

CSE 3100: Systems Programming

Lecture 1: Course Introduction

Department of Computer Science and Engineering
University of Connecticut

CSE 3100: Systems Programming

- Introduction to system-level programming with an emphasis on C programming.
- We will also focus on process management, and small scale concurrency with multi-threaded programming.
- Special attention will be devoted to proficiency with memory management and debugging facilities both in a sequential and parallel setting.



The Money Making Computer Problem



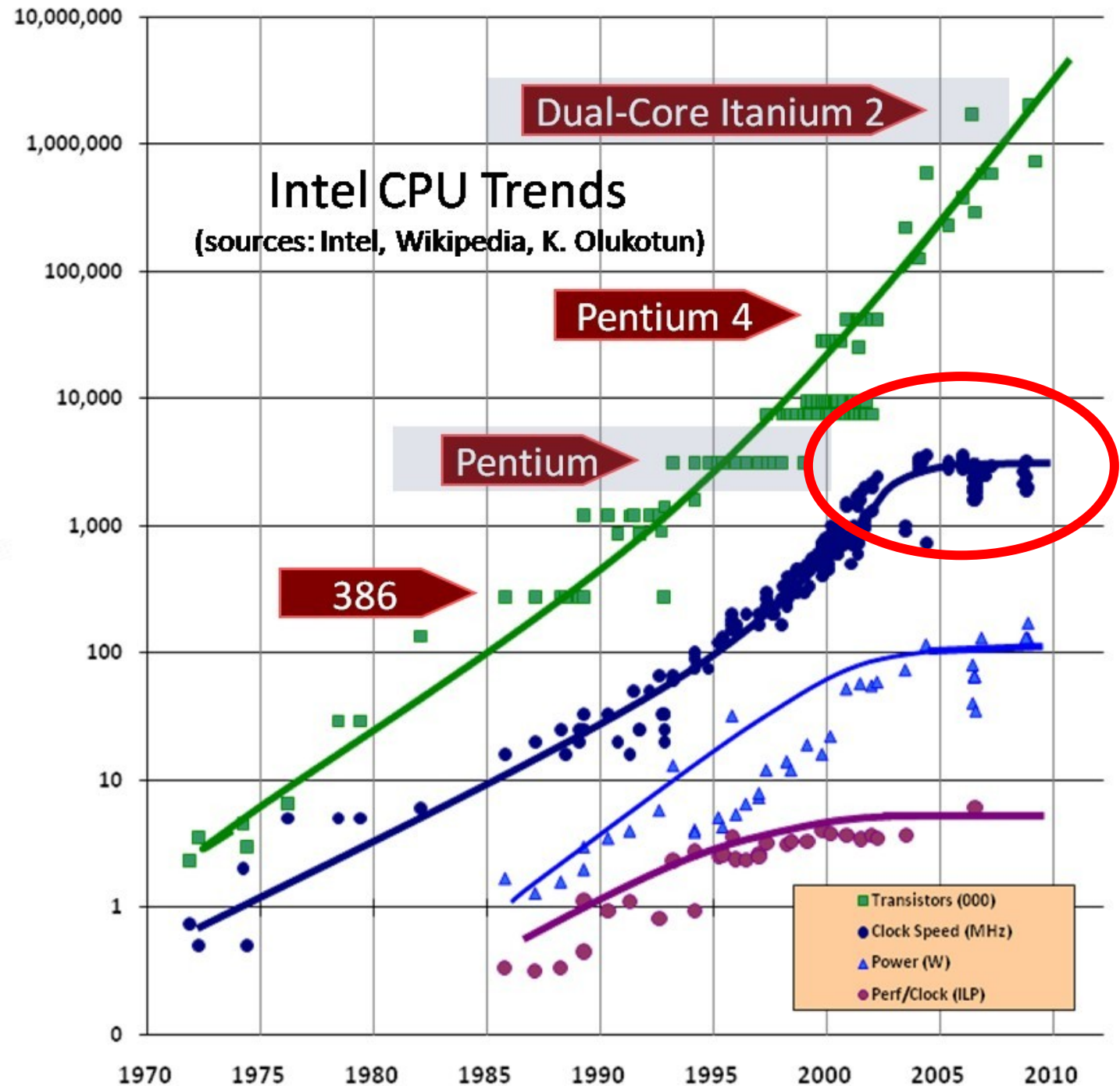
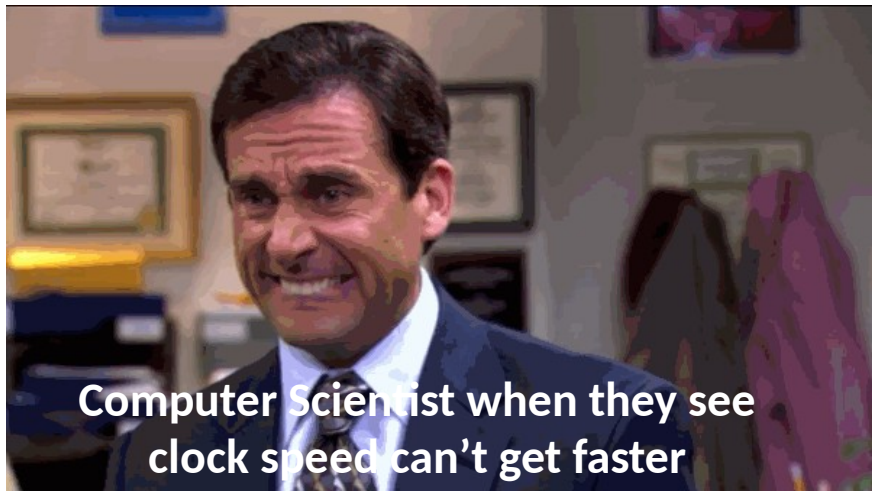
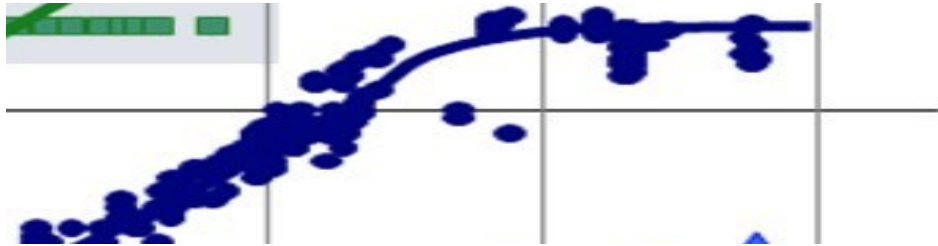
- Imagine you work for Apple. They have a computer program that does some computation in native Python code.
- Every time the computation completes you get 1 cent. Currently the program takes 1 hour to run. So you get 1 cent per hour.
- You want to get rich fast. How can you make the computer program run faster?

