CMPUT 175 (B1, B2)
Introduction to Foundations of Computing

UNIVERSITY OF ALBERTA

CIFAR amii

Osmar R. Zaïane, Ph.D.
McCalla, Killam Professor
Department of Computing Science

Amii Fellow
CIFAR AI Chair

443 Athabasca Hall
Edmonton, Alberta
Canada T6G 2E8

Telephone: Office +1 (780) 492 2860
Fax +1 (780) 492 1071
E-mail: **zaiane@cs.ualberta.ca**
http://www.cs.ualberta.ca/~zaiane/

奥斯馬爾 蔡頤安 زيان @ozaiane

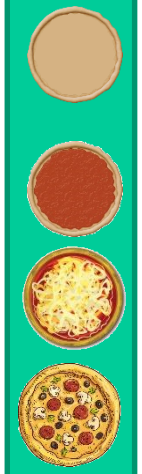
Let's make a vegetarian Pizza

Ingredients

Input

Recipe

1. Preheat the oven at 220 degrees C
2. Roll the dough in the shape of a large plate and place in a pizza pan
3. Bake the pizza crust until just crisp
4. Take out of the oven and spread tomato sauce on crust
5. Spread grated cheese
6. Cut vegetables as desired
7. Evenly spread vegetables on the cheese
8. Bake in the preheated oven 7 to 10 minutes until the cheese has melted and is bubbly
9. Take out of the oven, let it cool slightly. **Bon Appétit!**



Algorithm

Pizza

Output

Assumptions Made

Input

Ingredients

Recipe

1. Preheat the oven at 220 degrees C
2. Roll the dough in the shape of a large plate and place in a pizza pan
3. Bake the pizza crust until just crisp
4. Take out of the oven and spread tomato sauce on crust
5. Spread grated cheese
6. Cut vegetables as desired
7. Evenly spread vegetables on the cheese
8. Bake in the preheated oven 7 to 10 minutes until the cheese has melted and is bubbly
9. Take out of the oven, let it cool slightly. **Bon Appétit!**

Algorithm

Pizza

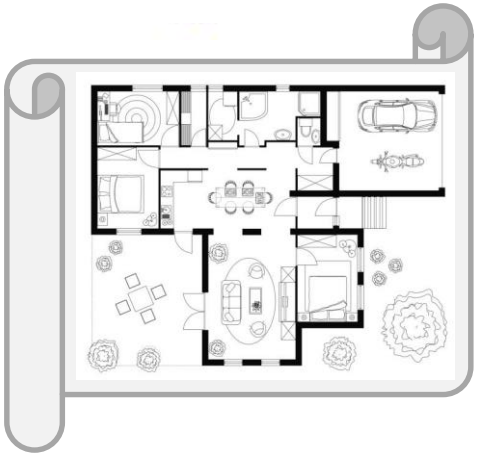
Output

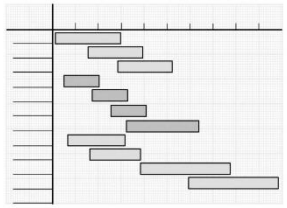
Tools

How ?

Another Procedure

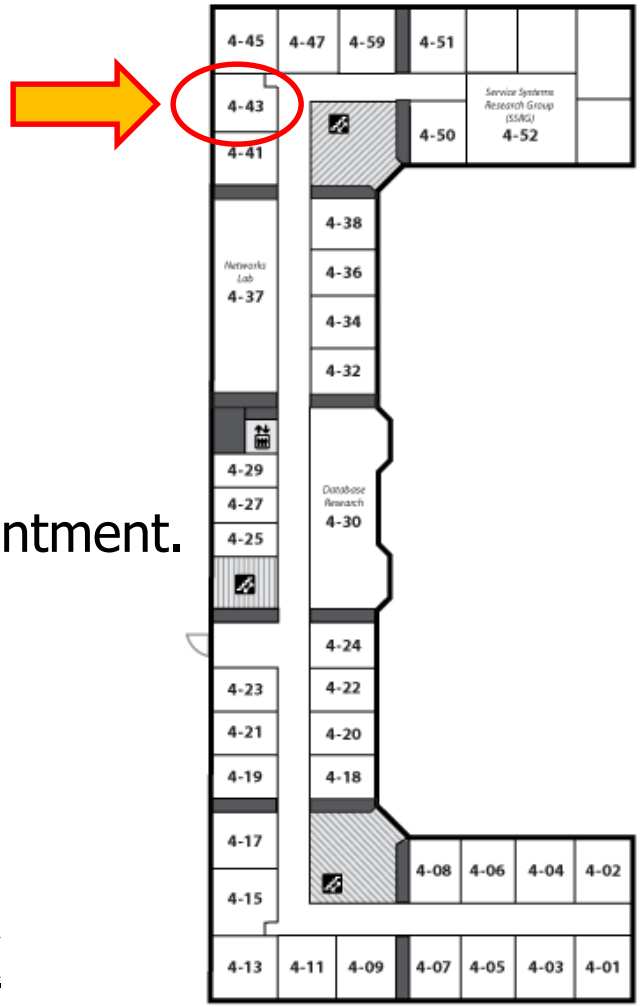
Introduction to Foundations of Computing





Instructor Information

- Prof. Osmar Zaïane
- Office: Athabasca Hall, room 4-43
- Office hours: 1:00-2:00pm, every Tuesday and Thursday (On-Line)
  or (exceptionally) by mutually agreed upon appointment.
  If in the first 30m no one shows up, the rest is canceled.
- TAs also have office hours
- E-mail: zaiane@cs.ualberta.ca
  (use your UofA account and prefix subject line with "**CMPUT175**")



Course Staff

- Instructors

Osmar Zaïane
B1 9:00-9:50
CCIS L1-140
168 students
B2 10:00-10:50
CCIS L1-140
170 students

Sadaf Ahmed
B3 12:00-12:50
CSC B2
170 students
B4 14:00-14:50
CCIS L 1-140
169 students

3 hour labs per week

- Teaching Assistants (B1, B2, B3, B4)
  - **Kseniia Priakhina** priakhin@ualberta.ca
  - **Shahin Atakishiyev** atakishi@ualberta.ca
  - **Abdulsalam Ba Sabaa** basabaa@ualberta.ca
  - **Dong Huo** dhuo@ualberta.ca
  - **G M Mashrur E Elahi** eelahi@ualberta.ca
  - **Yan Wang** yan28@ualberta.ca
  - **Mohammad Karimi Abdolmaleki** karimiab@ualberta.ca
  - **Ruiqin Pi** ruiqin@ualberta.ca
  - **Negar Mirgati** mirgati@ualberta.ca
  - **Ningyuan Pei** ningyuan@ualberta.ca
  - **Nathan Henderson** nthender@ualberta.ca
  - **Reza Mashayekhi** rmashaye@ualberta.ca
  - **Katyani Singh** katyani@ualberta.ca
  - **Mohan Sai Singamsetti** singamse@ualberta.ca
  - **Karan Chadha** kchadha1@ualberta.ca
  - **Mohammadjavad Matinkia** matinkia@ualberta.ca
  - **Kushankur Ghosh** kushanku@ualberta.ca
  - **Tahsin Mostafa** tmostafa@ualberta.ca
  - **Zahra Bashir** zbashir1@ualberta.ca
  - **Abilmansur Zhumabekov** zhumabek@ualberta.ca
  - **Rudraksh Kapil** rkapil@ualberta.ca
  - **Ali Hossein Gharari Foomani** hosseing@ualberta.ca
  - **Emily Halina** ehalina@ualberta.ca
  - **Sriram Karthik Akella** sriramka@ualberta.ca
  - **Vyome Agarwal** vyome@ualberta.ca
  - **Mahima Dhawan** mdhawan@ualberta.ca

