiarism handbook: Strategies for preventing, detecting, and dealing with plagiarism.* Los Angeles: Pyrczak Publishing, p. 13.

Important: You must use York's standards when submitting your work even if you were taught to document your sources differently in the past.

TUTORIAL http://www.yorku.ca/tutorial/academic_integrity/plagdef.html
POLICIES http://secretariat-policies.info.yorku.ca/policies/academic-honesty-senate-policy-on/

YORK UNIVERSITÉ UNIVERSITY

Software/Tools

- Java (programming language)
- Eclipse (Integrated development environment - IDE)
- Linux Environment (operating system - OS)

- (at home) > virtual box + image of linux (CentOS)

tutorial posted on course website

YORK UNIVERSITÉ UNIVERSITY

Resources

- Tutorials (will be linked/posted on course website)
- Java API
  http://docs.oracle.com/jav ase/8/docs/api/
- Java Development Kit (JDK) (ver 7)
  http://www.oracle.com/technetwork/java/javase/downloads/index.html
- Eclipse
  https://eclipse.org/downloads/
- Oracle's Java Tutorials
  http://docs.oracle.com/javase/tutorial/java/
- The type API – http://www.eecs.yorku.ca/teaching/docs/type-api/
- The imagePackage API
  – http://www.eecs.yorku.ca/course_archive/2015-16/F/1710/imagePackageAPI/

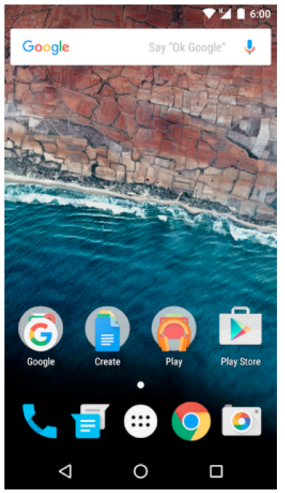
YORK UNIVERSITÉ UNIVERSITY

General Course Advice

- Make USE of available resources:
  - Academic advising services (every program of study has them)
    - what courses should I take? should I drop some courses?
    - should I change my program of study? I am unhappy with my program but what other options do I have?
  - Study supports (from your college, from your faculty)
    - http://bethune.yorku.ca/ , http://winters.yorku.ca
    - study groups, extra help
  - Learning Skills Services
    - http://lss.info.yorku.ca/
  - Study skills workshops, on-line resources, etc.

YORK UNIVERSITÉ UNIVERSITY

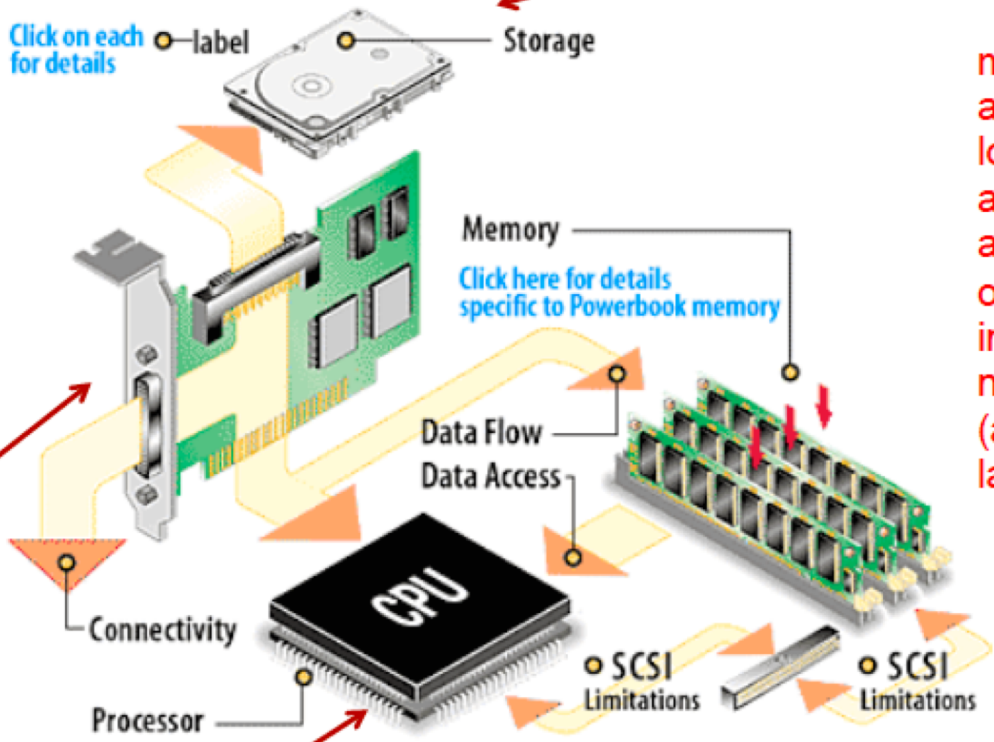
Why Java?



- Basis for Processing
- Basis for Android Programming
- Can deploy standalone or to browsers
- General Purpose
- Portable

- Serves as a good introduction to many programming concepts and constructs

Basic Computer Architecture



the hard drive, where the file system can be found

memory (RAM) # applications are loaded into memory# and get run.# apps are simply a set of instructions written in the processor's native language! (aka machine language)

I/O devices are connected here# (the keyboard, the screen)

the processor (e.g., Intel, Motorola) has its own native instruction set (Not interchangeable!!)

Why Java?

- Intention: “WORA – Write Once, Run Anywhere”
  - Most programming languages come with a compiler for a particular type of CPU/architecture
  - Java code is compiled into a common intermediary form (bytecode) that can be interpreted at run-time on many different architectures via the java virtual machine (JVM)

Your EECS System Account

- You require an EECS account to complete this course.
  - marked course material
  - course announcements
- Activate your account:
  - http://www.eecs.yorku.ca/activ8
  - This account can be used to access any of the PRISM Labs, such as LAS1002, LAS1004, LAS1006, and others
  - Linux and Windows (machines can boot into either OS)
- disk quota, web space, print quota

YORK UNIVERSITÉ UNIVERSITY

Checklist – for next lecture

- get an EECS account (if you don't have it already)
  - warning!! it can take up to 24 hours to get the account. Plan ahead.
- Start working on Lab 0
  - *[optional]* – follow tutorial posted on course website (in Useful Resources Section) for setting up your home computer/laptop environment using an EECS virtual machine (VM); then use VM to do lab 0
  - Otherwise do lab 0 directly in the lab