Basic ideas to make the program run faster:

1. Switch the language to C. In many cases C is the faster language to work with*.
2. Use a faster computer!

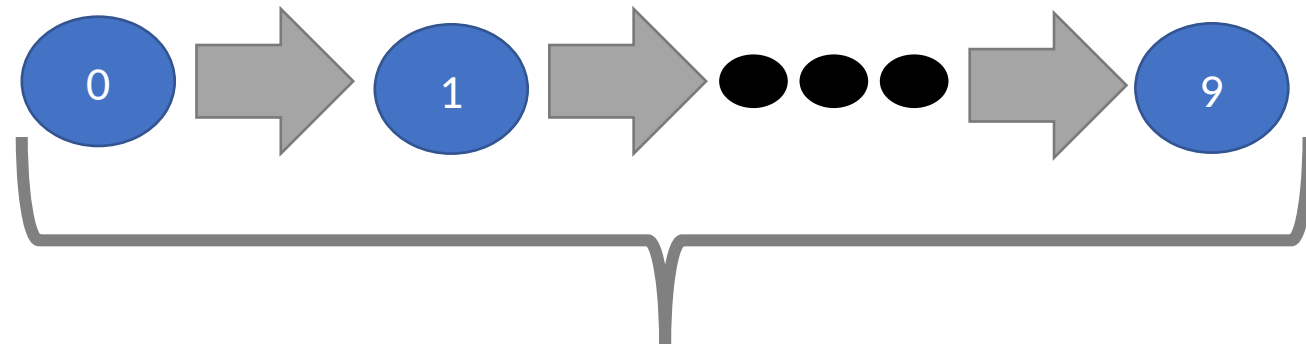
Every year computers get faster. So every year you should be able to make more money off the program right?

