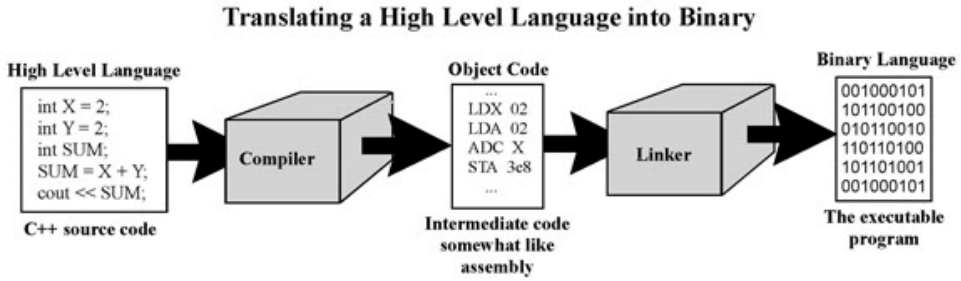
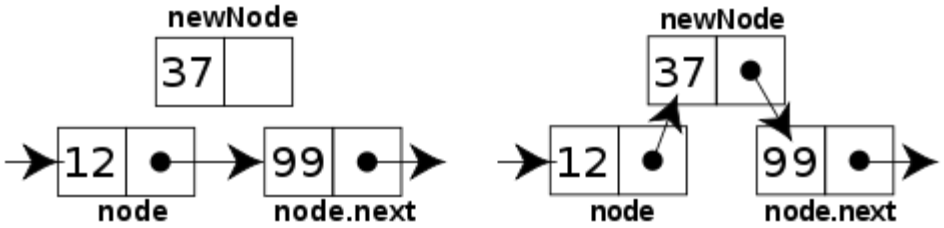
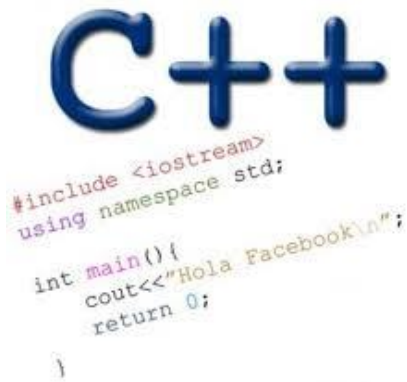


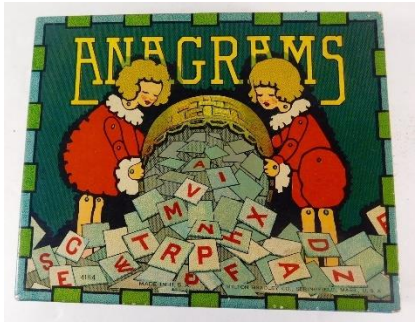


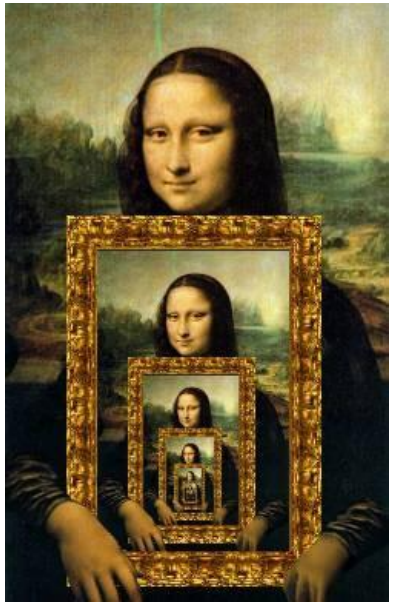
WELCOME TO CS 16!

