COURSE OVERVIEW

PRINCIPLES OF INSTRUCTION

Can Kultur, CMU

Summer - 2, 2015

Introductions

- Different levels
- Different fields
- Different perspectives

Why Are We Here?

- **Curiosity:** find out about computing technology and its many effects on society.
- **Professional development:** computing skills can make you more successful at work.
- **Academic requirement:** a computing course is required for your major. Why?
- **Intellectual growth:** computing changes how we think about problems. You can learn to think like a computer scientist.

Computation

- **Computer science** is the study of what can be computed and how to compute it:
  - **Computation**: Performance of a sequence of simple, well-defined steps that lead to the solution of a problem
  - **A computer:** Performs steps and remembers the results of those steps

What Kind of a Discipline is CS?

- **Science**: focuses on abstract, artificial things in a virtual world.
- **Engineering**: Building complex things by using techniques to manage complexity
- **Liberal arts**: Strong connections to traditional liberal arts of grammar, rhetoric, logic, arithmetic, geometry, music

High-level Goals of the Course

- With computational thinking you will be able to
  - **Identify problems** that are amenable to computation and express computations to find a solution
  - Understand the **power** and **limitations** of computational tools and techniques
  - **Ask new questions** that were not thought of or dared to ask because of scale, easily addressed computationally

Skills to Be Gained

- Systematic **problem solving**, applying **abstractions** as needed
- Reading, writing, and debugging small to medium-sized **programs** using the language Python
- Familiarity with **computational concepts** underlying pervasive technologies
- Familiarity with computational **vocabulary**

In their capacity as a tool, computers will be but a ripple on the surface of our culture. In their capacity as intellectual challenge, they are without precedent in the history of mankind.

Edsger Dijkstra,
1972 Turing Award Lecture

Course Organization

- Lectures (Mon/Tue/Wed/Thu/Fri 9:00 to 10:20)

  Instructor: Can Kultur (similar to John)

  Office: 6007 or 6009

- 2 recitation sections

  Teaching Assistants (TAs) to help you!

  - Hayley Zhang (Section U)
  - Clare Isaacson (Section E)
  - Joey Fernau

Resources

- Course web page:
  https://www.cs.cmu.edu/~15110-n15/

piazza: course message board
will be checked daily at certain times.

https://piazza.com/cmu/summer2015/15110

Office Hours

- Instructor: After the lecture hour
- TAs:
  - 4:30-5:20 on non-lab days (in Labs)
  - 5:30-6:30 on Lab days (to be announced)
- Schedule will be finalized and published on course web page.

Textbooks

- There is no designated textbook.
- See the course web page for recommended books.

Assignments

- **Labs:** do in recitation; hand in using Autolab.
- **Written problem sets (PS)**:
  - Handed in on paper in class (9:00). (See course web page for due dates)
- **Programming assignments (PA)**:
  - Due 11:59 PM on the scheduled day
  - Handed in using Autolab

Late Policy

- Assignments must be handed in on time.
  - Late assignments receive a grade of 0.

- We will drop **1 written assignment** and **1 programming assignment** without penalty (except where noted)

- We will drop **2 labs** without any penalty.

Exams

- You must take all the exams, at the time they are given.
- No makeups except for extreme circumstances (**major** illness, death in immediate family, or a university-sanctioned event with documentation and prior permission)
  - 2 Lab Exams (done on the computer)
  - 2 Written Exams
  - Final Exam

Grading

Homework Assignments:	30%
Lab Participation:	5%
2 Lab Exams:	10% (5% each)
2 Written Exams:	30% (15% each)
Final Exam:	25%

Expected Effort

- We assume that you have no prior knowledge in computing. Do you?
- Need to study daily.
- Summer schedule is tight.

Academic Integrity Policy

- University Policy on Cheating and Plagiarism
- Academic Integrity Form (see web page).
  - Print it out. **Read it**. **Sign it**.
  - **Bring it to your lecture (latest July 2).**

Getting Started With Computational Thinking

Computation

- A computer does 2 things
  - Performs instructions
  - Remembers their results

- Historically computation speed was limited by the human brain and the ability to record results by the human hand. But modern computers relieved us from those constraints.

At the very basics

- This is a series of words that is called a 'sentence'.
- A 'word' is a series of letters.
- What makes a letter different from another?

