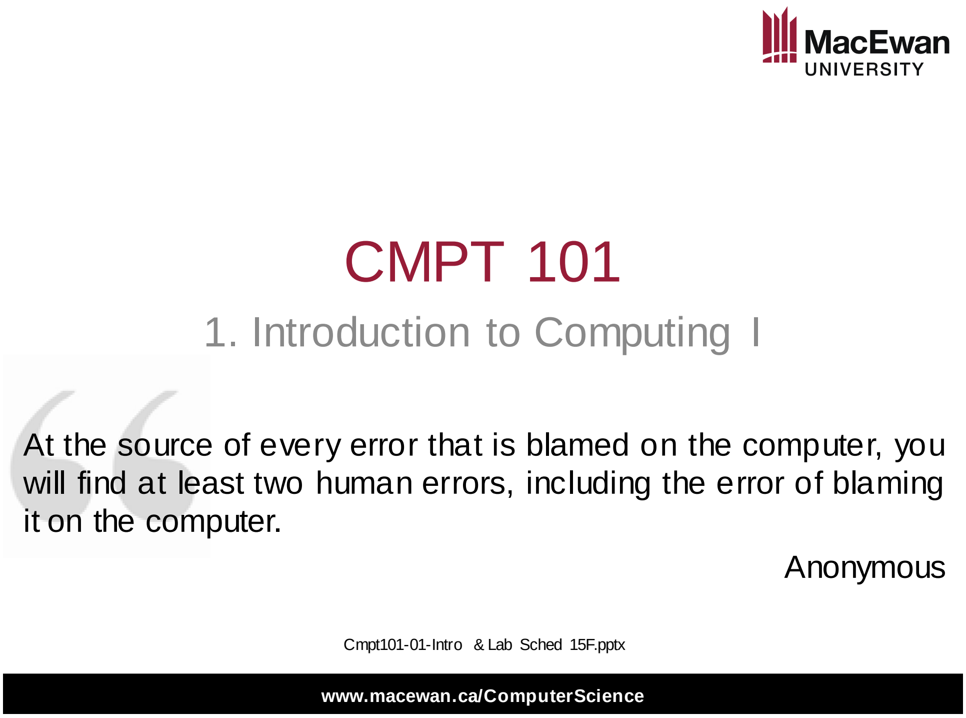


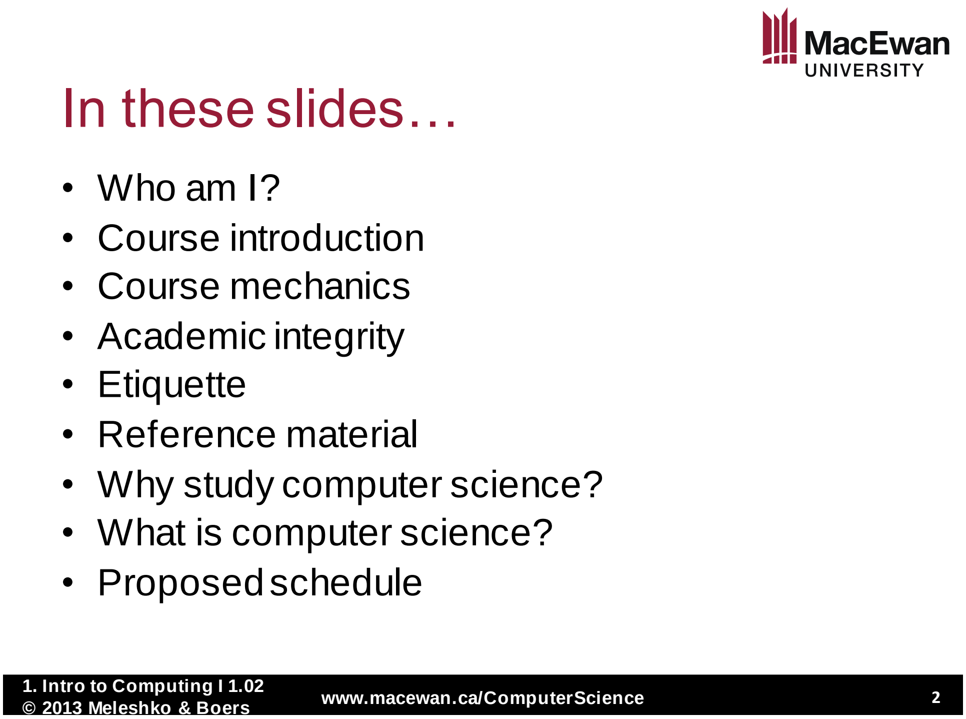
CMPT 101

1. Introduction to Computing I

> At the source of every error that is blamed on the computer, you will find at least two human errors, including the error of blaming it on the computer.
>
> Anonymous

Cmpt101-01-Intro & Lab Sched 15F.pptx

www.macewan.ca/ComputerScience



In these slides…

- Who am I?
- Course introduction
- Course mechanics
- Academic integrity
- Etiquette
- Reference material
- Why study computer science?
- What is computer science?
- Proposed schedule

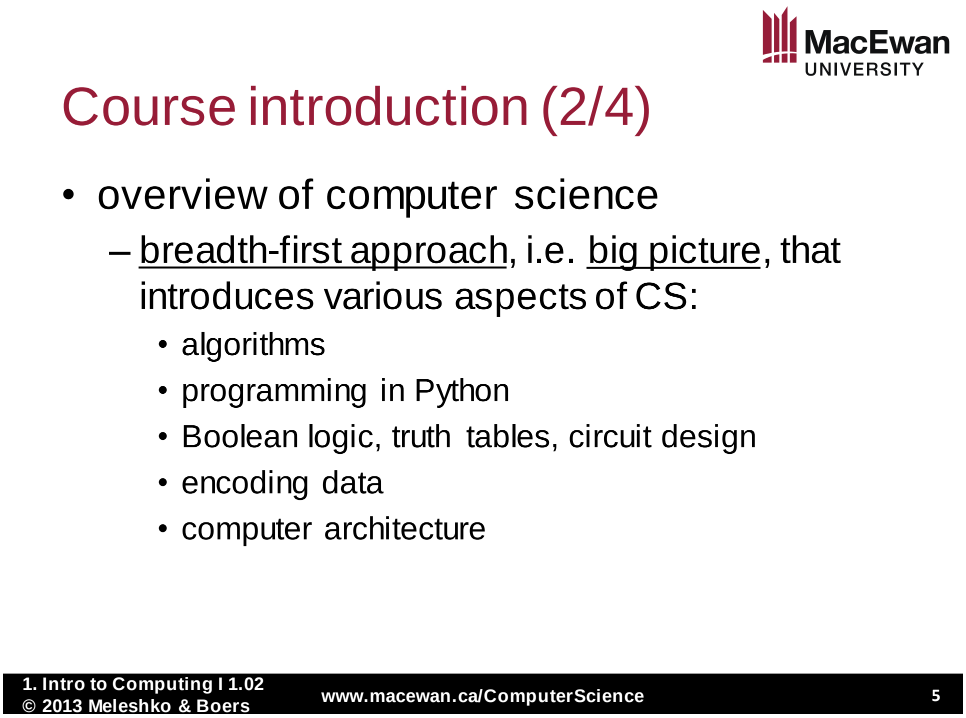
1. Intro to Computing I 1.02
© 2013 Meleshko & Boers
www.macewan.ca/ComputerScience
2

Who am I?