But what happens when you can't go faster?



Parallelism

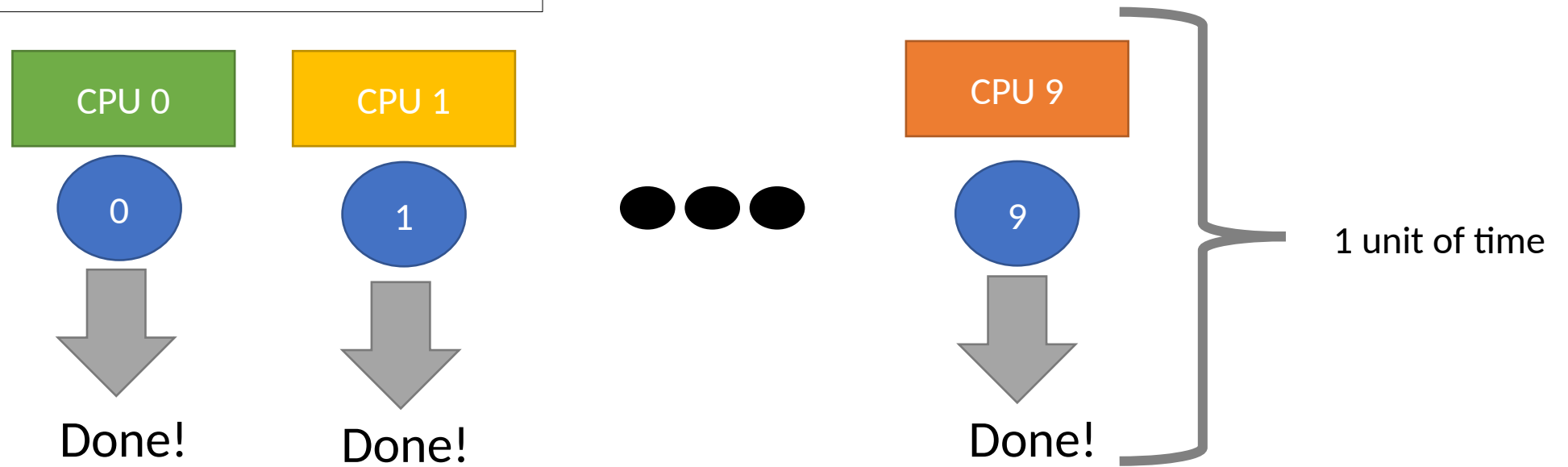
```
1 import numpy
2 x = numpy.zeros(10)
3 for i in range(0, 10):
4     x[i] = i + 2
```



10 units of time to run

- How long does this take to run if each iteration of the for loop executes in 1 unit of time?
- Are there any dependencies between iterations?


```
1 import numpy
2 x = numpy.zeros(10)
3 for i in range(0, 10):
4     x[i] = i + 2
```



- Idea: Run each part of the loop on a separate central processing unit (CPU).
- What would take 10 units of time for 1 CPU now can take 1 unit of time for 10 CPUs.

The Money Making Computer Problem Conclusions

Parallel C Program



- In order to execute code faster, it is better to work in C.
- To maximize the usage of our hardware, we need to understand how to do parallel programming (which can be done in C).
- Both these concepts will be taught in this course!