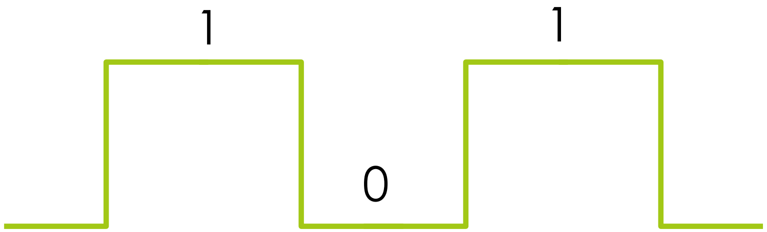
- Agreed forms are used to build words then sentences then paragraphs ...
  - Simple symbols → combinations → complexity and higher level

At the very basics

- On – Off
- Yes – No
- True – False
- Correct – Wrong

At the very basics

- On – Off
- 1 – 0 ← Electric pulses (High - Low)



At the very basics

- Ligth is on
- Light is off

- First light is on
- First light is off
- Second light is on
- Second light is off

How many options

2 x 2 x 2 x 2 = 16 options

How many options

2x2x2x2 = 16 options

Down (Off) → 0   Up (On) → 1

0 0 0 0	0 1 0 0	1 0 0 0	1 1 0 0
0 0 0 1	0 1 0 1	1 0 0 1	1 1 0 1
0 0 1 0	0 1 1 0	1 0 1 0	1 1 1 0
0 0 1 1	0 1 1 1	1 0 1 1	1 1 1 1

Let's say 'HI' to digital world

237 =

2	3	7
100s	10s	1s

2 hundreds + 3 tens + 7 ones

1	0	1	1
8s	4s	2s	1s

1 Eight + 0 Four + 1 Two + 1 One = **11**

Say ‘HI’ to digital world

0	1	0	0	1	0	0	0
128	64	32	16	8	4	2	1

72

0	1	0	0	1	0	0	1
128	64	32	16	8	4	2	1

73

H I

01000000	64	@
01000001	65	A
01000010	66	B
01000011	67	C
01000100	68	D
01000101	69	E
01000110	70	F
01000111	71	G
01001000	72	H
01001001	73	I
01001010	74	J

Mechanical Procedure

- Computers execute **mechanical procedures** -- procedures that can be followed without any thought.
- We need to give them **unambiguous instructions** such that when followed step by step the execution will finish and yield a result.
- An **algorithm** is a mechanical procedure that is guaranteed to eventually finish.

A Procedure Example from Real Life: Pizza Dough Recipe

- Combine the bread flour, sugar, yeast and salt in the bowl of a mixer.
- Start the mixer.
- Add water and 2 tablespoons of oil.
- Beat until the dough forms into a ball.
- If the dough is sticky, add additional flour and beat.
- If the dough is too dry, add additional water and beat.
- Otherwise, stop and knead.

Describing what to do

If I was a robot, can you describe me how to exit from the class

AND make me stop outside and hopefully bring me back ☺. **(Did you?)**

Alternative Algorithms

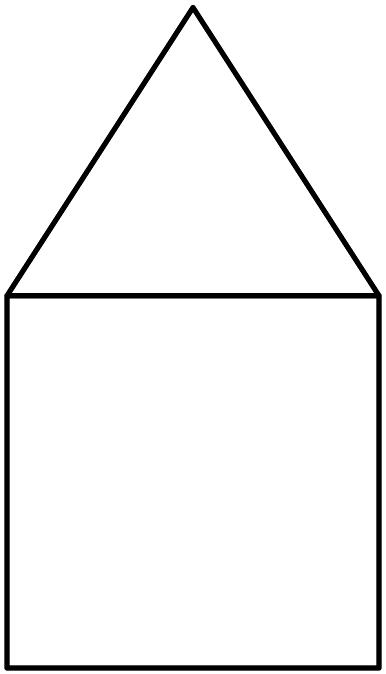
- What is the number of students in the class?
  - How can you count?
  - Can you count it in different ways?

Alternative Algorithms

1. Take a card having number 1 and stand up
2. Pair of with a student having a card in hand
3. Sum the numbers on your cards and write it as your new number.
4. One student sits the other goes back to step 2

Statements → Statement series

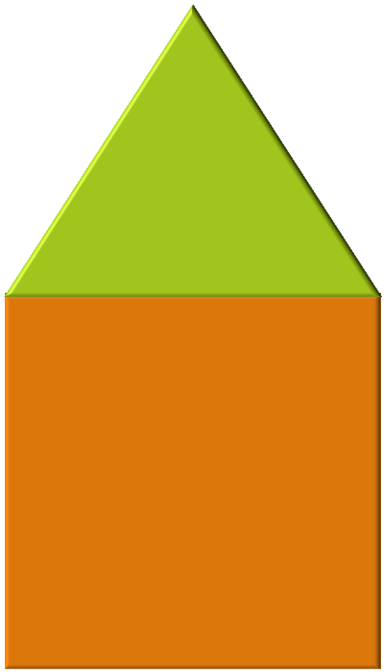
Can you describe how to draw this house (to the previous robot)?