- Dr. / Mr. Krieger
  - kriegera3@macewan.ca
    - **please use your <u>mymacewan.ca</u> account**
  - 780-497-4751
  - 5-173C

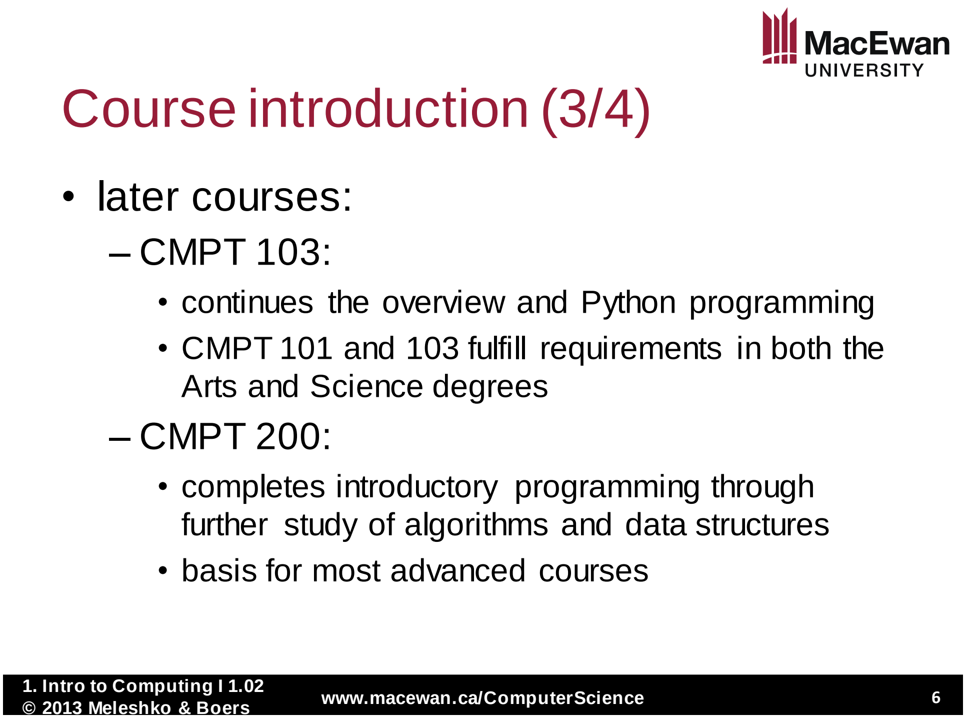
Course introduction (1/4)

- a computing science course
  (<u>not</u> a computer <u>literacy</u> course)
- literacy alternatives @ MacEwan:
  - EDIT 202
  - Business → Continuing Education
- literacy alternatives @ NAIT, Norquest:
  - various computer technology streams



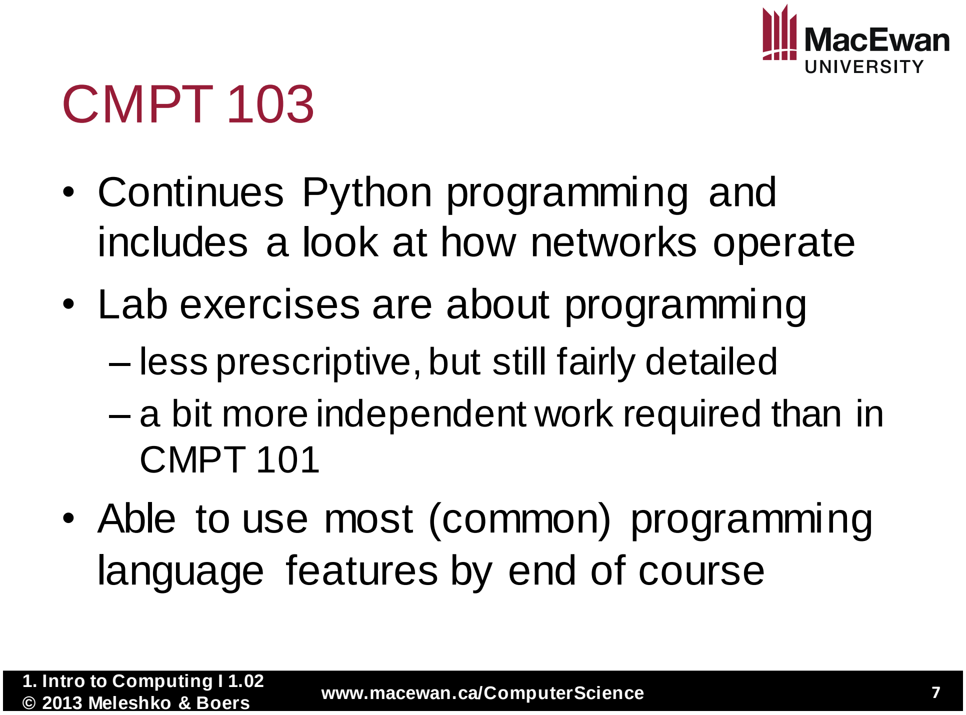
Course introduction (2/4)

- overview of computer science
  - breadth-first approach, i.e. big picture, that introduces various aspects of CS:
    - algorithms
    - programming in Python
    - Boolean logic, truth tables, circuit design
    - encoding data
    - computer architecture



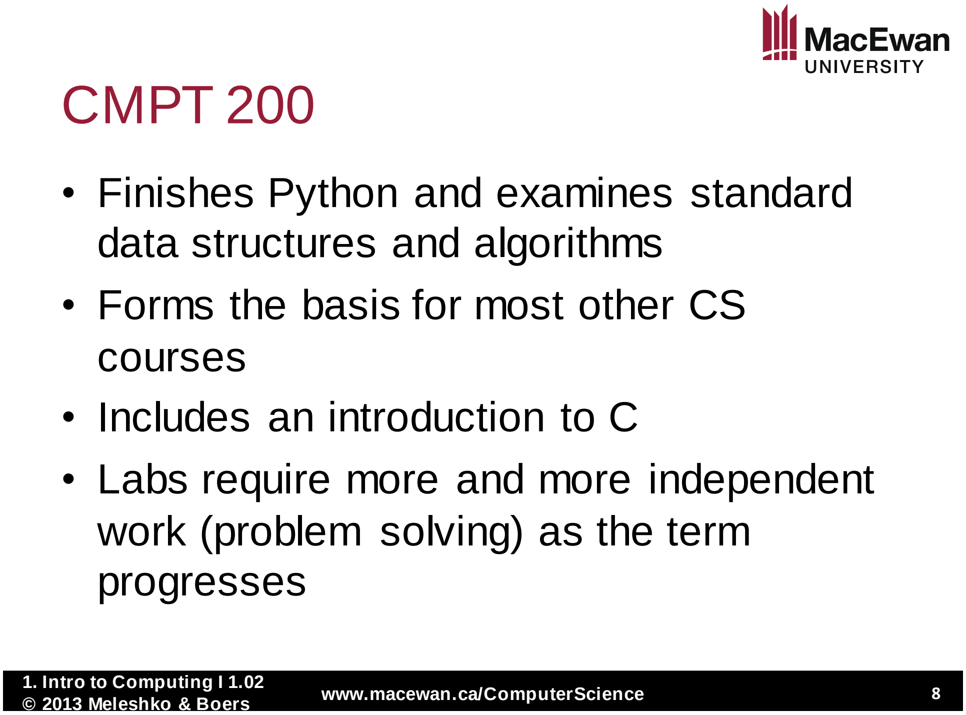
Course introduction (3/4)

- later courses:
  - CMPT 103:
    - continues the overview and Python programming
    - CMPT 101 and 103 fulfill requirements in both the Arts and Science degrees
  - CMPT 200:
    - completes introductory programming through further study of algorithms and data structures
    - basis for most advanced courses



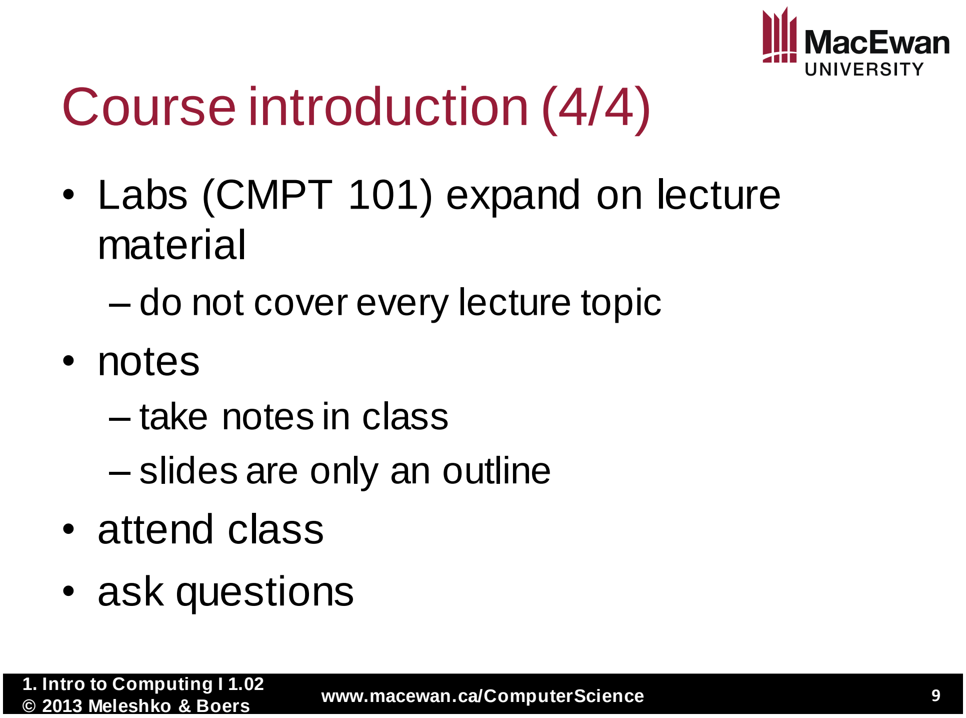
CMPT 103

- Continues Python programming and includes a look at how networks operate
- Lab exercises are about programming
  - less prescriptive, but still fairly detailed
  - a bit more independent work required than in CMPT 101
- Able to use most (common) programming language features by end of course



