THE UNIVERSITY OF ALABAMA AT BIRMINGHAM

Welcome to CS330

Computer Organization and Assembly Language Programming

Spring 2022

Lab 1

Today's Agenda

- Introductions
- Success Heuristics
- Course (Lab) Overview
- Environment Set-up
- Typical Workflow

Introductions

Dr Mahmut Unan

unan@uab.edu

Office Hours (UH 4159)

W: 2:00 – 5:00

Zoom Link:
https://uab.zoom.us/j/9475934695

TA

Dylan Calvin

dylcal13@uab.edu

Office Hours (UH1009)

T: 11:00-12:30

H: 11:00-12:30

TA

John Bedingfield

bedingjd@uab.edu

Office Hours (UH1009)
M: 8:00 – 10:00

W: 1:10 – 2:00

Tutor

Ian Toy

iantoy@uab.edu

Office Hours (UH1008)
T: 5:00 – 6:30

F: 5:00 – 6:30

Success Heuristics
or How to be successful in CS330 and have the most fun

- Start homework / labs as soon as possible
- Begin with design (this is hard since we all want to start with code)
  - Describe the problem, draw a picture, sketch something on a napkin
  - Break the problem into smaller pieces, solve, integrate as you go
- Learn how to debug and test – beyond print statements
- Use Labs and Office Hours to maximum advantage
- Code every day, it will get easier, we promise
- You (mostly) can't break the computer, so experiment
  - If you're curious about how/if something works, so are we, let's break things together
- Ask questions
  - Please allow at least 24 hours for email replies
- Help us help you
  - Describe your problem (line number, compiler error messages), Describe the symptoms
  - Send a copy of your code
  - What have you tried so far to fix the problem? (So we don't waste time on things that don't work)
  - If you solve the issue, please tell us (so we can stop working on it
- Make a 'safety' submittal, sometimes Canvas crashes. No penalty for multiple submittals, we'll grade the last one
- Have fun!

Course (Lab) Overview

- If there is a conflict, the Syllabus rules
- If we say something different than Dr Unan, go with Dr Unan (and let us know)
- Honor Code
- It doesn't matter which lab you attend, but priority seating to those assigned
- 300-level course, expectations are a little higher

Draft Lab Outline (subject to change)

Things we'll learn:
C
Command Line interface
Bits, Booleans, how the Computer "sees" the world
Assembly
Memory, Registers
How computers work "behind the scenes"
How to improve performance and our higher level code

Match faces with names ! (Well, at least eyeballs, eyebrows, and beards)

Why CS330?
To understand something, it's often helpful to understand one level below
Helps us to optimize our code, understand 'why', make valuable stackoverflow comments

- Lab1: Env config, Hello World!
- Lab2: C: Basics, Make
- Lab3: C: Basics Part II
- Lab4: Binary, Boolean algebra, Two's, IEEE754
- Lab5: C: pointers, GDB intro
- Lab6: C: Arrays, Strings
- Lab7: C: Debugging
- Lab8: C to Assembly
- Lab9: AS: Intro
- Lab10: AS: Arithmetic Ops
- Lab11: AS: Conditional / Unconditional Jumps
- Lab12: AS: Arrays, Functions
- Lab13: Performance

Why? One example


```
15  float Q_rsqrt(float number){
16      long i;
17      float x2, y;
18      const float threehalfs = 1.5F;
19
20      x2 = number * 0.5F;
21      y = number;
22      i = * (long *) &y;                      // evil floating point bit hack
23      i = 0x5f3759df - (i >> 1);              // what the f*$% ?
24      y = * (float *) &i;
25      y = y * (threehalfs - (x2 * y * y));    // 1st iteration
26  //  y = y * (threehalfs - (x2 * y * y));    // 2nd iteration, can be removed
27
28      return y;
29  }
```

THE UNIVERSITY OF ALABAMA AT BIRMINGHAM

Environment Set-Up

Follow the handout for your preferred operating system

THE UNIVERSITY OF ALABAMA AT BIRMINGHAM

Typical Workflow

How to create and submit your work

Sample Workflow

Create

1 Open text editor of choice, and write your C or AS program

```
#include <stdio.h>

int main(){
    printf("Hello World!");
    return 0;
}
```