Problem Solving with Computers-I



https://ucsb-cs16-wi17.github.io/



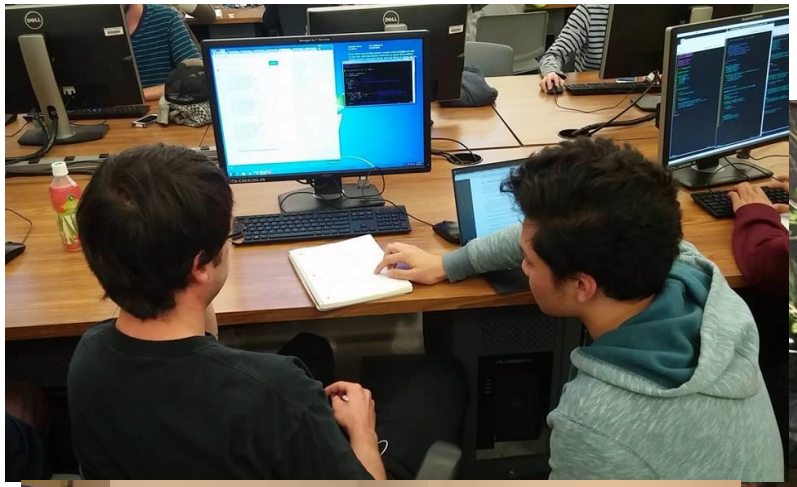


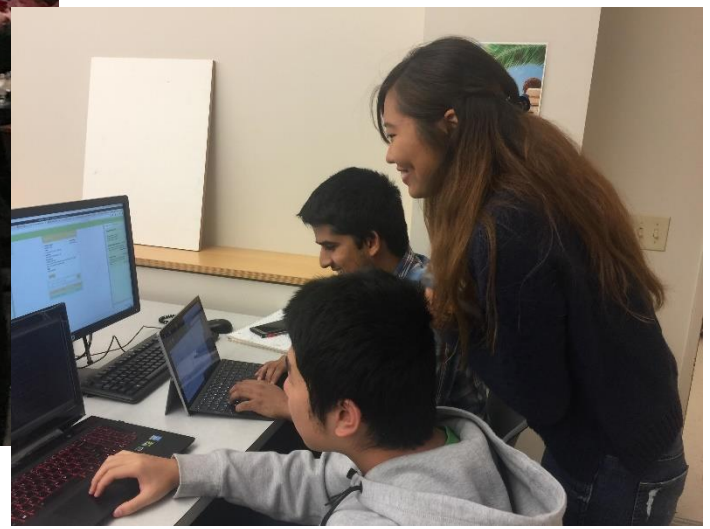
Instructor

- Diba Mirza (dimirza@cs.ucsb.edu)
  - PhD (Computer Engineering, UCSD)
  - New teaching faculty at the department of Computer Science, UCSB!
  - Before this: Teaching faculty at UCSD for three years

- Office: HFH 1155
- Office hours:
  - Mon, Wed: 10am – 11am
  - Tues: 5pm – 6pm
  - Or by appointment

Undergraduate tutors and TAs from my last class at UCSD





Our teaching staff and brand new tutor program !

Harshitha Murthy (TA)

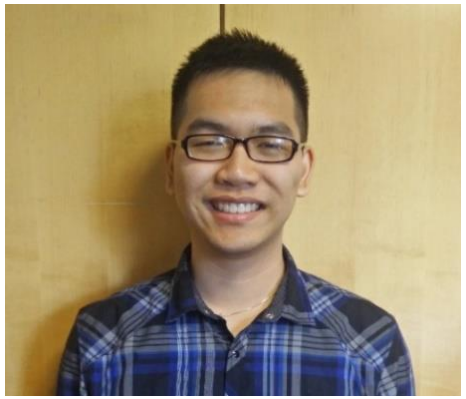
Lin Chai (TA)

Yan Kong (Reader)

Daria Rudneva (TA)

Angela Yung
(UG tutor)

Barbara Korycki
(UG tutor)



Jimmy Le
(UG tutor)

Sayali Kakade
(UG tutor)

Sean Shelton
(UG tutor)

Steven Fields
(UG tutor)

About this course

Why C++?

We will learn:

- A new programming language: C++
- Computer hardware from a programmer's perspective
- Abstractions used in programming and problem solving
- Tools and practices used by professional programmers : git, unix, test driven development (TDD)
- Big ideas that have shaped computing

Solving problems with computers…like a pro!

Clickers out – frequency AB

About you…

How far along are you in your undergraduate program?