Lectures and Labs

	8-9AM	9-10AM	10-11	11-12	12-1PM	1-2PM	2-3PM	3-4
Monday		B1	B2		B3		B4	
Tuesday								
Wednesday		B1	B2		B3		B4	
Thursday								
Friday		B1	B2		B3		B4	

	8-9	9-10	10-11	11-12	12-1	1-2	2-3	3-4	4-5	5-6	6-7	7-8
Monday					H1	23/23		H2	23/23		H3	23/23
Tuesday		H5	23/23					H8	23/23		H9	23/23
Wednesday					H11	23/23		H12	23/23		H14	23/23
Thursday		H16	23/23		H17	23/23		H18	23/23		H20	23/23
Friday		H21	23/23		H22	23/23		H23	23/23			

	8-9	9-10	10-11	11-12	12-1	1-2	2-3	3-4	4-5	5-6	6-7	7-8
Monday												
Tuesday	CSC 129	H4	24/24	ED 128	H6	54/54	ED 128	H7	53/54	CSC 129	H10	23/23
Wednesday										CAB 269	H13	54/54
Thursday	CSC 125	H15	24/24							CSC 129	H19	23/23
Friday							AF 113	H24	53/53			

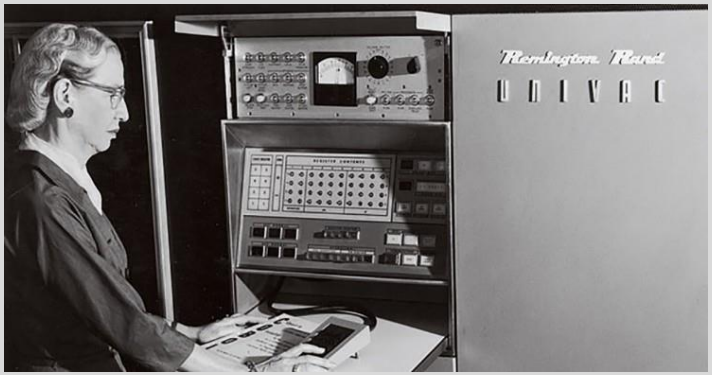
CSC-1-29
ED 128
CAB 269
CSC 1-25
AF 1-13
CSC-1-53

Computing Science Class 1980s

26 students: 14 girls and 12 boys

Evelyn Berezin in 1976, when she was president and founder (1969) of the Redactron Corporation, with Data Secretary, the first computerized word processor, which she designed and marketed.
CreditCreditBarton Silverman/The New York Times

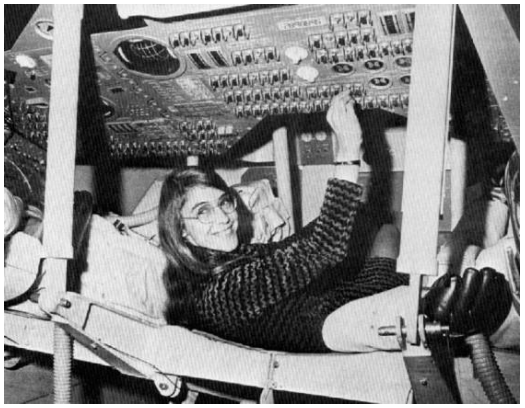
She died in Manhatan December 8th, 2018. She was 93.



Grace Hopper laid the foundation for the Information Age
Suggested the use of English language words as instructions rather than opaque symbols and numbers.
Developed the first compiler in 1949.
Introduced the notion of code reuse.

Women and Programming

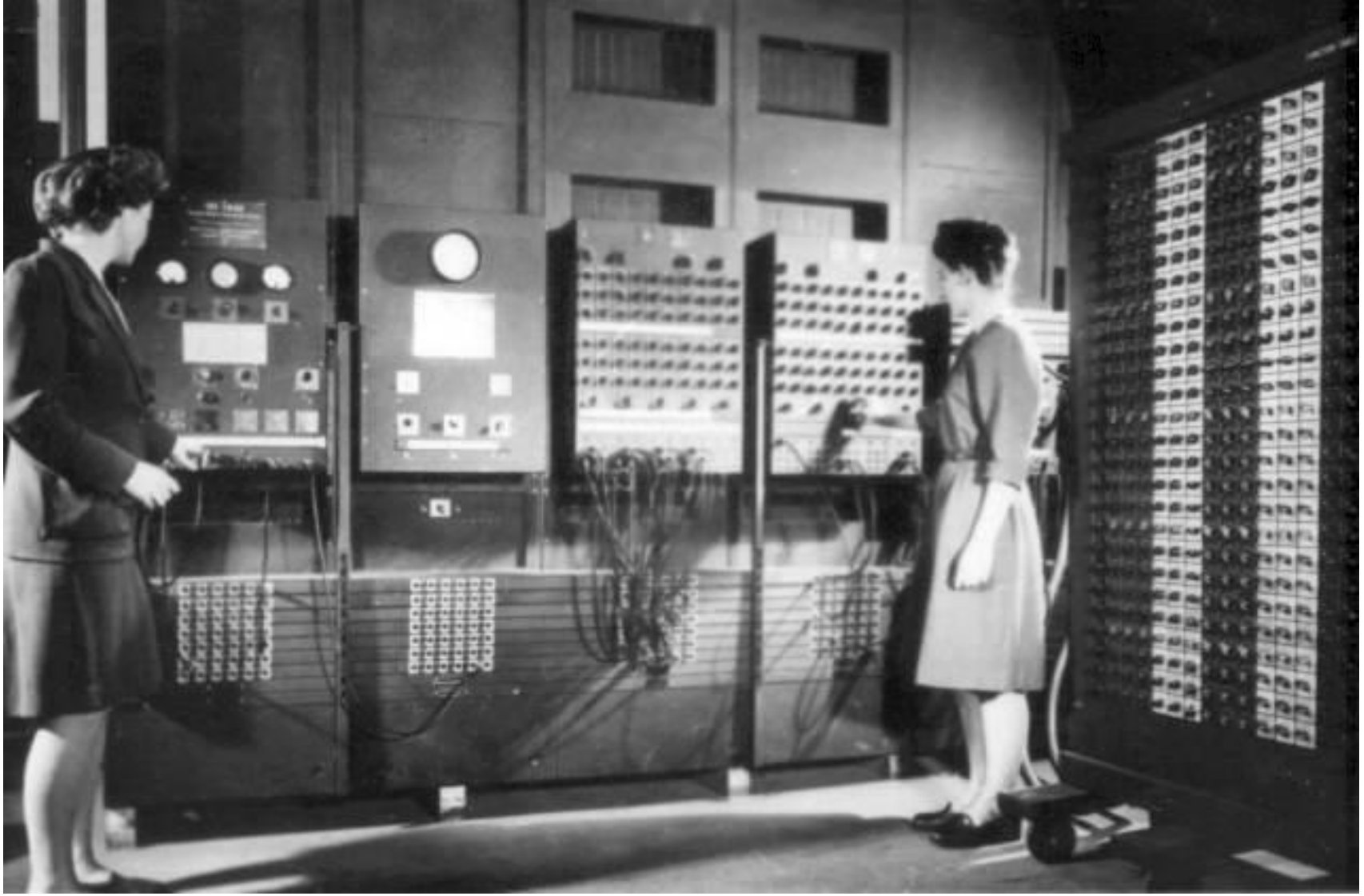
1969: standing beside listings of the software developed by the team she was in charge of.