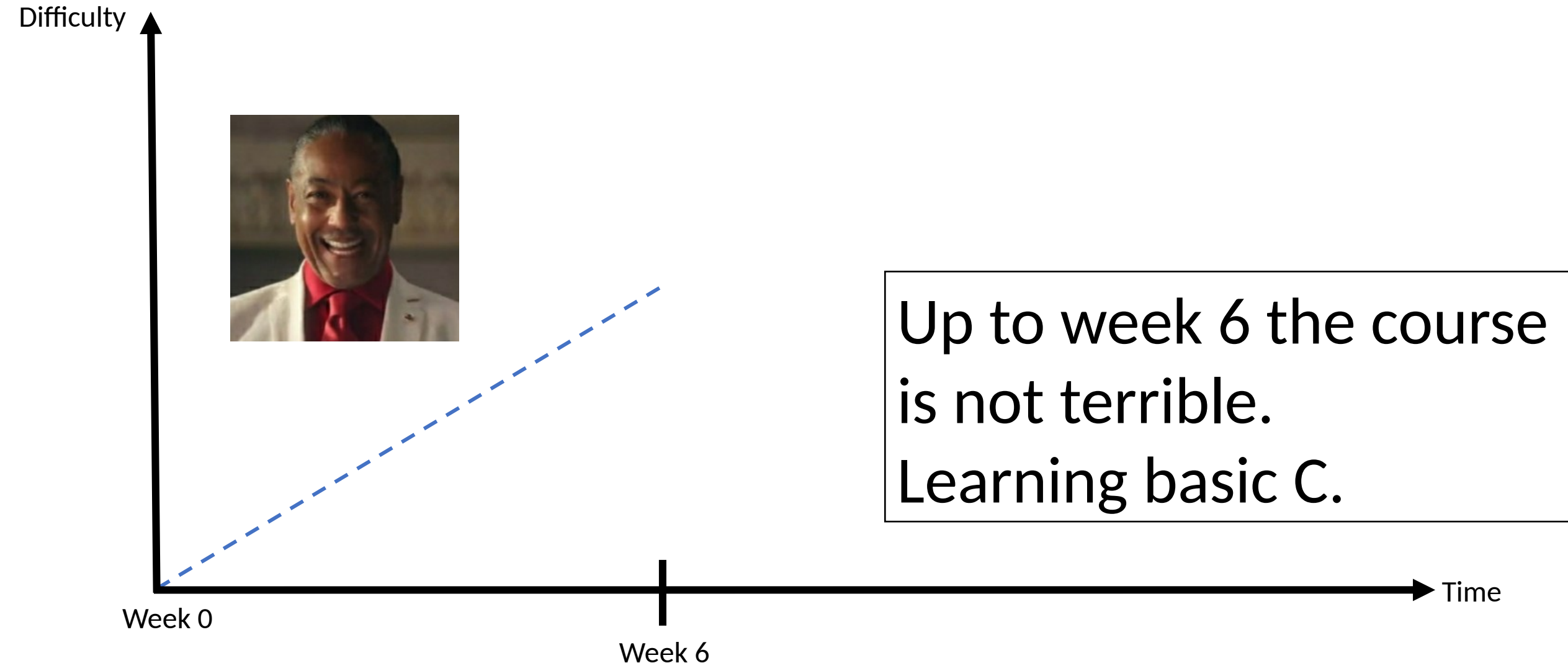
Course Logistics

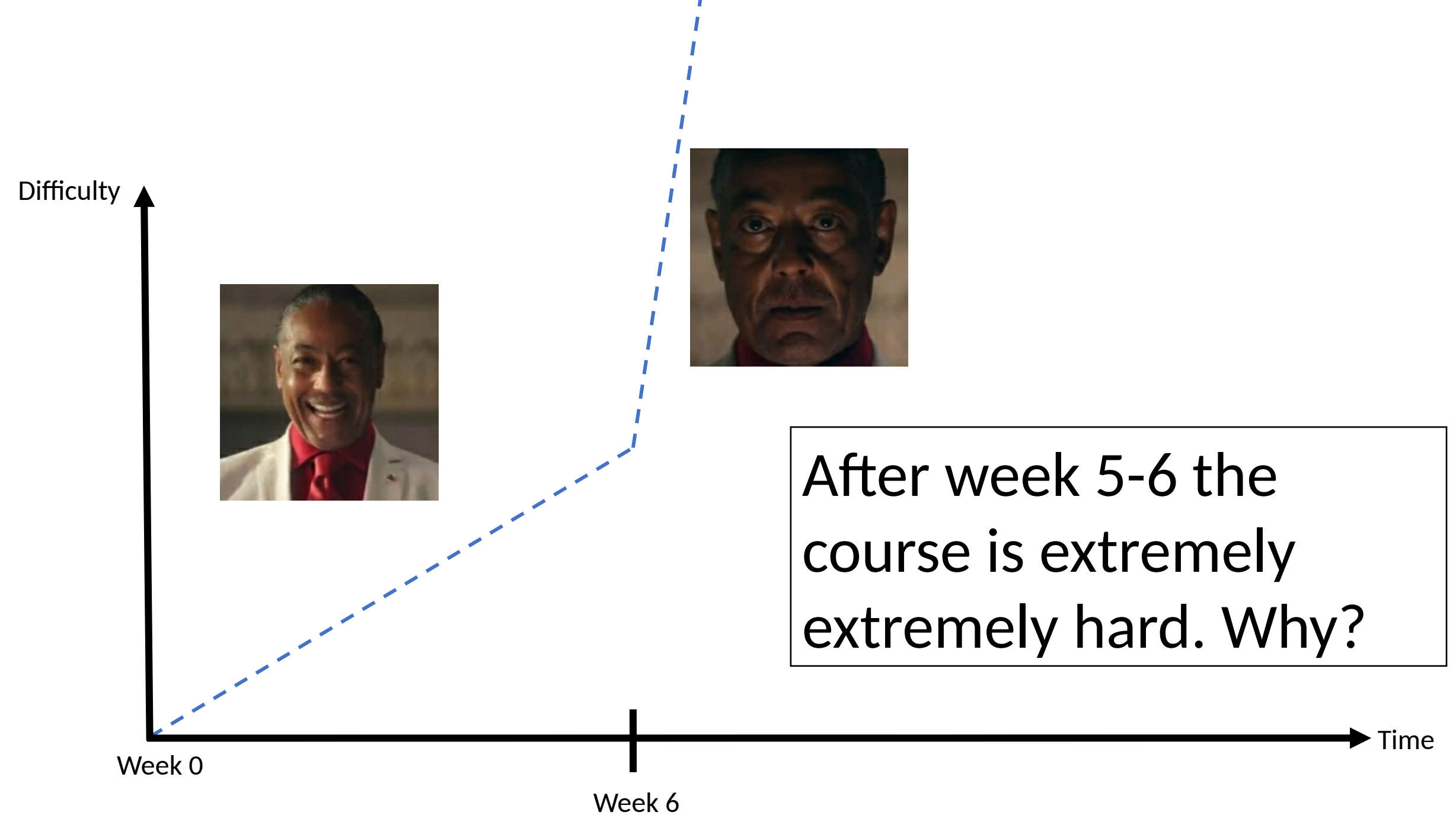
- *How will you be graded?*
- *How hard is the course?*
- *How can you do well in the course?*

How will you be graded?

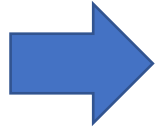
Course Components	Weight
Lab assignments	10%
Homework assignments	30%
Exam #1	15%
Exam #2	20%
Exam #3	25%
Extra Credit	6%

How hard is the course?





After week 5-6 the course is extremely extremely hard. Why?



Week #5	Pointers and structures (ABC Ch9, K&R Ch6)
Week #6	I/O and library functions (ABC Ch11, K&R Ch7)
Week #7	Processes (ABC Ch12)
Week #8	Pipes (ABC Ch12). Intro to threads
Week #9	Thread management. Thread synchronization: mutex and condition variables
Week #10	Thread synchronization: read-write locks and barriers
Week #11	Threads misc. topics: local storage, cancellation, real-time scheduling, false sharing
Week #12	Intro to Sockets (Beej's guide)
Week #13	Client-server communication using sockets (Beej's guide)
Week #14	Signal (ABC Ch12) and Misc. Topics