A. Freshman
B. Sophomore
C. Junior
D. Senior

About you…

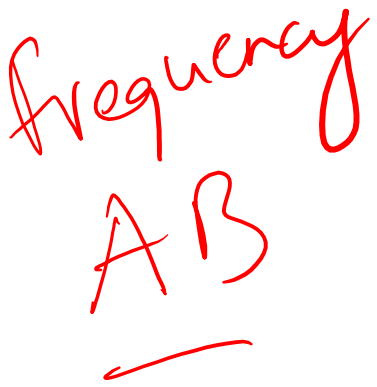
What is your major?

A. Computer Science
B. Computer Engineering
C. Other

About you…

What is your past programming experience?

A. Have never programmed.
B. Have programmed before “just for fun”
C. Have taken an introductory CS course
D. I have a lot of programming experience



About you…

What is your familiarity/confidence with programming in C++?

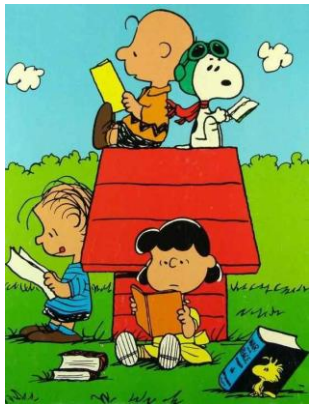
A. Know nothing or almost nothing about it.
B. Used it a little, beginner level.
C. Some expertise, lots of gaps though.
D. Lots of expertise, a few gaps.
E. Know too much; I have no life.

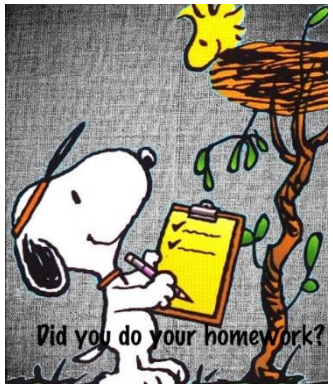
About you…

What is your familiarity/confidence with using version control with Subversion, Git or any other VCS?

A. Know nothing or almost nothing about it.
B. Used it a little, beginner level.
C. Some expertise, lots of gaps though.
D. Lots of expertise, a few gaps.
E. Know too much; I have no life.

Structure of the class





In class, learn by:

- Taking notes
- Discussing with your peers to identify gaps in your knowledge
- Exercise your meta-cognitive skills!

Before class:

Do the readings and homework!

In class:

- Submit your homework
- Follow class policy on the use of electronic devices

In sections, learn by

- Doing the programming assignments
- Working closely with your pair partner
- Brainstorming with our TAs and tutors!

Have you been in a class that used peer instruction before?

A. Yes
B. No
C. I’m not sure

Clickers, Peer Instruction, and PI Groups

- Find 1-2 students sitting near you. If you don't have any move.
- Introduce yourself.
- This is your initial PI group (at least for today)

iClickers: You must bring them

- Buy an iClicker at the Bookstore
- Register it on GauchoSpace by Friday (01/13)
- Bring your iclicker to class AND section

Assigned Reading from

- Problem Solving with C++, Walter Savitch, Edition 9

You must attend class and lab sections
You must prepare for class
You must participate in class

Course Logistics

- Grading
  - Class and section participation (iclickers): : 2%
  - Homeworks (due every lecture) : 13%
  - Lab (programming) Assignments(due weekly on Fridays) : 35%
  - Midterm exams: (two, 15% each) : 30%
  - Final exam : 20%

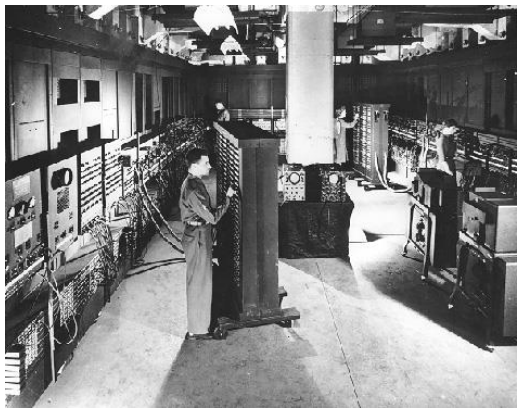
- Less than 75% iClicker response ≡ missing a class/section
- No makeups for exams. Make sure you have no scheduling conflicts with exams
- No LATE submissions unless you have a real emergency!