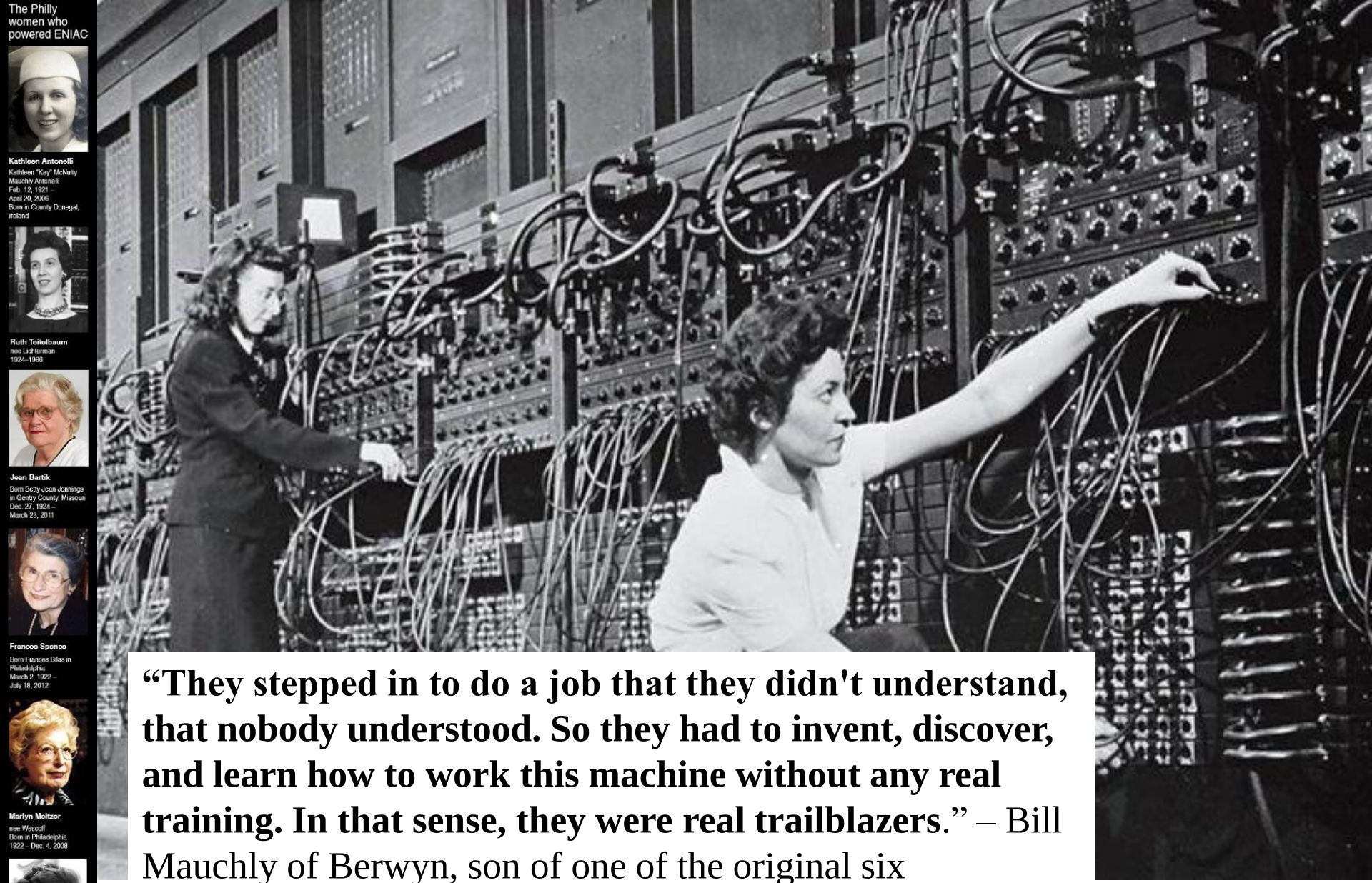
Coined the term "Software Engineering"

Margaret Hamilton, the Engineer who took the Apollo to the Moon

Thanks to her Neil Armstrong and Buzz Aldrin could take a walk on the Moon.



Electronic Numerical Integrator and Computer (ENIAC), the world's first general-purpose electronic computer. Feb 15 (2011) Eniac Day (Philadelphia) University of Pennsylvania.



"They stepped in to do a job that they didn't understand, that nobody understood. So they had to invent, discover, and learn how to work this machine without any real training. In that sense, they were real trailblazers." – Bill Mauchly of Berwyn, son of one of the original six programmers

Some Rules

Cellphones should be turned off in Class/Lab
No cellphones or PDAs in exams

No recording (audio or video) without explicit authorization by instructor.

Laptops or Tablets allowed exclusively for taking notes. **No Facebooking, emailing, etc.**

Doctor's note not required to justify absence.
There is a form to fill in (statutory declaration).
Instructor would have the final say.

Deferral of term work is a privilege and not a right

Prerequisites

- The course draws heavily on CMPUT 174

Computing Science

Computing science is not about computers.
Computing science is about **Problem Solving.**
Comput**er** science is a misnomer. Think astronomer and telescope

Computing science is about developing a step-by-step list of instructions to solve an instance of a given problem.
The ordered list of instructions is called an **Algorithm.**

For a given problem there could be many possible algorithms to solve it. Some problems are not solvable (i.e. don't have a known solution). They are not computable.

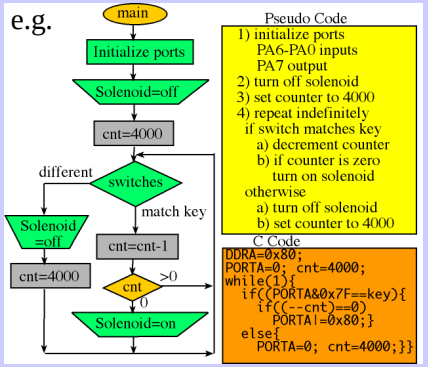
Algorithms are converted into programs for the computers to interpret them (i.e. execute them).

Prerequisites

- The course draws heavily on CMPUT 174

Programming

Programming = Data + Algorithms



Without an algorithm there can be no program

Thinking in pseudo-code

The way to go

Data Structures + Pseudo-code Algorithms

Thinking in Python

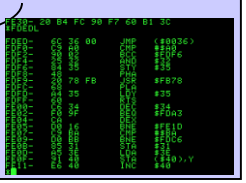
Not recommended

Translated into a programming language:
e.g. Python, Java, C++, Perl, Pascal, Basic, etc.

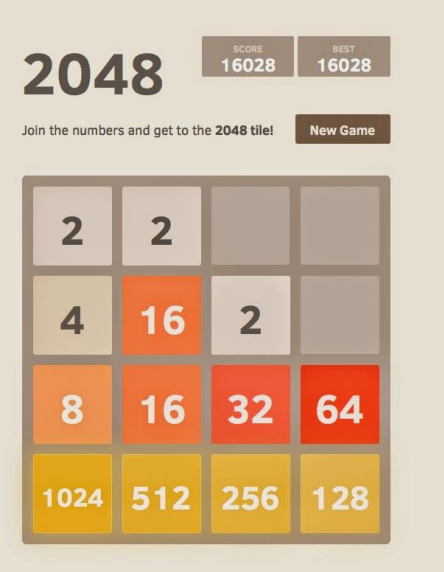
Thinking in Machine code

Impossible

Compiled or interpreted into machine code



Example for illustration



Data + Algorithms

4 x 4 matrix
Direction choice

- Add one new 2 or 4 in random empty cell
- Get direction; Move cells based on direction
- If two adjacent cells have the same value, Add them up in one cell in the same direction
- Stop when one cell has 2048 or all cells full

Data Structures + Pseudo-code Algorithms

A list of 4 lists of 4 elements
A characters

M=[[2,2,0,0],[4,16,2,0],[8,16,32,64],[1024,512,256,128]]
ch= “a”

- Add a 2 or 4 in any cell with a 0 (random)
- Get direction
- If direction is left
    for each line
        move all cells to the left
            if two cells are identical
                add them up in one cell

…

Prerequisites

- The course draws heavily on CMPUT 174

Computers are dumb

A program is a list of instructions that instruct the computer what to do. The computer does exactly what it was instructed to do; nothing more, nothing less.