1b Save your C file as ***filename*.c**
Save Assembly as **filename.s**

Compile

2 Open your terminal, shell

2b Navigate to the folder where your file is stored

2c Compile via command:
gcc *filename*.c
(and any optional command line flags)

2d If there are errors, correct file,
Save, and re-compile

Run / Test

3 In the same terminal window,
In the same folder,
Run via command:
./a.out or ./name

Code must run on Vulcan

Can push to Vulcan via:
Mac / Linux: ssh, scp
Windows: Putty

Note: can also use GIT

Submit to Canvas

4
- Good to make a 'safety submit' early in case Canvas has issues. We'll grade the last submittal
- ***If something in your code doesn't work correctly, please say so***

Text Editors You Can Use

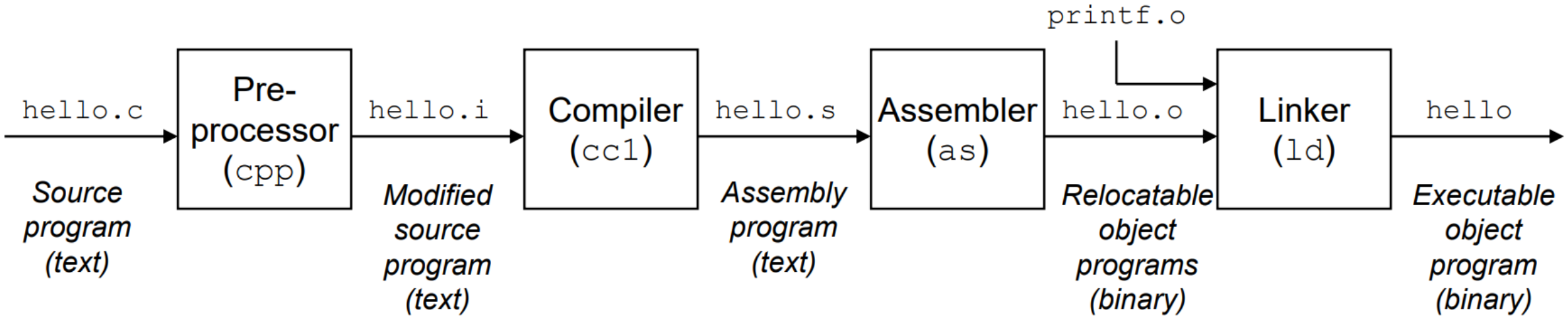
- We'll use VSCode for this course, but you can also use:
  - Notepad++
  - Nano
  - Vim (:q to quit) ← check these out
  - EMACS ← check these out
  - ATOM
  - TextEditor
- NOT Microsoft Word (it adds embedded characters)

C (and Assembly) is a Compiled Language

We'll be using the GNU Compiler Collection (GCC)
When we type:

gcc hello.c –o hello

The code is compiled as follows:



Compilation System

Common Compiler Options

- The most basic form: *gcc hello.c*
  executes the complete compilation process and outputs an executable: *a.out*
- Use option –o: *gcc hello.c –o hello*
  to produce an output file with name 'hello'
- Use option –Wall: *gcc –Wall hello.c –o hello*
  to enable all warnings
- Use option –g: *gcc –Wall –g hello.c –o hello*
  to produce debugging information for GDB (debugger)
- Use option –O: *gcc –O hello.c -o hello*
  to set the compiler's optimization level, various levels (-O, -O1, -O2, -O3, -Ofast)
- Use option –std=<standard>: *gcc hello.c –std=c89 –o hello*
  Swaps between different syntax standards. Vulcan defaults to c89, your laptop is likely c99. One difference, you cannot declare a variable inside a for loop on Vulcan (c89)

Use –lm: *gcc hello.c -0 hello -lm*
to link math.h (e.g. sqrt function)

Use –lpthread:
to link pthread.h

- Use option –E: *gcc –E hello.c > hello.i*
  to produce the output of preprocessing stage
- Use option –S: *gcc –S hello.c > hello.s*
  to produce only the assembly code
- Use option –C: *gcc –C hello.c*
  to produce only the compiled code (without linking)

Most Commonly Used to compile C:
gcc –Wall –g hello.c –o hello

Today's Assignment

- "Hello World"
- Submit the binary (not the source), and Independent Completion Form to Canvas

Next Week

- MLK, Jr Day on Mon – no Labs next week