Set of Statements → Efficiency

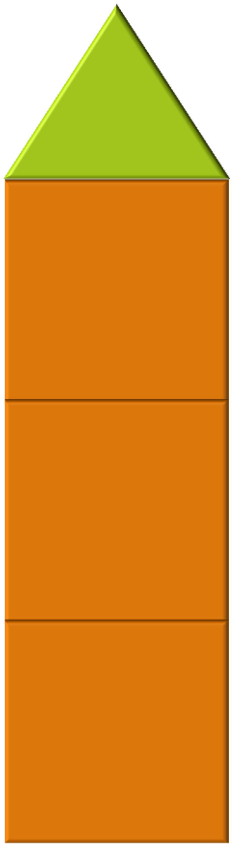
Using other concepts

Can you describe how to draw this house?



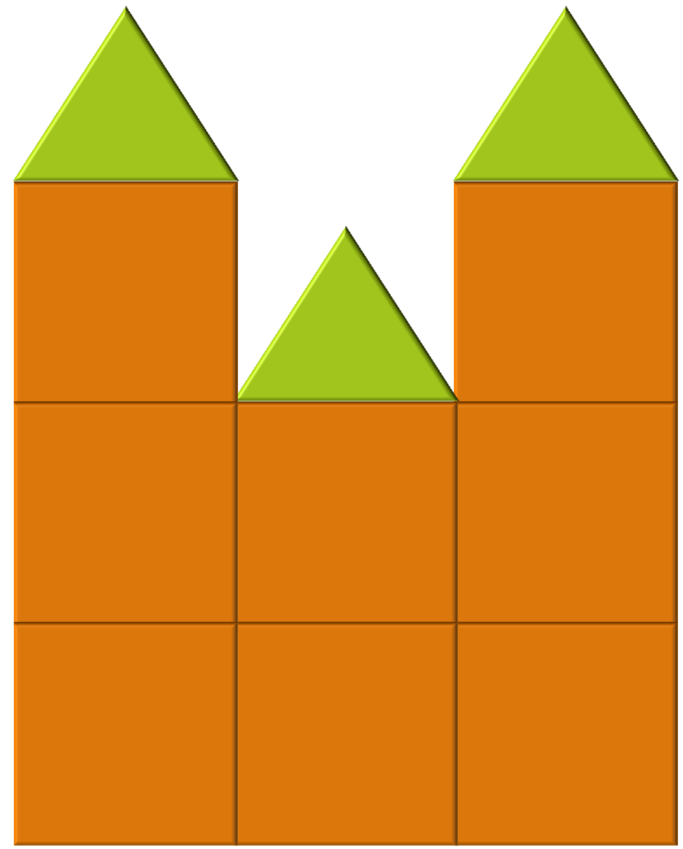
Set of Statements → Reusability

Can you describe how to draw this apartment?



Computational thinking

Helps us solve more complex and higher level problems



Remembering: Variables

Assume that you forget everything very quickly

An information is given to you

What can you do to use it later?

Remembering: Variables

Write it to a place **and put a label (or name it)**

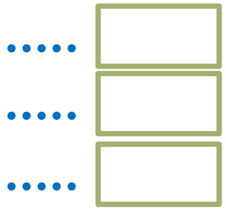
Name of box 1

Name of box 2

Name of box 3

A Mechanical Procedure Example (1)

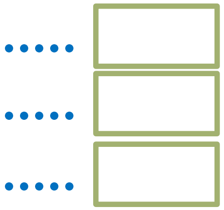
Procedure to calculate the average of numbers 1 .. 10.



1. **Input** integers **1** through **10**
2. Set **sum** to **0**
3. Set **current_number** to **1**
4. Set **sum** to **sum** + **current_number**
5. If **current_number** is **10** then jump to step 7
6. Otherwise increment **current_number** by **1** and jump to step 4
7. Set **average** to **sum** / **10**
8. **Output average**

A Mechanical Procedure Example (2)

By using **abstraction** we can calculate for any range 1 .. ***N***.



1. **Input** integers **1** through **N**
2. Set **sum** to **0**
3. Set **current_number** to **1**
4. Set **sum** to **sum** + **current_number**
5. If **current_number** is **N** then jump to step 7
6. Otherwise increment **current_number** by **1** and jump to step 4
7. Set **average** to **sum / N**
8. **Output** **average**

What to do in order to calculate for any range M ... N?

General Purpose Computing

- We could design a machine specifically intended for computing averages.

- Earlier computing machines were fixed program machines that were designed for specific calculations (fixed program computers).