Computers take instructions literally as they are given to them. **If the computer doesn't do what you expect it to do that means the algorithm or the program is wrong.**

Computers can repeat things again and again, and are usually fast

Computers don't remember anything unless you instruct them to memorize information.

Course Structure

- 3 lecture hours + 3 lab hours per week + online videos for each week
- Topics
  - Catching exceptions, iterators, inheritance
  - Data structures: containers, lists, stacks, queues, trees, hash tables
  - Recursion
  - Sorting and searching
  - Efficiency of Algorithms - Complexity analysis: time and space

Learning Objectives

By the end of this course, students should be able to:

A. Use a range of predefined python data structures including strings, sets, lists, nested lists (e.g. 2D lists) and dictionaries in a program.
B. Write python programs that handle exceptions including throwing and catching exceptions, as well as assertions.
C. Write python programs which use files for reading input and writing output.
D. Generate and format strings using python for output on a screen or file.
E. Analyse algorithms/programs and compare their efficiency in terms of time and space complexity.
F. Manually trace the execution of a python program and explain the expected outcome.
G. Debug a python program by tracing control flow and variable states using a debugger.
H. Extend a python class by exploiting inheritance, overriding its methods and augmenting its interface.
I. Design new Abstract Data Types (ADTs) when the need arises and implement the corresponding data structures in python.
J. Appropriately employ encapsulation and information hiding when writing a program in python
K. Use abstraction to express object behaviour using a class interface and implement it using a class.
L. Use recursion in solving iterative problems and traversing specific data structures.

Values Objectives

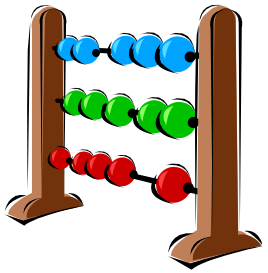
- Value the importance of devising a well-designed algorithm prior to coding a program.
- Appreciate the benefits and importance of isolation testing (unit testing) prior to testing and executing a full program.
- Acknowledge that different solutions to the same problem may diverge significantly in terms of solution quality, time efficiency and space requirements even if the output is the same.

Course Web Site

- Everything is on e-class – (with CCID login)
- https://eclass.srv.ualberta.ca/course/view.php?id=74934
- Slides, Videos, Labs, assignments, resources, etc.

Go to:
https://eclass.srv.ualberta.ca/my/

Final Mark



No Final Exam but 3 Midterms

- Breakdown:

Lab Exercises (10):	10%	35%
Assignments (3):	20%	
Video online assessment	5%	
In-class Assessment	5%	65%
Midterm 1 (Feb 14th):	15%	
Midterm 2 (Mar 14th):	20%	
Midterm 3 (Apr 8th):	25%	

Last day of classes

In-Class Assessment

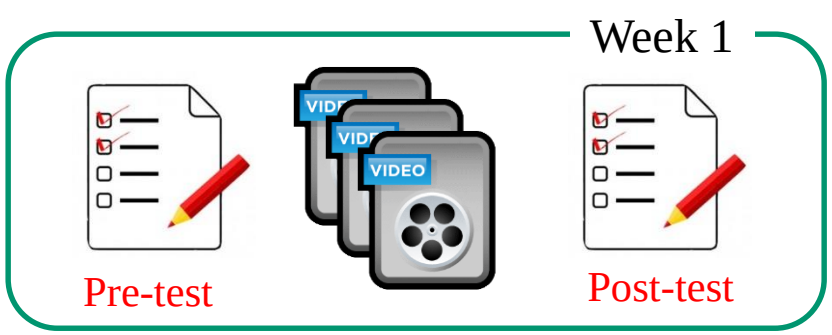
To encourage attendance and regular revision

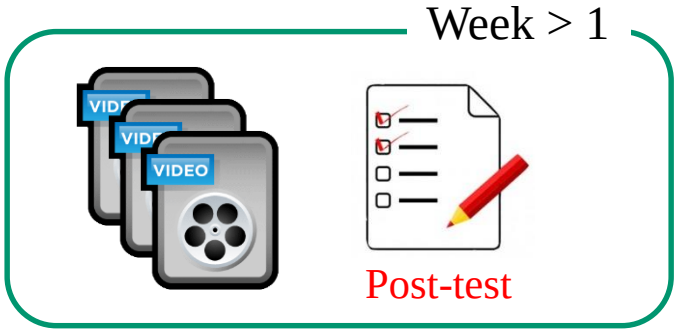
- 5% of the overall grade
- In each lecture, up to 3 names will be randomly selected and asked a simple question about the previous lecture.
- If student absent → -3/10 (all students start with a full mark)
- Student oral answer marked out of 3/10
- Student might be called up to 3 times in the semester
- If student is never called → full mark.

Video Online Assessment

To encourage preparation before class

- 5% of the overall grade
- Each week.
- One or more short video lectures to see
- Short Post Test (with end-date)
- First week: Pre-test → no mark.
  Post-test marked.