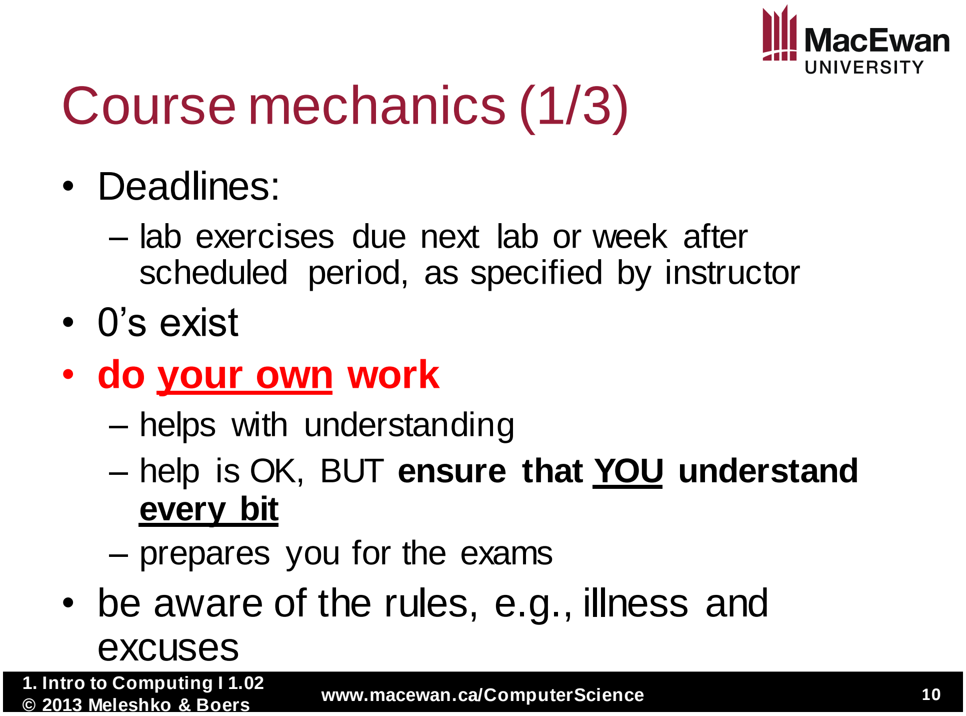
CMPT 200

- Finishes Python and examines standard data structures and algorithms
- Forms the basis for most other CS courses
- Includes an introduction to C
- Labs require more and more independent work (problem solving) as the term progresses



Course introduction (4/4)

- Labs (CMPT 101) expand on lecture material
  - do not cover every lecture topic
- notes
  - take notes in class
  - slides are only an outline
- attend class
- ask questions



Course mechanics (1/3)

- Deadlines:
  - lab exercises due next lab or week after scheduled period, as specified by instructor
- 0's exist
- **do <u>your own</u> work**
  - helps with understanding
  - help is OK, BUT **ensure that <u>YOU</u> understand <u>every bit</u>**
  - prepares you for the exams
- be aware of the rules, e.g., illness and excuses

Course mechanics (2/3)

- stay current
  - material **builds on <u>preceding</u>** material
    - difficult to catch up if you fall behind especially the programming
- refer to the textbook and/or references
  - different examples
  - more details about the code

Course mechanics (3/3)

- Blackboard:

  http://learn.macewan.ca/
  - lecture notes, course syllabus, links
- website:

  http://academic.macewan.ca/meleshkor/cmpt101
  - course outline, schedule, student responsibilities

MacEwan UNIVERSITY

Academic integrity (1/3)

- Academic Integrity Policy (C1000):
  http://www.macewan.ca/PolicyManual/
  - parts within the calendar
- rule of thumb:
  - **do <u>your own</u> work**

MacEwan UNIVERSITY

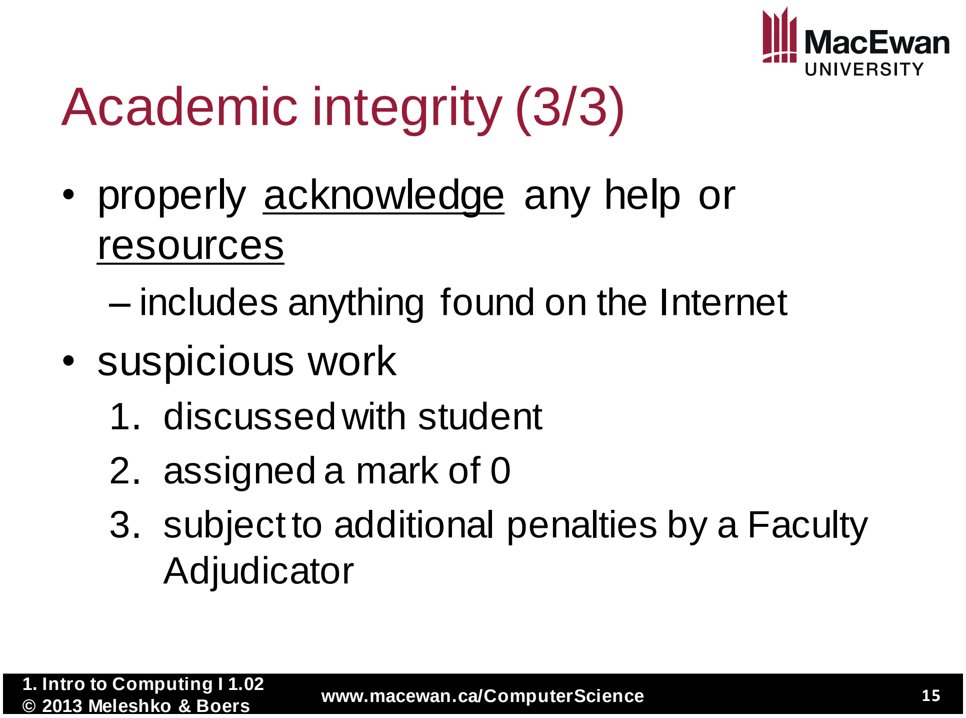
Academic integrity (2/3)

- submitting someone else’s work or part of your work:

NOT ACCEPTABLE

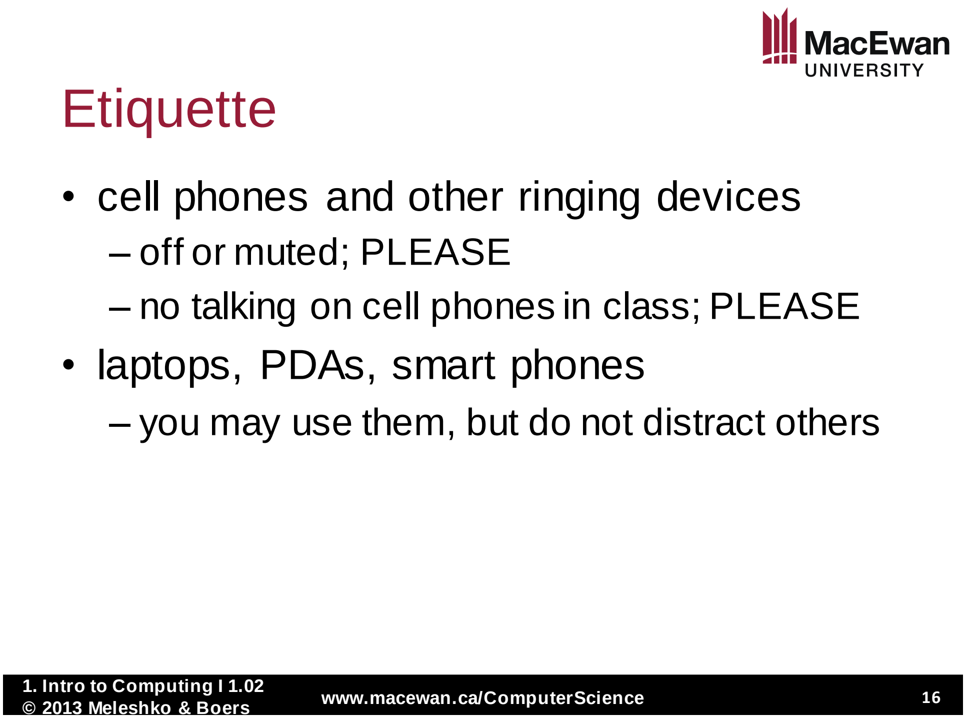
- sharing your work, or even part of your work, with someone else:

