WELCOME TO CS 16!

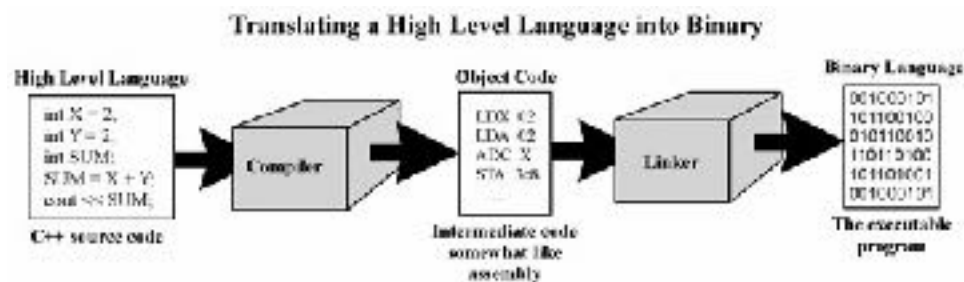Problem Solving with Computers-I

About this course

You will learn :

- **C++** (really the C part of C++) - why?
- Understand **what goes on under the hood** of C++ programs - why?
- Learn how to **debug** better
- **Solve fun problems :)**

iClickers: You must bring them

- Buy an iClicker at the Bookstore
- Register it on GauchoSpace
- Bring your iclicker to class

Assigned Reading from

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

Clickers out

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 UNIX command line
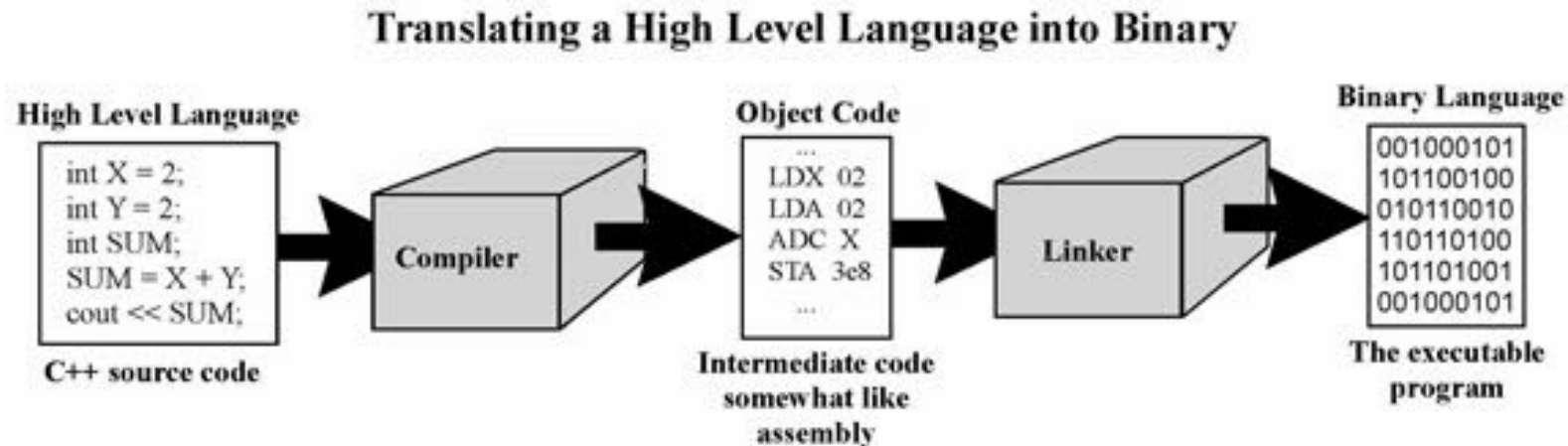
A. Know nothing or almost nothing about it.

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.