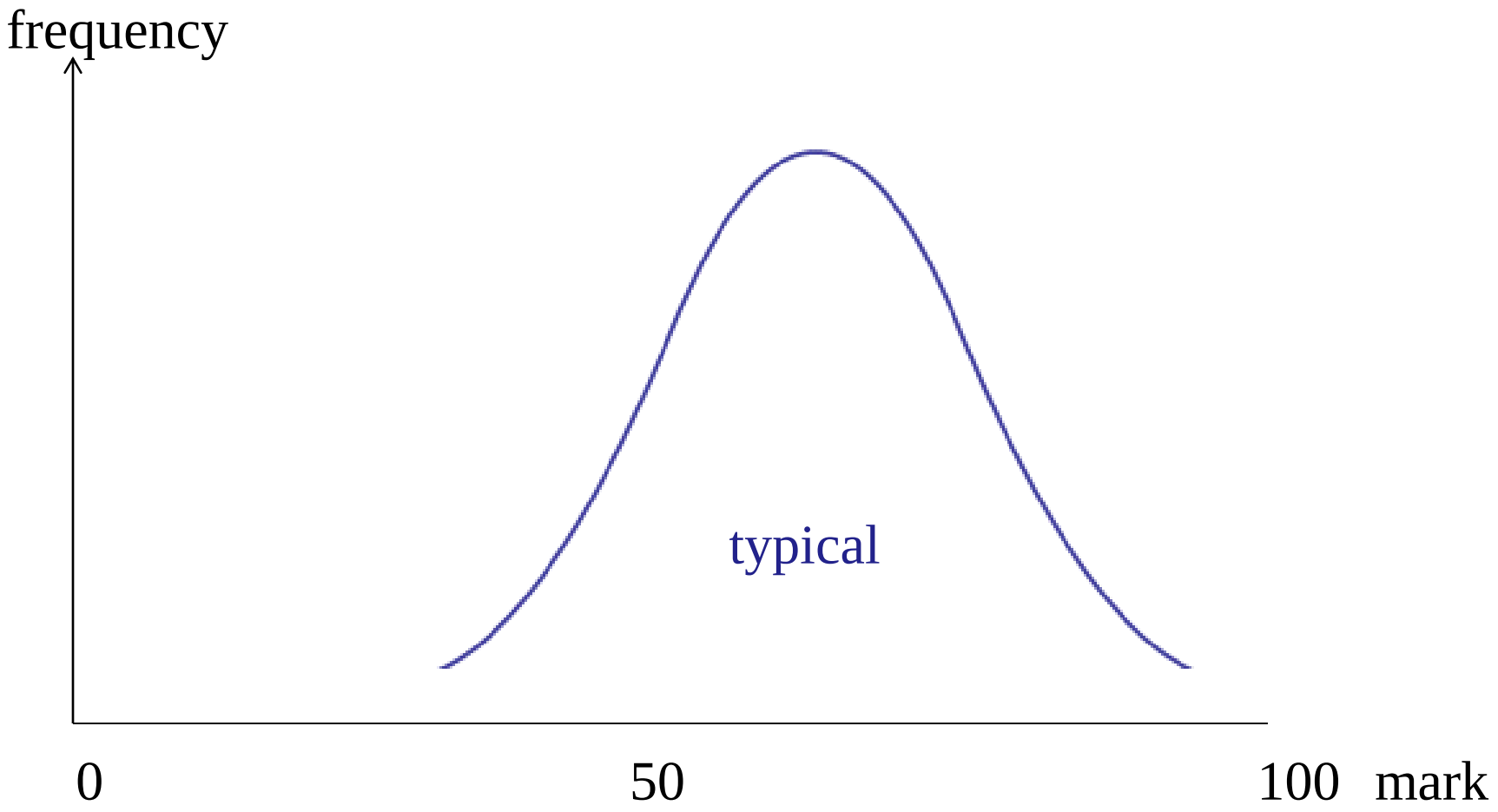




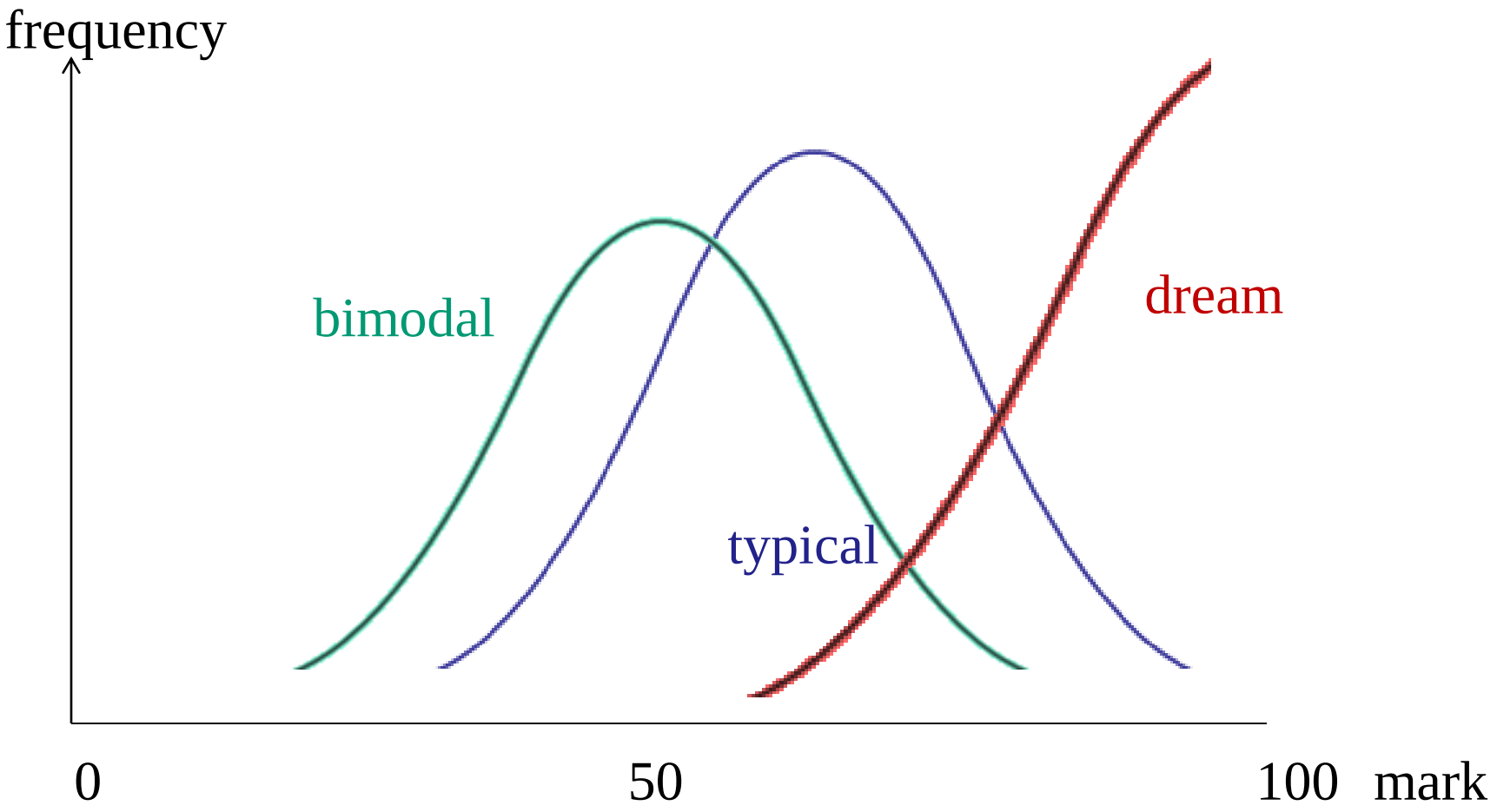
Final Grade Assignment

- No fixed breakpoints (e.g., 50% does not necessarily mean pass)
- Impossible to predict your letter grade (A, B, C, …) until everybody's marks are in (including the last midterm marks)
- I do not necessarily follow a particular curve.
- **Marks reflect mastery of course material (instructor's judgment)**

Mark distribution



Mark distribution



Mark distribution

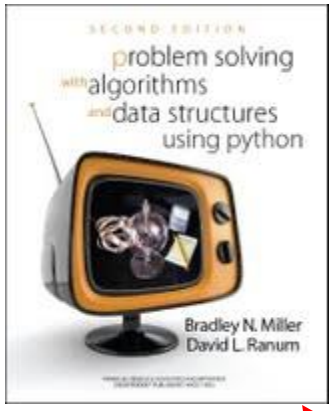
frequency

I want you
to help me make it happen

Practice, practice
Participate
Practice, practice

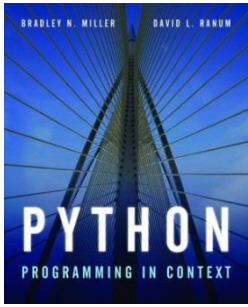
0 50 100 mark

No Imposed Textbook



This semester available on eclass

- Problem Solving with Algorithms and Data Structures using Python
- By Brad Miller and David Ranum
- Any text that covers the list of topics is recommended (there are many on-line tutorials)
- Other instructors in the past have used another book (Python programming in context) by the same authors



Slides

- The instruction is based mostly on slides, online videos and lab exercises
- Slides are available in PDF off the website. Each instructor may have different slides.
- Please (e)print them before the lecture and bring them with you to class to take notes
- Slides used in class could be slightly updated with examples (but not necessarily put on-line)

Labs

- Labs start next week (January 10th, 2022)
- All guidelines and news are on the web
  - Please read carefully to avoid unpleasant surprises
- Please check the News Forum often

Missing Components

- The detailed policy is described on the web
- If you have to miss something (e.g., a lab) please make prior arrangements
- If you missed something unexpectedly then you can apply for an EA
  - Not automatic, need documentation
  - Tight timeline
  - Not given for some components
  - No make-up exam ➔ weight transfer

Exams

- All three midterms will be administered on-line
- A midterm is one hour long
- You will be able to take the midterm anytime during a 24h period
- Some students will be selected for a short live verification online oral exam.
  If the result is above a threshold, the midterm grade is maintained. If it is below, then it will be used to average the midterm final mark.