NOT ACCEPTABLE



Academic integrity (3/3)

- properly acknowledge any help or resources
  - includes anything found on the Internet
- suspicious work
  1. discussed with student
  2. assigned a mark of 0
  3. subject to additional penalties by a Faculty Adjudicator

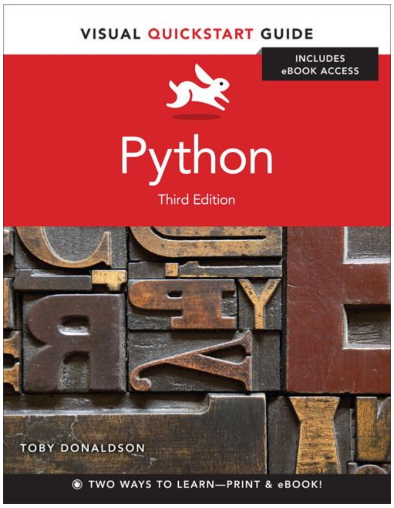


Etiquette

- cell phones and other ringing devices
  - off or muted; PLEASE
  - no talking on cell phones in class; PLEASE
- laptops, PDAs, smart phones
  - you may use them, but do not distract others

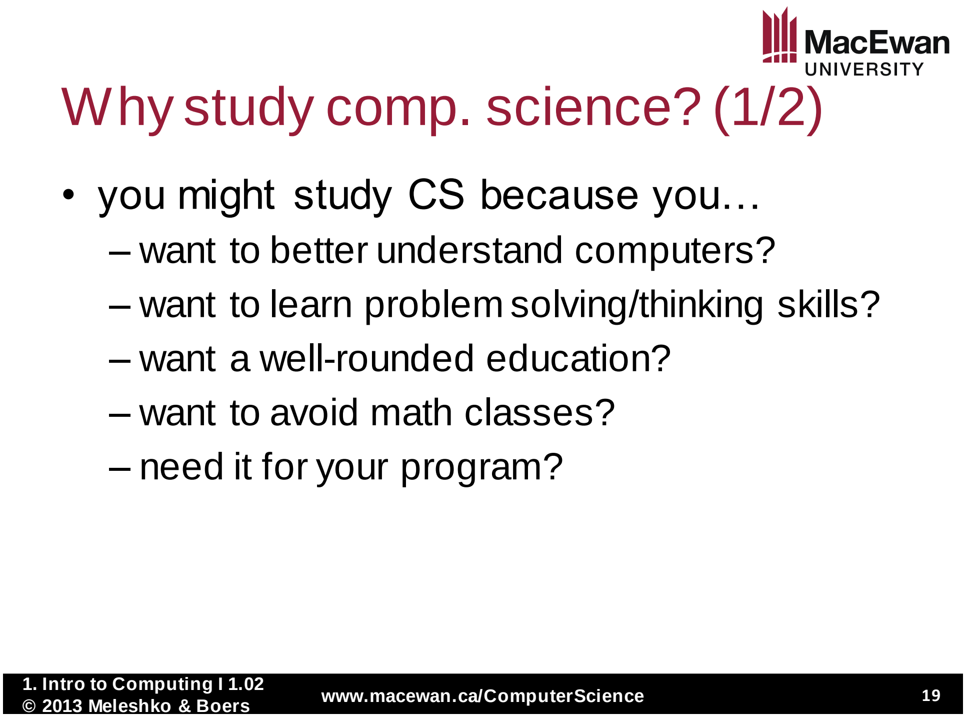
Primary Reference - Optional

Toby Donaldson, *Visual Quickstart Guide Python* (3rd ed.), Peachpit Press, 2013.



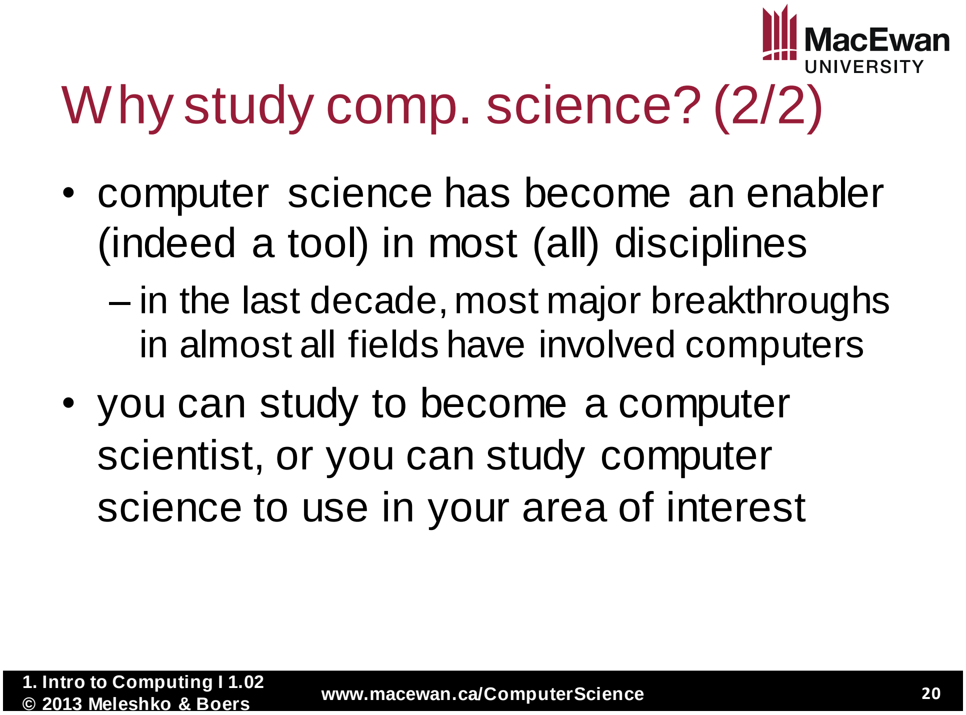
OTHER USEFUL REFERENCES

- Allen Downey. *Think Python, How to Think Like a Computer Scientist*, Green Tea Press – VERY good, FREE: www.greenteapress.com/thinkpython/thinkpython.pdf
- Quick reference and study guide, on Blackboard:
  File: Cmpt101 - Quick Ref & Study Guide 106.pdf
- Interactive Python
  http://interactivepython.org/runestone/static/pythonds/index.html
- TutorialPoints - Python
  www.tutorialspoint.com/python/
  www.tutorialspoint.com/python/python_tutorial.pdf
- Many others, easy to find with Google