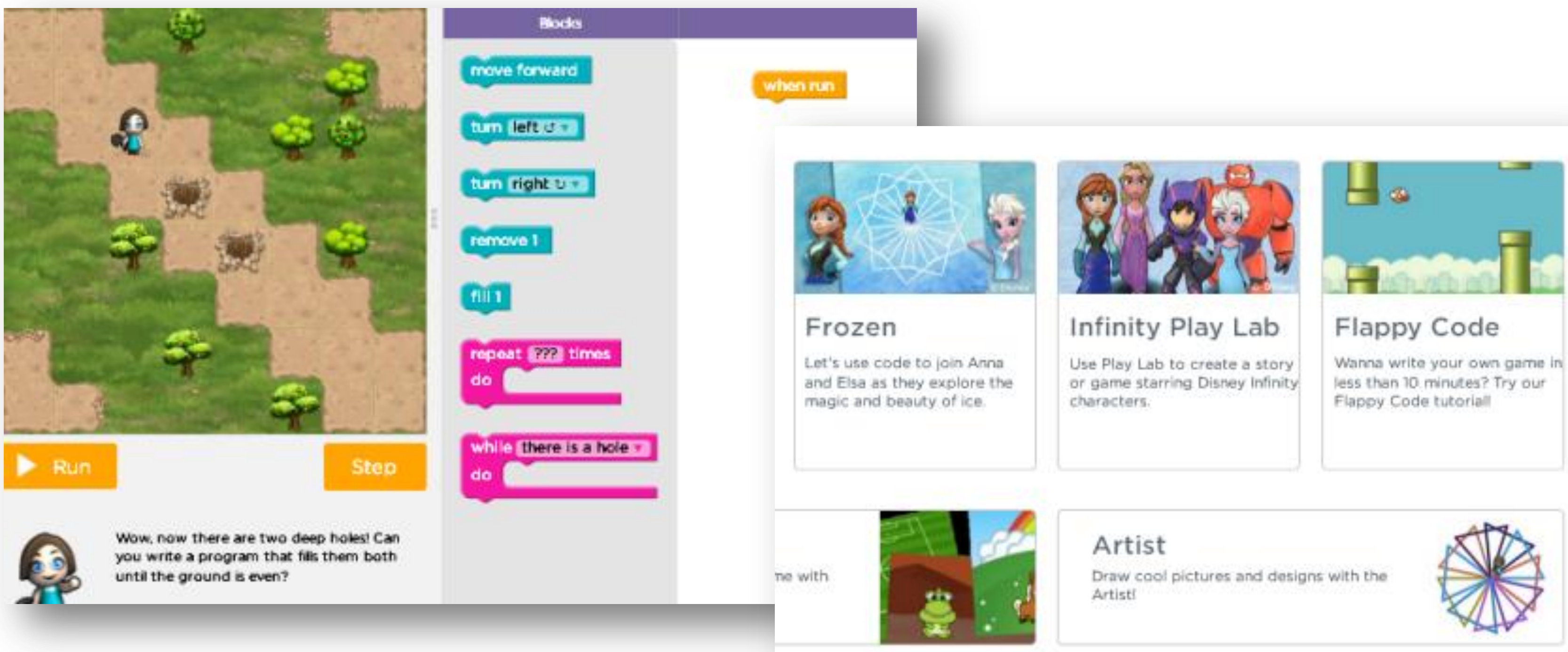
- Modern computers store sequences of instructions and execute them (stored-program computers).

Programs to describe mechanical procedures

- What if we have millions of steps to specify for a computer?
- We typically use higher-level programming languages to describe computations.

Today's Lab and First PA

Programming with Blockly → http://code.org/



Piazza & Online communication

- Email addresses are on the course web page

BUT as a teaching team we prefer **Piazza. Why?**

- Do not copy your source code in your posting to the piazza.
- Help each other but do not say the answer directly.

ASSIGNMENTS

- **Bookmark** course page: www.cs.cmu.edu/~15110-n15
  - Read the syllabus (home + schedule)
- **Check** Piazza and Autolab accounts
  - Upload your **photo** to Piazza
  - Reply to "**Introductions through a game**" topic
- Read and sign the **Academic Integrity Form** (7/2)
- Today's **Lab** → **1st PA** due to Tomorrow 11:59pm

Reading

- Blown to Bits
  Chapter 1 “Digital Explosion”.

Your Turn

- As pairs can you create an algorithm for yourselves in order to program your daily routine for this class (until the end of course)
  - Consider things to check
  - Consider potential needs (e.g. if you have a problem in ... what to do or how to act)

Thank you

CS IS FUN

PROGRAMMING IS FUN

HAVE FUN