Collaboration

- Collaboration allowed on:
  - Challenge problems
  - Assignments
  - Lab exercises

- Collaboration needs to be acknowledged explicitly in the work you hand in:
  - // I collaborated with P. Brosnan on this problem. Method hook-up() in
  - // class C007 was written jointly with M. Yeoh.
- Each student must be able to explain everything handed in

Submissions are individual. Each student is responsible for his/her submission.

Collaborations

- Absolutely NO collaboration on Midterms

Not just for midterms, but any assignment, lab exercise etc.

Plagiarism.

Work submitted by a student that is the work of another student or any other person, is considered plagiarism. Cases of plagiarism are immediately referred to the Dean, who determines what course of action is appropriate.

- Violations of this policy will be forwarded to the Dean's office
- A long, unpleasant process
- Permanent ramifications

Collaborations

Example of unacceptable collaboration:
1. Fred has a problem with his code
2. Fred asks Barney for help
3. Barney provides Fred with code
4. Fred submits Barney's code (all or part)

* Both Fred AND Barney will get in trouble

Example of acceptable collaboration:
1. Fred has a problem with his code
2. Fred asks Barney for help
3. Barney explains the issue to Fred
4. Fred understands, fixes, and submits his own code

* Neither Fred or Barney is in trouble

Code Of Student Behaviour

- …is available on the web
- Please peruse it carefully
- Contact me if in doubt

Course Schedule

(Tentative, subject to changes)

There are 14 weeks from January 5th to April 8th

Attendance is HIGHLY recommended
There are weekly exercises to be done in the lab.

Midterm 1 **week 6 (Monday February 14th 2022)**

Midterm 2 **week 10 (Monday March 14th 2022)**

Midterm 3 **week 14 (Friday April 8th 2022)**

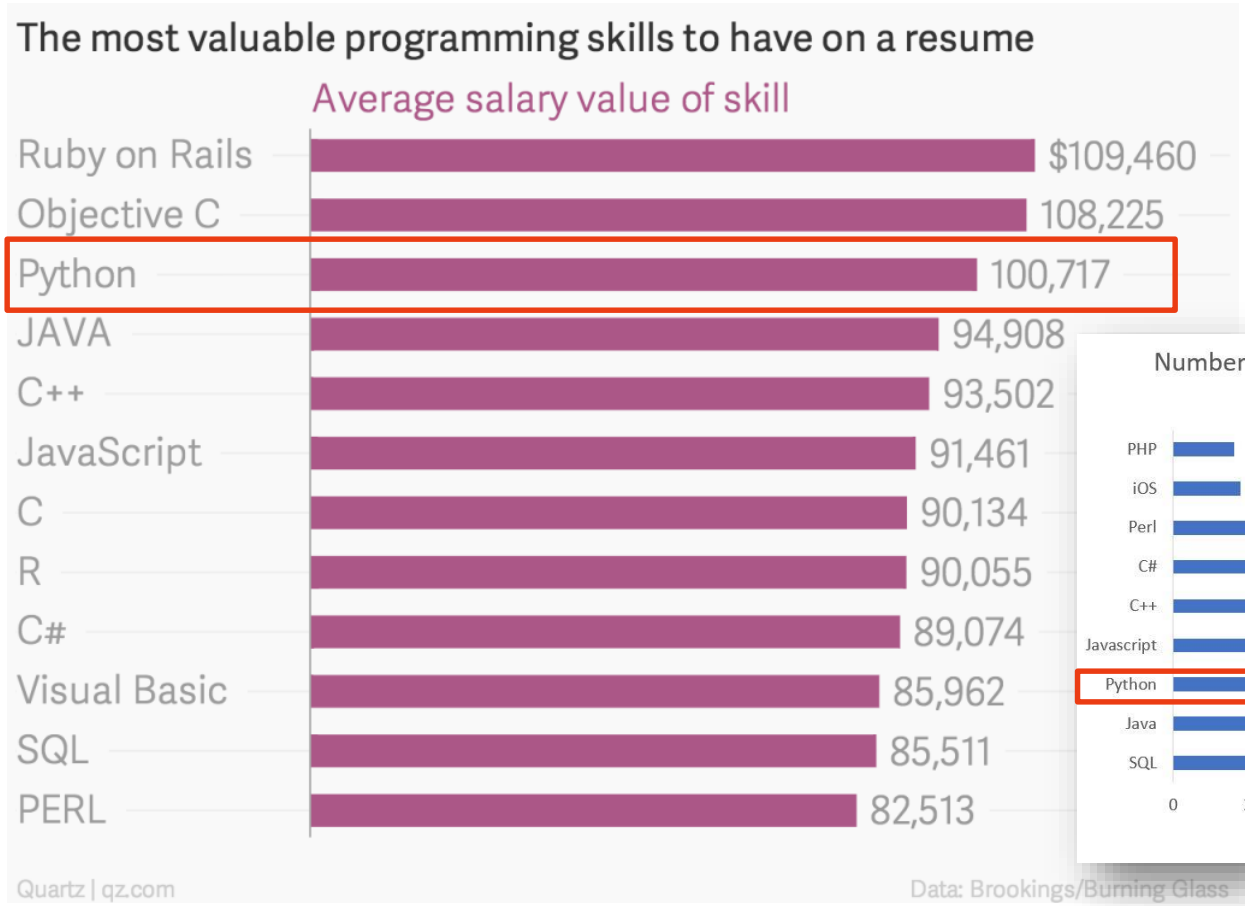
No class from February 21st to 25th (Reading week)

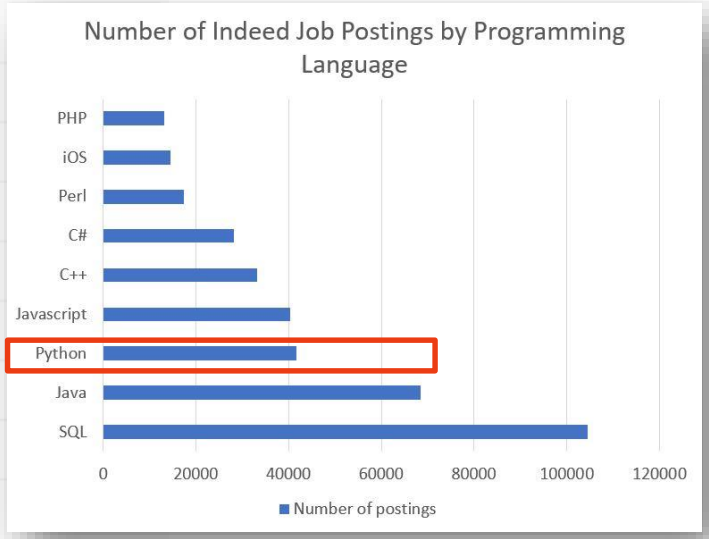
General Schedule per week

Week 1	Week 5	Week 10	Week 14
W: Syllabus + Intro	M: Stack	M: Search	M: Revision
F: Python 101	W: Handling Exceptions	W: Sorting	W: **Midterm 3**
Week 2	F: Handling Exceptions	F: Sorting	
M: Python 101	Week 6	Week 11: **Midterm 2**	
W: Enciphering	M: Handling Exceptions	M: Sorting	
F: Enciphering/Hangman	W: Queue	W: Sorting	
Week 3	F: Queue	F: Sorting	
M: Hangman	Week 7: **Midterm 1**	Week 12	
W: Algorithm Analysis	M: Reverse Polish Notation	M: Trees	
F: Algorithm Analysis	W: Reverse Polish Notation	W: Trees	
Week 4	F: Linked Lists	F: Trees	
M: Algorithm Analysis	Week 8: **Reading Week**	Week 13	
W: ADT	Week 9	M: Binary search trees	
F: Stack	M: Linked Lists	W: Hash tables	
	W: Doubly-Linked Lists	F: Hash tables	
	F: Recursion		