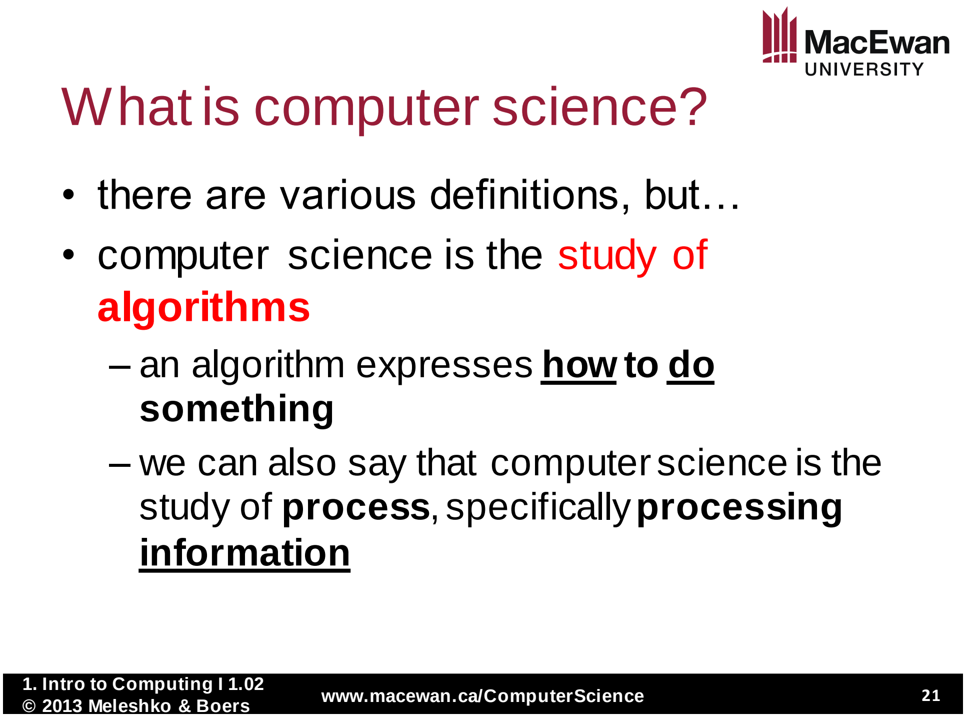
Why study comp. science? (1/2)

- you might study CS because you…
  - want to better understand computers?
  - want to learn problem solving/thinking skills?
  - want a well-rounded education?
  - want to avoid math classes?
  - need it for your program?



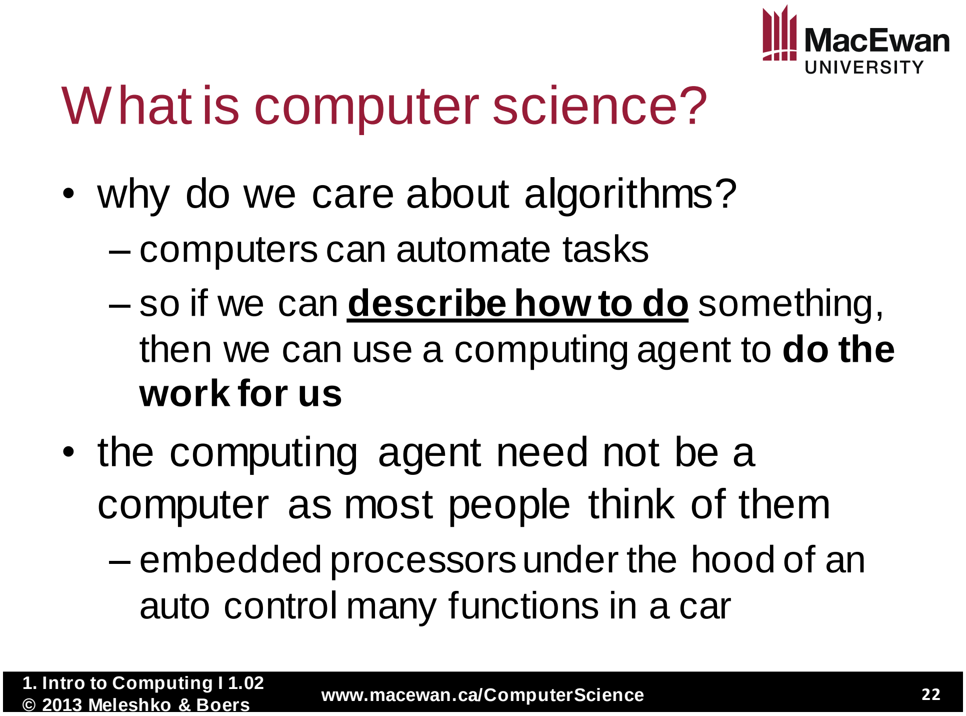
Why study comp. science? (2/2)

- computer science has become an enabler (indeed a tool) in most (all) disciplines
  - in the last decade, most major breakthroughs in almost all fields have involved computers
- you can study to become a computer scientist, or you can study computer science to use in your area of interest



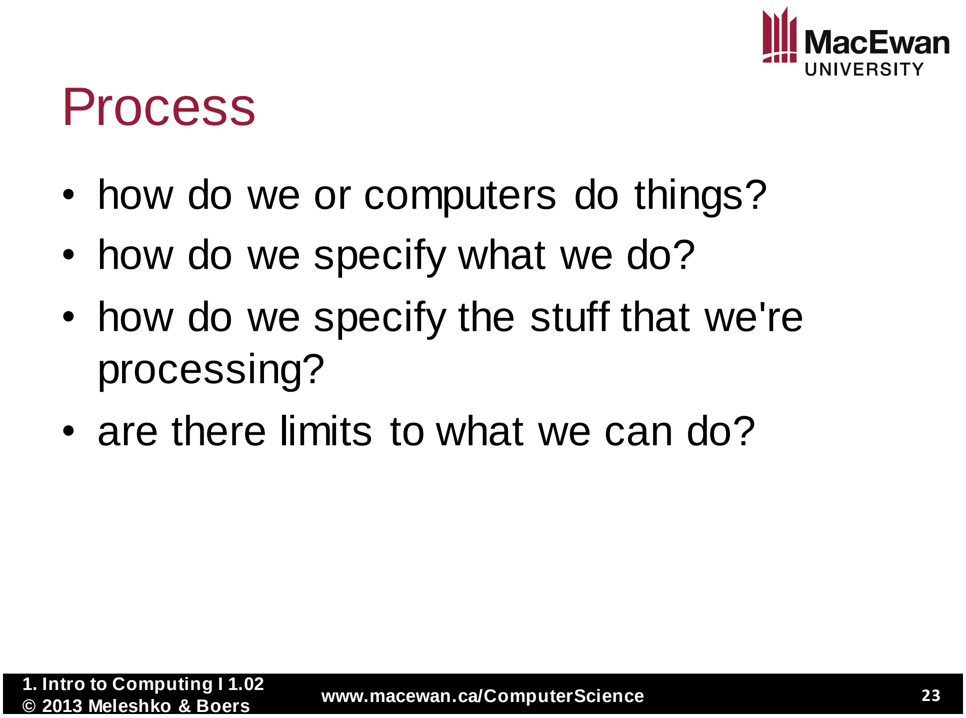
What is computer science?

- there are various definitions, but…
- computer science is the study of **algorithms**
  - an algorithm expresses **how to do something**
  - we can also say that computer science is the study of **process**, specifically **processing information**



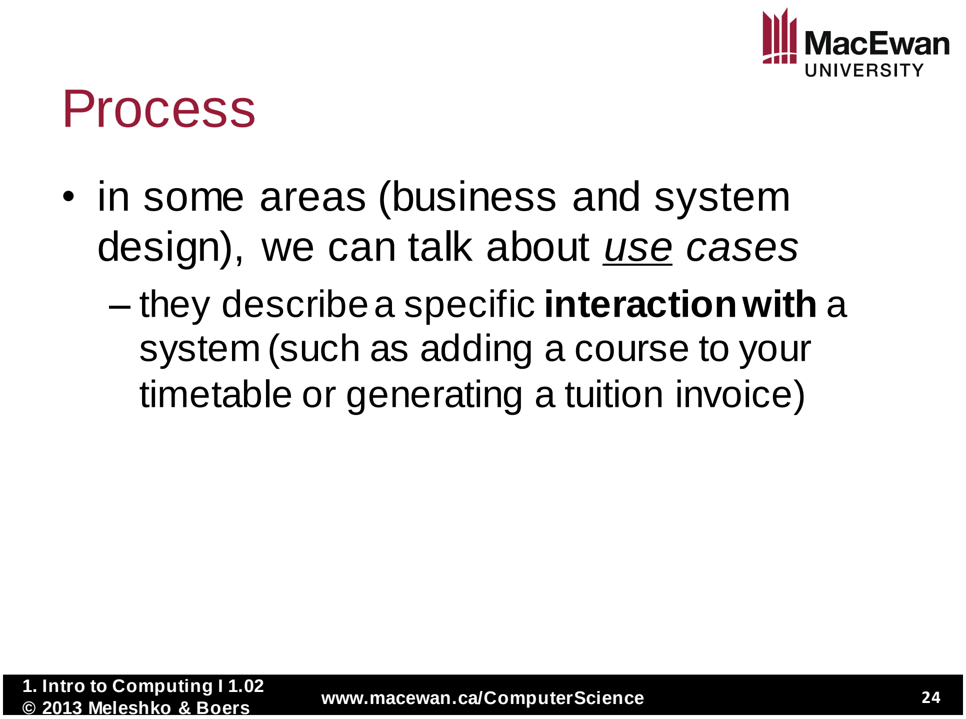
What is computer science?