Assignment Calendar

Week	S	M	T	W	R	F	S
1	01/08	01/09 First day of classes	01/10 h01 assigned lab00 assigned lect01: Course overview, a gentle intro to C++ and git	01/11	01/12 h01 due 03:30pm h02 assigned lect02: C++ variables and data types, input-output and simple flow control	01/13 lab00 due 11:59pm lab01 assigned	01/14
2	01/15	01/16 Univ Holiday	01/17 h02 due 03:30pm h03 assigned lect03: How is data stored on a computer?	01/18	01/19 h03 due 03:30pm h04 assigned lect04: How are programs translated to a form understandable by a computer?	01/20 lab01 due 11:59pm lab02 assigned	01/21

- For more information, see our Assignment Calendar: https://ucsb-cs16-wi17.github.io/info/calendar/
- All sections will be in PHELPS 3252
- Open labs: CSIL in Harold Frank Hall
- The schedule for sections, office hours and open lab hours is available on our class Google Calendar: https://ucsb-cs16-wi17.github.io/info/schedule/

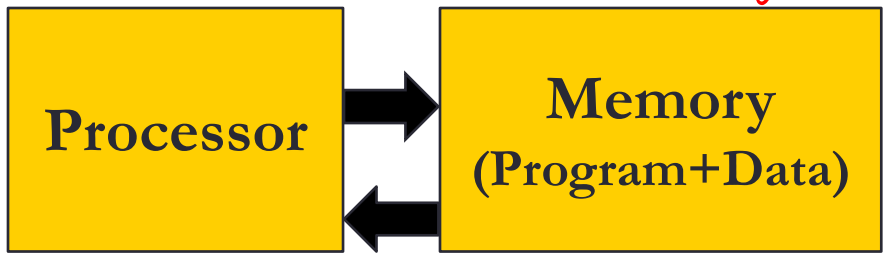
Know your computer – a brief historical perspective!



ENIAC: 1st Large Scale, General Purpose Electronic Computer

- Early computing machines used a "fixed program model"
- ENIAC was "reprogrammable": Took three weeks to set up a program
- Next generation of computers: Von Neumann architecture based on the "stored program" model (modern computers use this!)

Key aspects of the "stored program model"
- hardware / processor is designed to understand a set of simple instructions
- All programs are expressed in terms of these instructions & stored in memory along with data