Abstracted view of a computer:
Five hardware components

- Input devices
- Output devices
- Processor
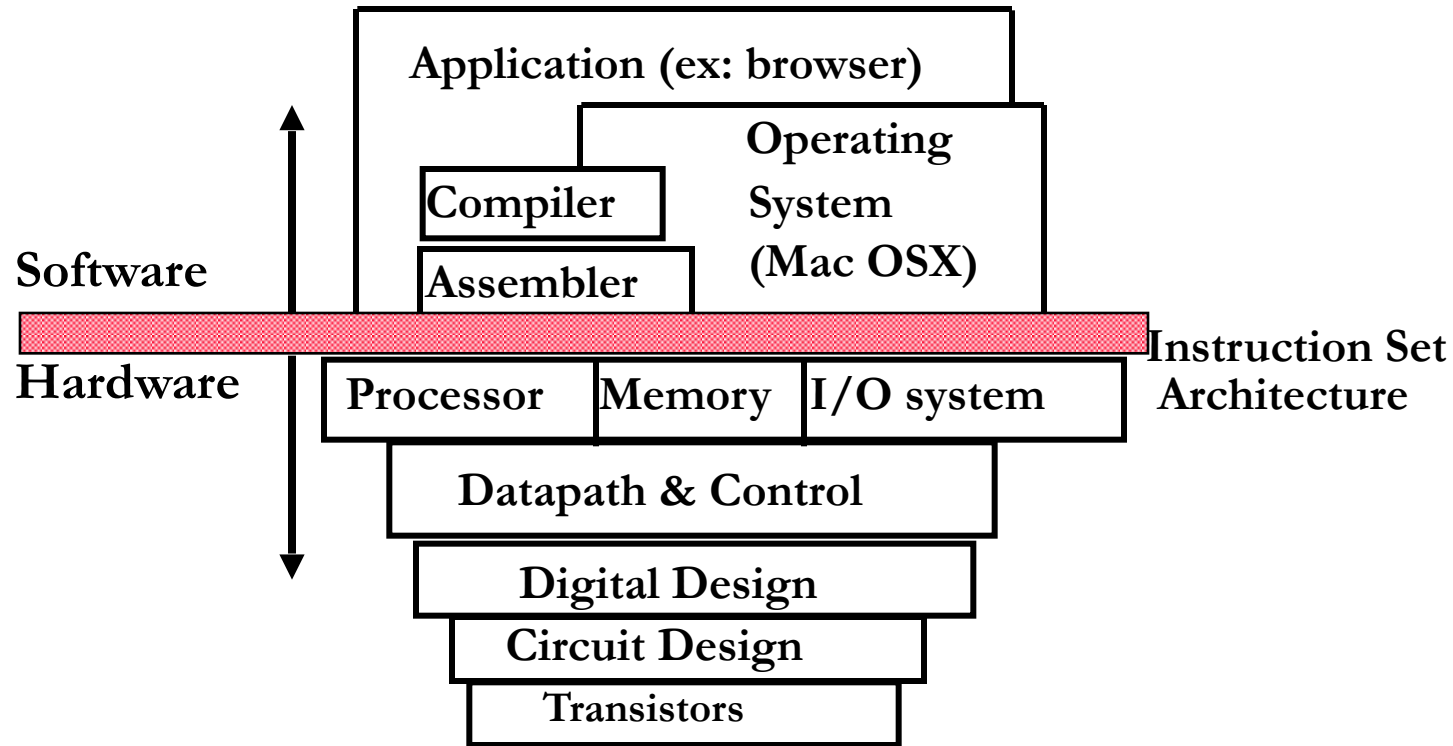- Main memory
- Secondary memory

The different stages of writing C++ code

- Editing – basically entering code in a text file
- Compiling – converting your code in a form the processor can understand (using another program called a compiler)
- Running – executing the binary version of your program on the processor

Translating a High Level Language into Binary

High Level Language

```
int X = 2;
int Y = 2;
int SUM;
SUM = X + Y;
cout << SUM;
```
C++ source code

→ **Compiler** →

Object Code

```
...
LDX  02
LDA  02
ADC  X
STA  3c8
...
```
Intermediate code somewhat like assembly

→ **Linker** →

Binary Language

```
001000101
101100100
010110010
110110100
101101001
001000101
```
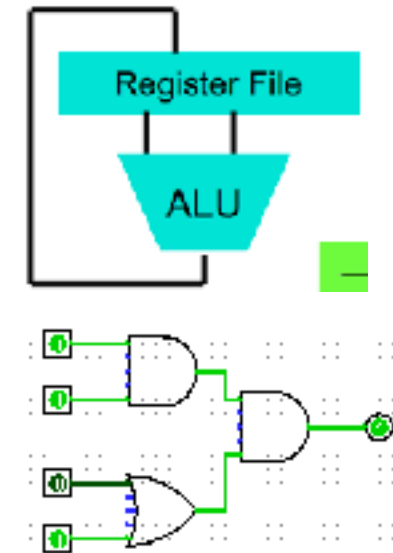The executable program

LIVE DEMO of writing a simple C++ program

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]
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
```

Application (ex: browser)

Compiler

Operating System (Mac OSX)

Assembler

Software

Hardware

Processor   Memory   I/O system

Instruction Set Architecture

Datapath & Control

Digital Design

Circuit Design

Transistors

Register File

ALU

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

Q: Which of the following converts a high level language to machine language

    A. Main Memory
    B. Secondary Memory
    C. Processor
    D. Compiler
    E. Operating System

Lab 00: Must be done individually

Before coming to the lab:

• Read the lab00 writeup

• Get a CoE account if you don't have one already.

• You can check if you have a working account by trying to remotely log into csil-02.cs.ucsb.edu

Key learning goals of lab00:

• 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

LIVE DEMO

Basic structure of a C++ program

// name of the program as a comment: hello.cpp

// Everything after the double slash is a comment

```cpp
#include <iostream>
// Include the "modules" needed for basic input output
using namespace std; // using the Standard C++ library

int main(){
    //Write code here
 return 0;
}
```