- why do we care about algorithms?
  - computers can automate tasks
  - so if we can **describe how to do** something, then we can use a computing agent to **do the work for us**
- the computing agent need not be a computer as most people think of them
  - embedded processors under the hood of an auto control many functions in a car



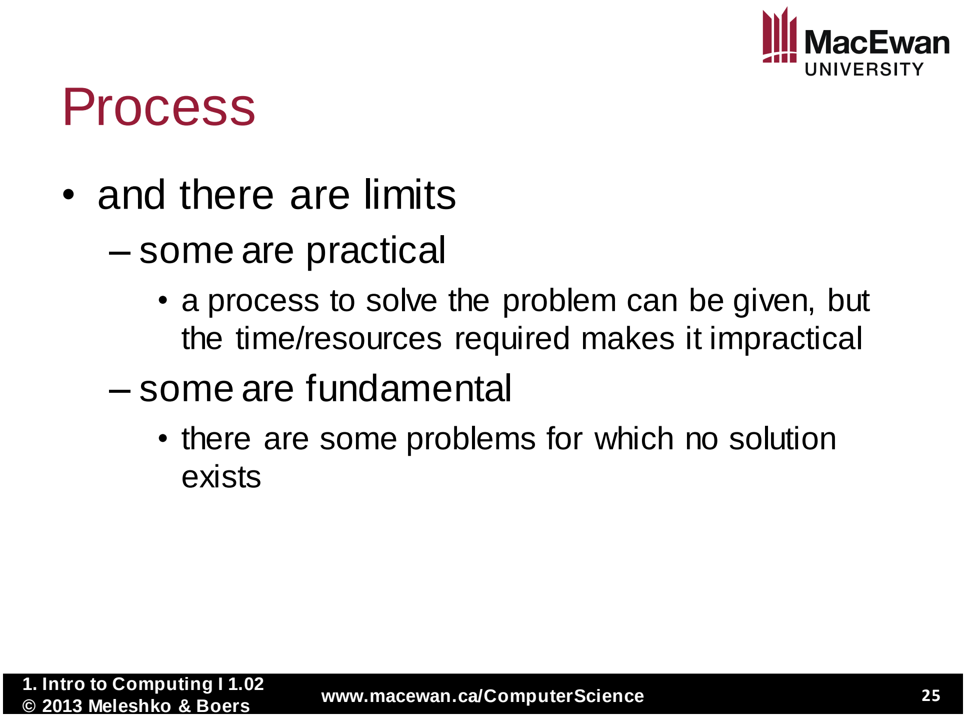
Process

- how do we or computers do things?
- how do we specify what we do?
- how do we specify the stuff that we're processing?
- are there limits to what we can do?



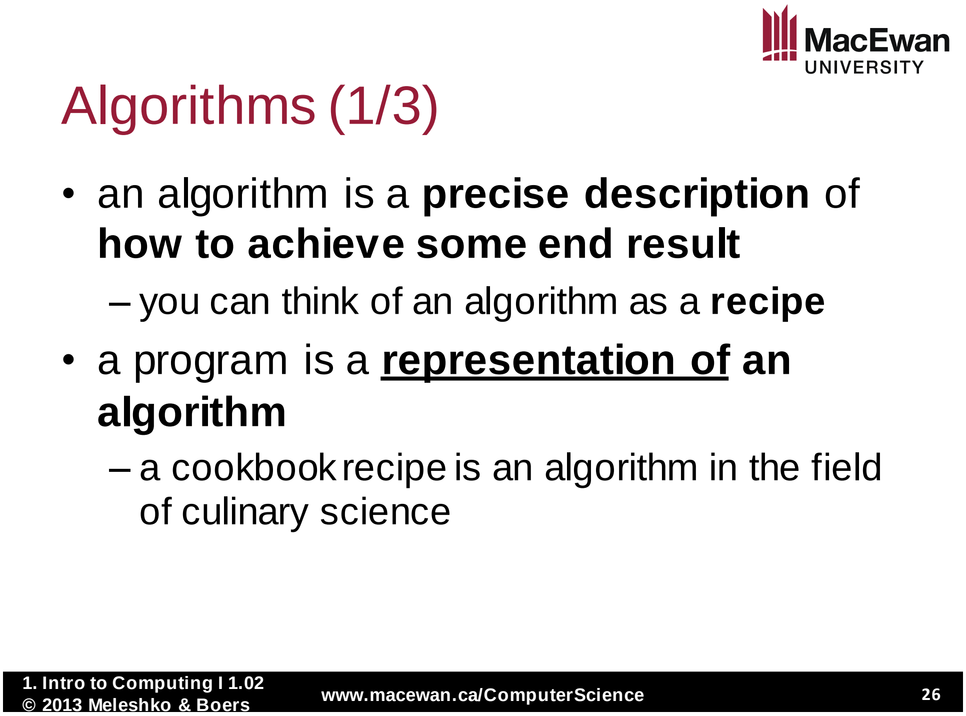
Process

- in some areas (business and system design), we can talk about *<u>use</u> cases*
  - they describe a specific **interaction with** a system (such as adding a course to your timetable or generating a tuition invoice)



Process

- and there are limits
  - some are practical
    - a process to solve the problem can be given, but the time/resources required makes it impractical
  - some are fundamental
    - there are some problems for which no solution exists



Algorithms (1/3)

- an algorithm is a **precise description** of **how to achieve some end result**
  - you can think of an algorithm as a **recipe**
- a program is a **representation of an algorithm**
  - a cookbook recipe is an algorithm in the field of culinary science

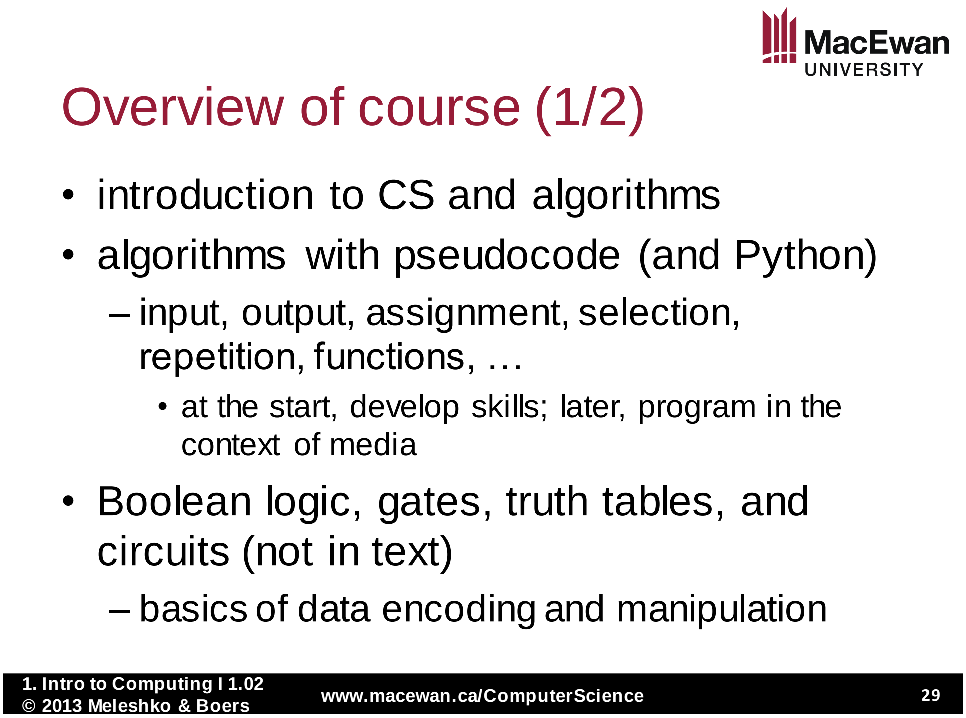
Algorithms (2/3)

- examples:
  - instructions for opening a combination lock
  - instructions for converting an audio CD to a folder of MP3s
  - quadratic formula for roots of a 2nd degree polynomial