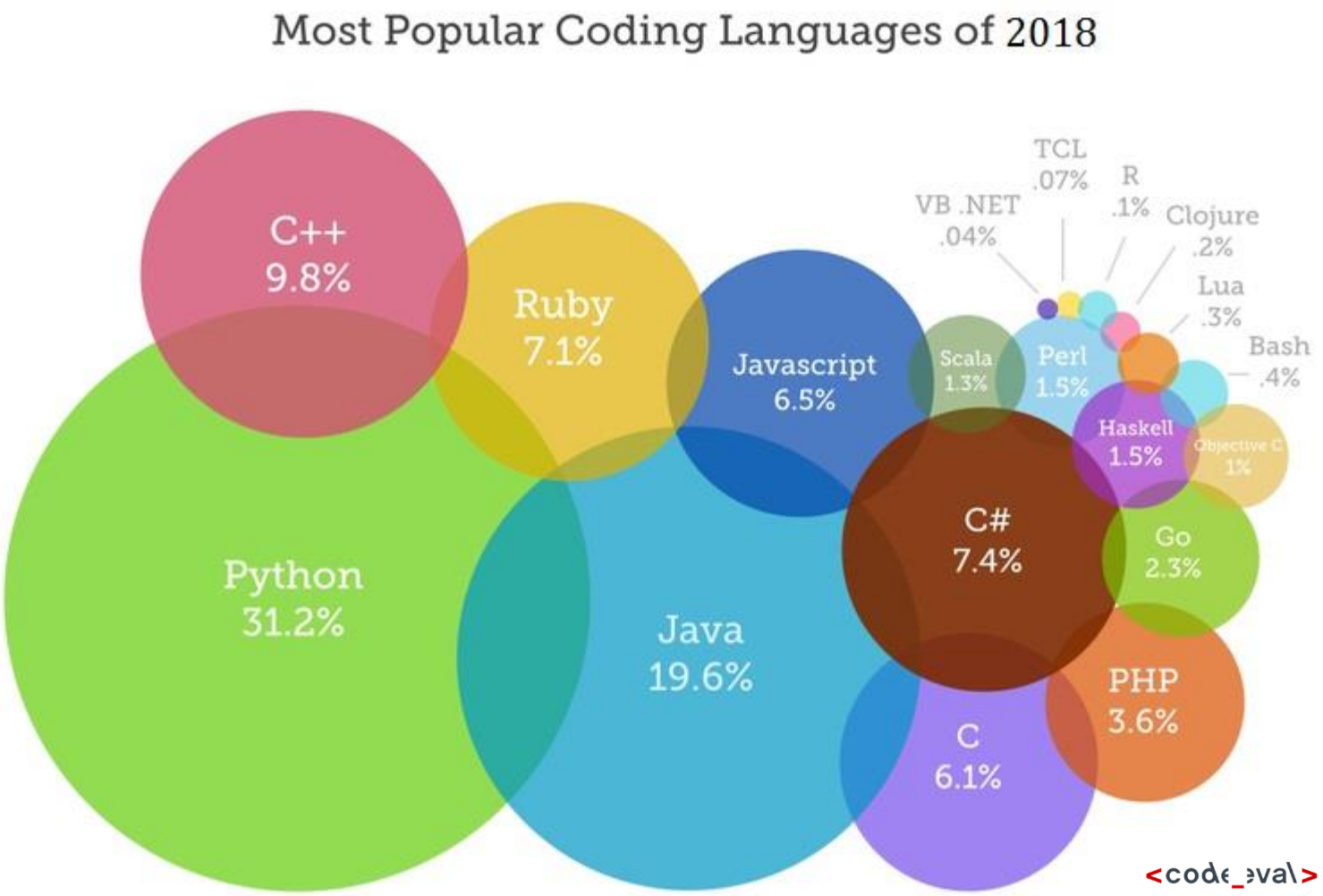
Why Python

About 25 years old; became lately very popular; easy language and easy to learn.