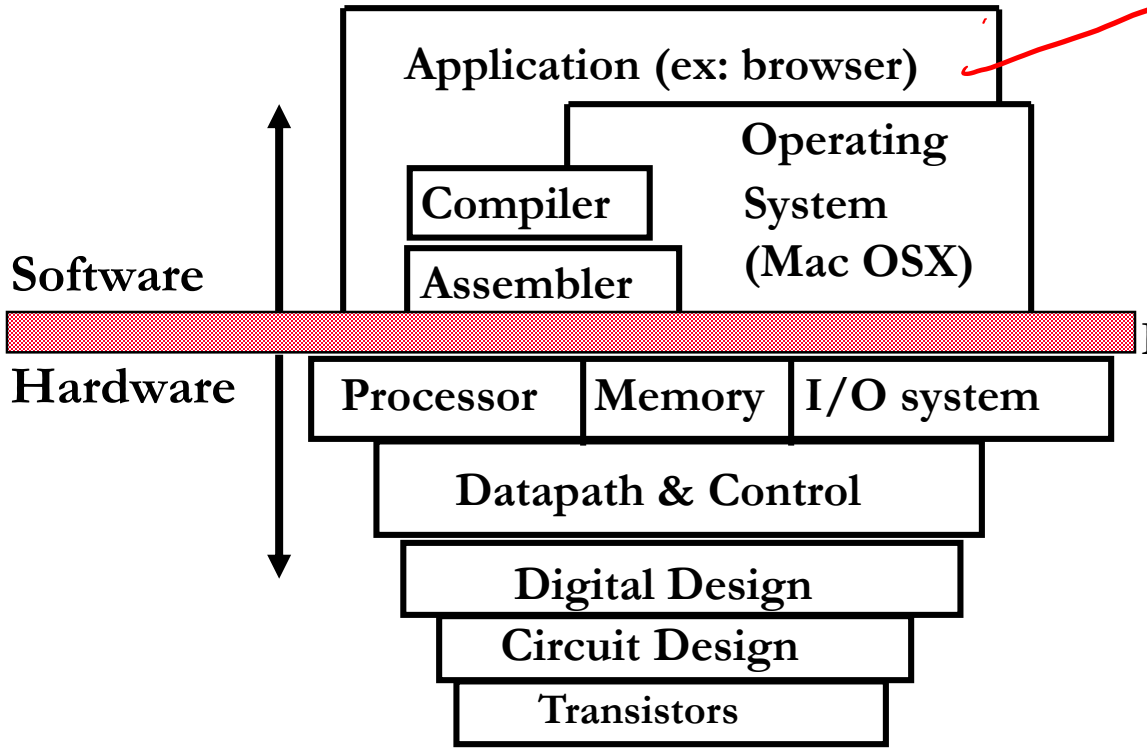


Stored Program Model

Five important components of a computer
1. Processor [circuit that runs programs]
2. Main memory [Place to store programs & data while they are being run]
3. Secondary memory [files stored here]
4. Input devices (like keyboard)
5. Output devices (like monitor)

The big idea is that many different programs can be written without changing the hardware

How do we handle complexity?




```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```

```
ldr   r0, [r2]
ldr   r1, [r2, #4]
str   r1, [r2]
str   r0, [r2, #4]
```

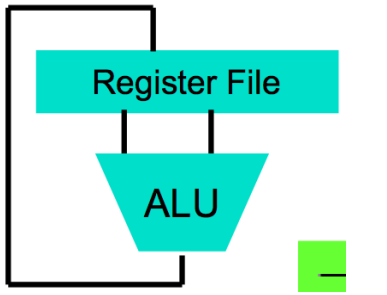
High level language Like C++

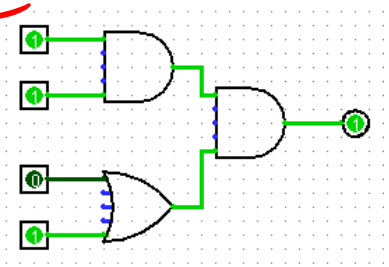
Compiler translates to machine code

```
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111
```

Machine code is in 1's & 0's (binary)

Machine code is the program expressed in terms of very simple instructions that the machine (processor) can understand!





- Big idea: Coordination of many *levels of abstraction*

Lab 00: Must be done individually

Key learning goals:

- Connect remotely to the CSIL unix servers (csil-0X.cs.ucsb.edu)
- Get familiarized with basic UNIX commands
- Create your first C++ program, compile and run it
- Get started with github

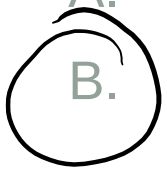
LIVE DEMO

Academic Integrity Scenario

You finish your CS 16 programming assignment and submit it. Your friend is really struggling and asks you to send him/her your code. Since you have already submitted the assignment, you do so. Is this OK for you to do?

A. Yes, this is fine

B. No, this is a violation of the Academic Integrity Policy for CS 16



Academic Integrity Scenario

You finish your CS 16 programming assignment and submit it. You're hoping to get an internship this summer, so you post your code in a public github repo so that potential employers can see it. Is this OK to do?

A. Yes, this is fine

B. No, this is a violation of the Academic Integrity Policy for CS 16

Integrity Guidelines

Basic rules

- Do not look at or copy other people's code and do not share your code with others (other than your partner). Period.
- "Other people" includes what you can find/share on the internet.
- Read the Integrity Statement carefully. Ask if you have questions.
- Read and sign the Academic Integrity form: https://goo.gl/forms/SkFNl9r7fwzeB1PD2

Integrity

- You will be tested on your ability to understand and write code in C++ in this class (and invariably during interviews)
- Cheaters will likely get "caught" during the exam because exams, for the most part, make your grade in this class.
- Why else shouldn't you cheat?
  - Its unethical
  - Its unfair to students who do the work legitimately
  - Hurts the reputation of the UCSB's degree programs

Next time

- C++ variables and data types
- Simple flow control