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

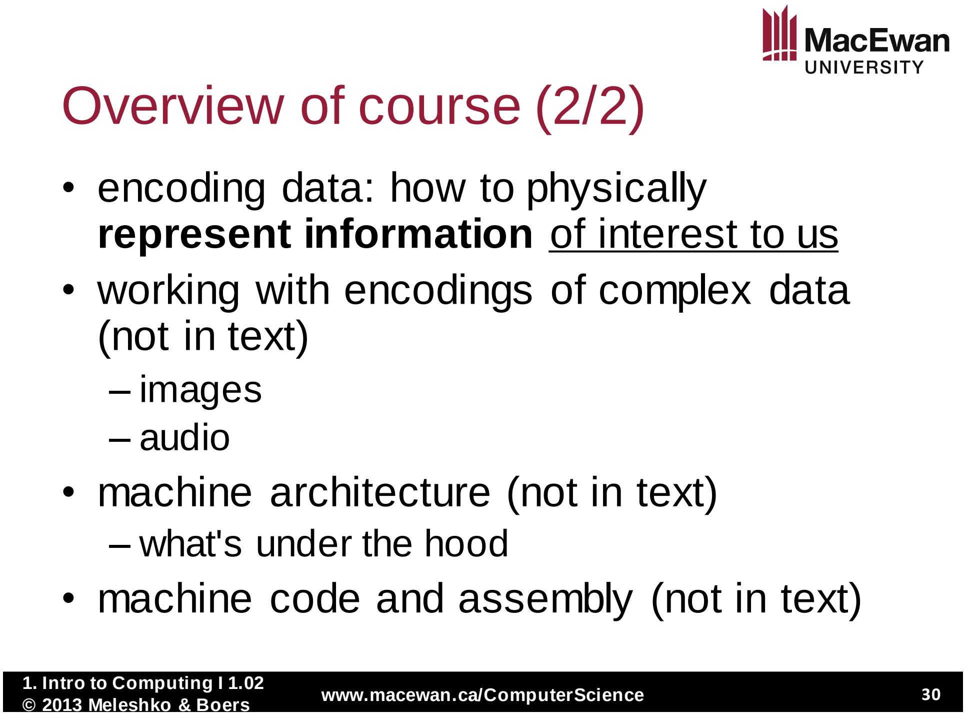
Algorithms (3/3)

- given a good algorithm (or recipe), we don't need to understand the problem being solved; just follow the instructions
- an algorithm allows us to **automate a task**
  - programs, hardware implementations, people
  - specific kind of recipes: graphics, computer music, games, e-commerce and web sites, numerical software



Overview of course (1/2)

- introduction to CS and algorithms
- algorithms with pseudocode (and Python)
  - input, output, assignment, selection, repetition, functions, …
    - at the start, develop skills; later, program in the context of media
- Boolean logic, gates, truth tables, and circuits (not in text)
  - basics of data encoding and manipulation



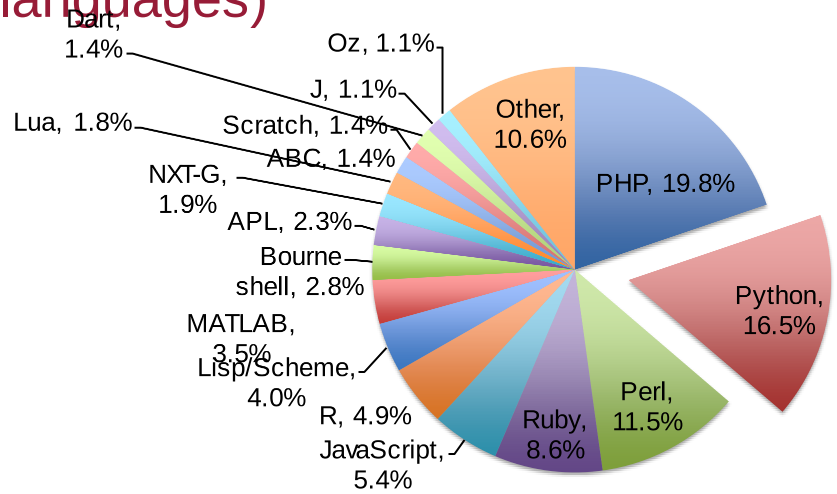
Overview of course (2/2)

- encoding data: how to physically **represent information** of interest to us
- working with encodings of complex data (not in text)
  - images
  - audio
- machine architecture (not in text)
  - what's under the hood
- machine code and assembly (not in text)

Programming languages

- computer science is not programming
  - programming: a skill that enables you to implement algorithms using a computer
- many programming languages exist
  - each with their own strengths/weaknesses
- choice of a first language is contentious
  - Python is becoming more popular

Python's popularity (scripting languages)



Source: http://lang-index.sourceforge.net from July 1, 2013

Overall Popularity (July 1, 2013)

- C – 17.7%
- Java – 14.7%
- Objective-C – 8.2%
- C++ - 6.8%
- Basic – 5.5%
- PHP – 4.4%
- Python – 3.7%
- C# - 3.3%

Python (1/2)

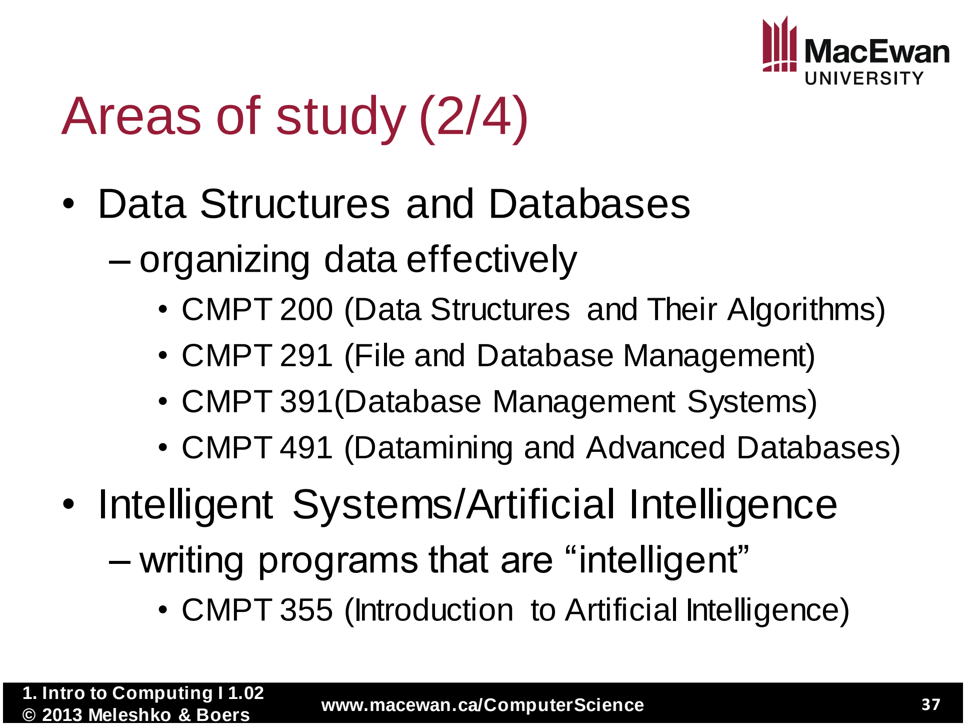
- designed to be simple and easy to understand
- extensible
- named after Monty Python, not the snake

Python (2/2)

- Python is an interpreted language
  - lines are executed as they are entered
  - you can load prepared lines from a file
- interpretation means that the code will run slower than a compiled language
  - a compiled language is translated into the hardware language of the computer's processor
- Note: Python code can also be compiled for faster execution