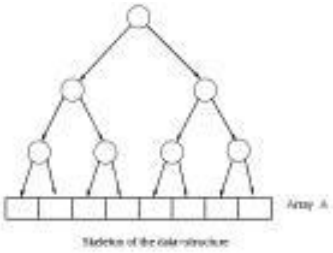


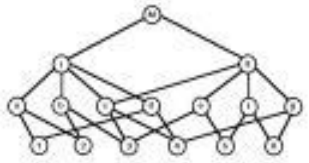


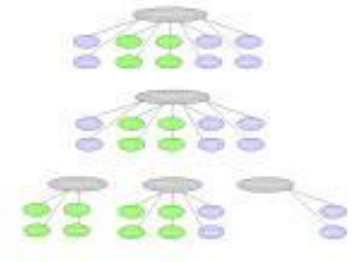
Why Python

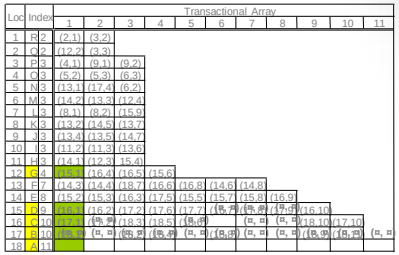


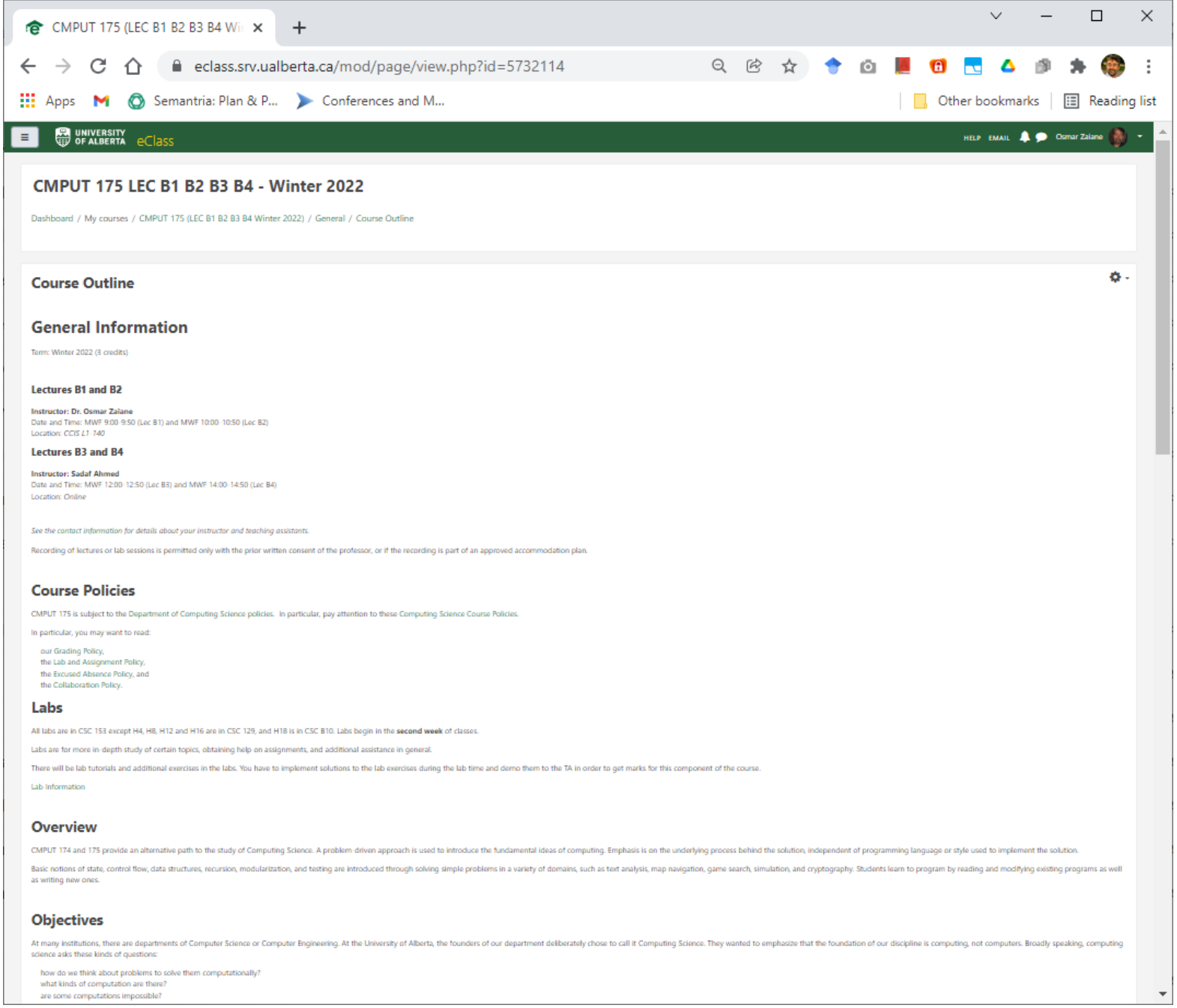
Questions



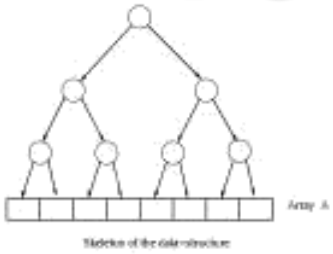


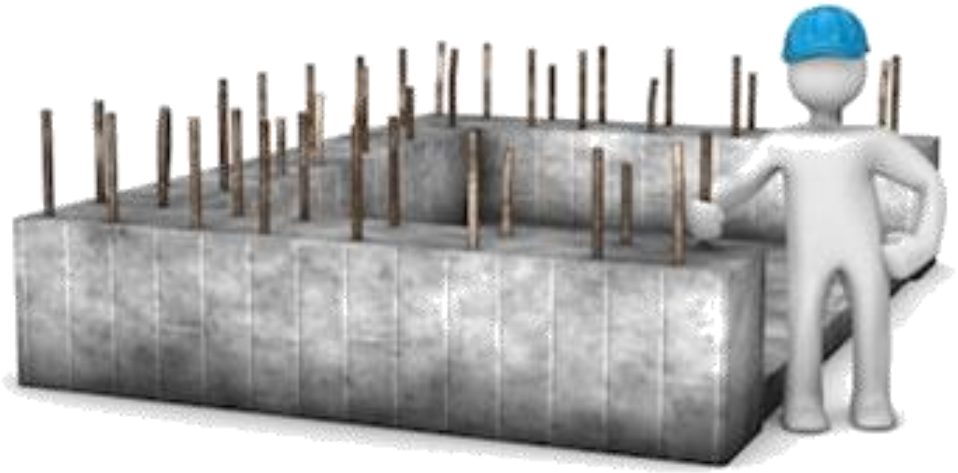


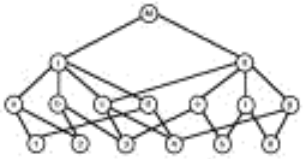


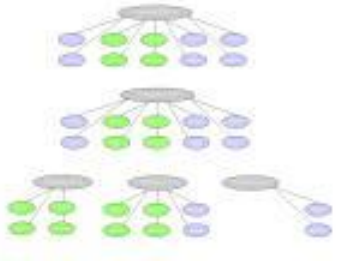

CMPUT 175 (LEC B1 B2 B3 B4 Wi
eclass.srv.ualberta.ca/mod/page/view.php?id=5732114
Apps
Semantria: Plan & P...
Conferences and M...
Other bookmarks
Reading list
UNIVERSITY OF ALBERTA eClass
HELP EMAIL Osmar Zaiane
CMPUT 175 LEC B1 B2 B3 B4 - Winter 2022
Dashboard / My courses / CMPUT 175 (LEC B1 B2 B3 B4 Winter 2022) / General / Course Outline
Course Outline
General Information
Term: Winter 2022 (3 credits)
Lectures B1 and B2
Instructor: Dr. Osmar Zaiane
Date and Time: MWF 9:00-9:50 (Lec B1) and MWF 10:00-10:50 (Lec B2)
Location: CCIS L1-140
Lectures B3 and B4
Instructor: Sadaf Ahmed
Date and Time: MWF 12:00-12:50 (Lec B3) and MWF 14:00-14:50 (Lec B4)
Location: Online
See the contact information for details about your instructor and teaching assistants.
Recording of lectures or lab sessions is permitted only with the prior written consent of the professor, or if the recording is part of an approved accommodation plan.
Course Policies
CMPUT 175 is subject to the Department of Computing Science policies. In particular, pay attention to these Computing Science Course Policies.
In particular, you may want to read:
our Grading Policy,
the Lab and Assignment Policy,
the Excused Absence Policy, and
the Collaboration Policy.
Labs
All labs are in CSC 153 except H4, H8, H12 and H16 are in CSC 129, and H18 is in CSC B10. Labs begin in the second week of classes.
Labs are for more in-depth study of certain topics, obtaining help on assignments, and additional assistance in general.
There will be lab tutorials and additional exercises in the labs. You have to implement solutions to the lab exercises during the lab time and demo them to the TA in order to get marks for this component of the course.
Lab Information
Overview
CMPUT 174 and 175 provide an alternative path to the study of Computing Science. A problem driven approach is used to introduce the fundamental ideas of computing. Emphasis is on the underlying process behind the solution, independent of programming language or style used to implement the solution.
Basic notions of state, control flow, data structures, recursion, modularization, and testing are introduced through solving simple problems in a variety of domains, such as text analysis, map navigation, game search, simulation, and cryptography. Students learn to program by reading and modifying existing programs as well as writing new ones.
Objectives
At many institutions, there are departments of Computer Science or Computer Engineering. At the University of Alberta, the founders of our department deliberately chose to call it Computing Science. They wanted to emphasize that the foundation of our discipline is computing, not computers. Broadly speaking, computing science asks these kinds of questions:
how do we think about problems to solve them computationally?
what kinds of computation are there?
are some computations impossible?

Introduction to Foundations of Computing









Data Structures:

We will build new tools for our toolbox