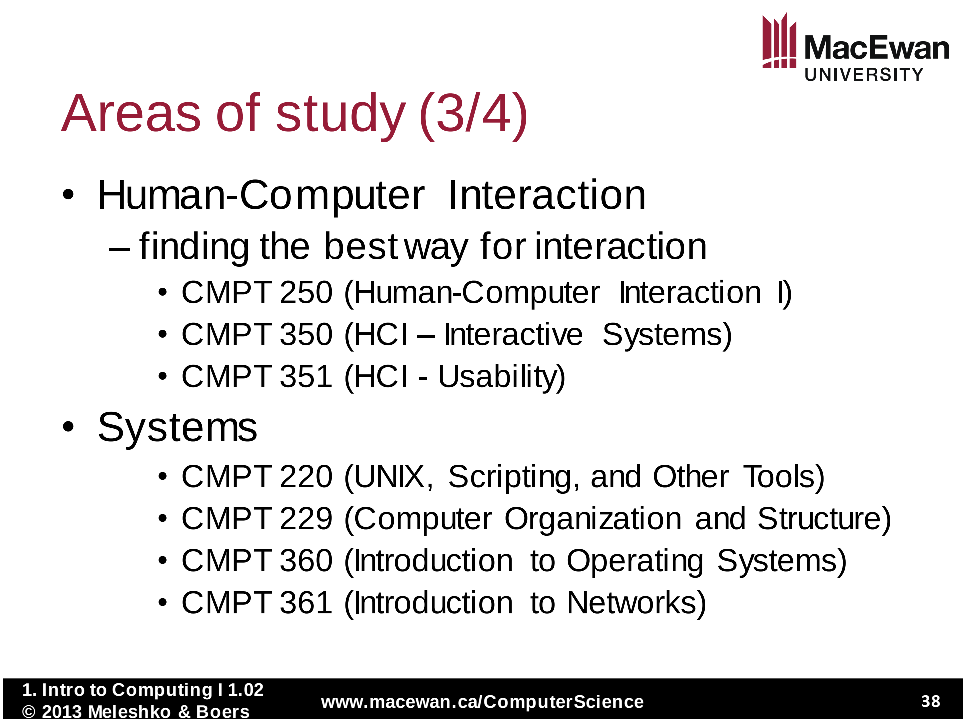
Areas of study (1/4)

- Theory
  - evaluating and comparing algorithms
  - finding better ways of doing things
    - CMPT 204 (Algorithms I)
- Software Engineering
  - collaborating effectively within large groups
    - CMPT 395 (Introduction to Software Engineering)



Areas of study (2/4)

- Data Structures and Databases
  - organizing data effectively
    - CMPT 200 (Data Structures and Their Algorithms)
    - CMPT 291 (File and Database Management)
    - CMPT 391(Database Management Systems)
    - CMPT 491 (Datamining and Advanced Databases)
- Intelligent Systems/Artificial Intelligence
  - writing programs that are “intelligent”
    - CMPT 355 (Introduction to Artificial Intelligence)



Areas of study (3/4)

- Human-Computer Interaction
  - finding the best way for interaction
    - CMPT 250 (Human-Computer Interaction I)
    - CMPT 350 (HCI – Interactive Systems)
    - CMPT 351 (HCI - Usability)
- Systems
    - CMPT 220 (UNIX, Scripting, and Other Tools)
    - CMPT 229 (Computer Organization and Structure)
    - CMPT 360 (Introduction to Operating Systems)
    - CMPT 361 (Introduction to Networks)

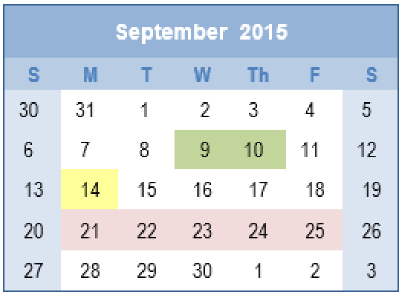
MacEwan UNIVERSITY

Areas of study (4/4)

- Graphics and Gaming
  - designing games
    - CMPT 230 (Introduction to Computer Games)
    - CMPT 330 (Introduction to Real Time Gaming)
    - CMPT 370 (Introduction to Computer Graphics)
    - CMPT 430 (3D Game Development and AI)
- There are other areas, too…
    - numerical methods, web-centric computing, ethics/law, non-procedural languages, ….

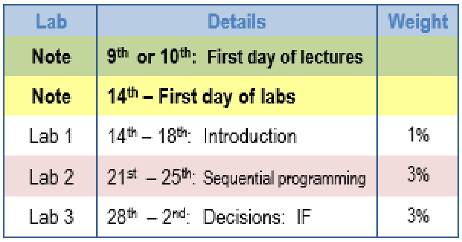
MacEwan UNIVERSITY

September



September 2015

S	M	T	W	Th	F	S
30	31	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	1	2	3



Lab	Details	Weight
Note	**9th or 10th: First day of lectures**	
Note	**14th – First day of labs**	
Lab 1	14th – 18th: Introduction	1%
Lab 2	21st – 25th: Sequential programming	3%
Lab 3	28th – 2nd: Decisions: IF	3%

- September 14th is the first week of labs

MacEwan UNIVERSITY

October

October 2015						
S	M	T	W	Th	F	S
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Lab	Details		Weight
Lab 4	5th – 9th:	Repetition: Loops	3%
	Oct 12th:	Thanksgiving (No lab)	
Lab 5	12th – 16th:	Functions	3%
Lab 6	19th – 23rd:	Lists	3%
Lab 7	26th – 30th:	Sorting	3%

- No labs on Thanksgiving (Monday, October 12th).
- **Monday labs** will have to make up the missed lab work. Details will be provided later on in the semester.

www.macewan.ca/ComputerScience

MacEwan UNIVERSITY

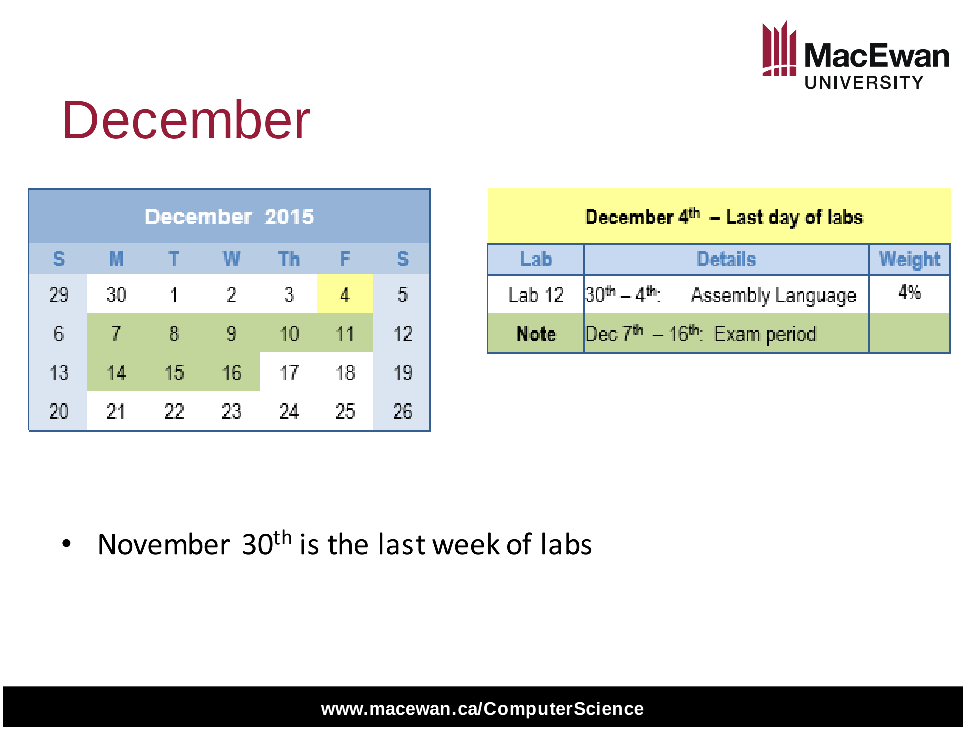
November

November 2015						
S	M	T	W	Th	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28

	Nov 11th:	Remembrance day (No lab)	
Lab	Details		Weight
Lab 8	2nd – 6th:	Circuit design	3%
Lab 9	9th – 13th:	Binary Numbers	3%
Lab 10	16th – 20th:	Image modification I	3%
Lab 11	23rd – 27th:	Image modification II	3%

- No labs on Remembrance Day (Wednesday, November 11th).
- **Wednesday labs** will have to make up the missed lab work. Details will be provided later on in the semester.

www.macewan.ca/ComputerScience



MacEwan UNIVERSITY

December

December 2015						
S	M	T	W	Th	F	S
29	30	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26

December 4th – Last day of labs

Lab	Details	Weight
Lab 12	30th – 4th: Assembly Language	4%
Note	Dec 7th – 16th: Exam period	

- November 30th is the last week of labs

www.macewan.ca/ComputerScience