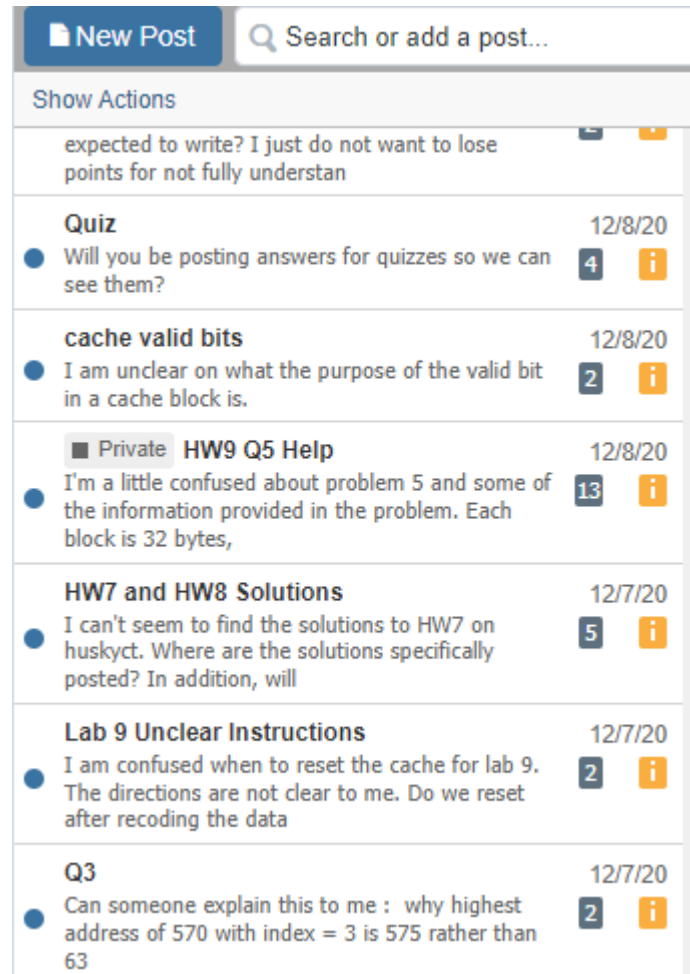
How can you do well in the course?

- Start coding assignments *EARLY*.
- When you struggle, get help. Three types of help offered!
 1. TA Office Hours
 2. Piazza
 3. Professor Office Hours

1. For Homework/Lab Help: TA Office Hours

TBD

2. For Homework/Lab Help: Piazza



The screenshot displays the Piazza forum interface. At the top, there is a 'New Post' button and a search bar labeled 'Search or add a post...'. Below this is a 'Show Actions' button. The main content area lists several posts, each with a title, a brief description, a date, and a response count. The posts are as follows:

Post Title	Date	Response Count
expected to write? I just do not want to lose points for not fully understand	12/8/20	4
Quiz	12/8/20	4
Will you be posting answers for quizzes so we can see them?	12/8/20	2
cache valid bits	12/8/20	2
I am unclear on what the purpose of the valid bit in a cache block is.	12/8/20	13
Private HW9 Q5 Help	12/8/20	13
I'm a little confused about problem 5 and some of the information provided in the problem. Each block is 32 bytes,	12/7/20	5
HW7 and HW8 Solutions	12/7/20	5
I can't seem to find the solutions to HW7 on huskyclt. Where are the solutions specifically posted? In addition, will	12/7/20	2
Lab 9 Unclear Instructions	12/7/20	2
I am confused when to reset the cache for lab 9. The directions are not clear to me. Do we reset after recoding the data	12/7/20	2
Q3	12/7/20	2
Can someone explain this to me : why highest address of 570 with index = 3 is 575 rather than 63	12/7/20	2

3. For Concept Questions: Professor Office Hours

Name	Email	Room or online Link
Wei Zhang	<u>wei.13.zhang@uconn.edu</u>	Monday: 8-9am ITE 456 https://uconn-cmr.webex.com/uconn-cmr/j.php?MTID=me66bf299a951bdbd19d6f855e72c747c

Other Announcements

- There is a lab 0 due this week!
- Please make sure you can log into the virtual machine and do the lab.

Course Questions?

The C Language

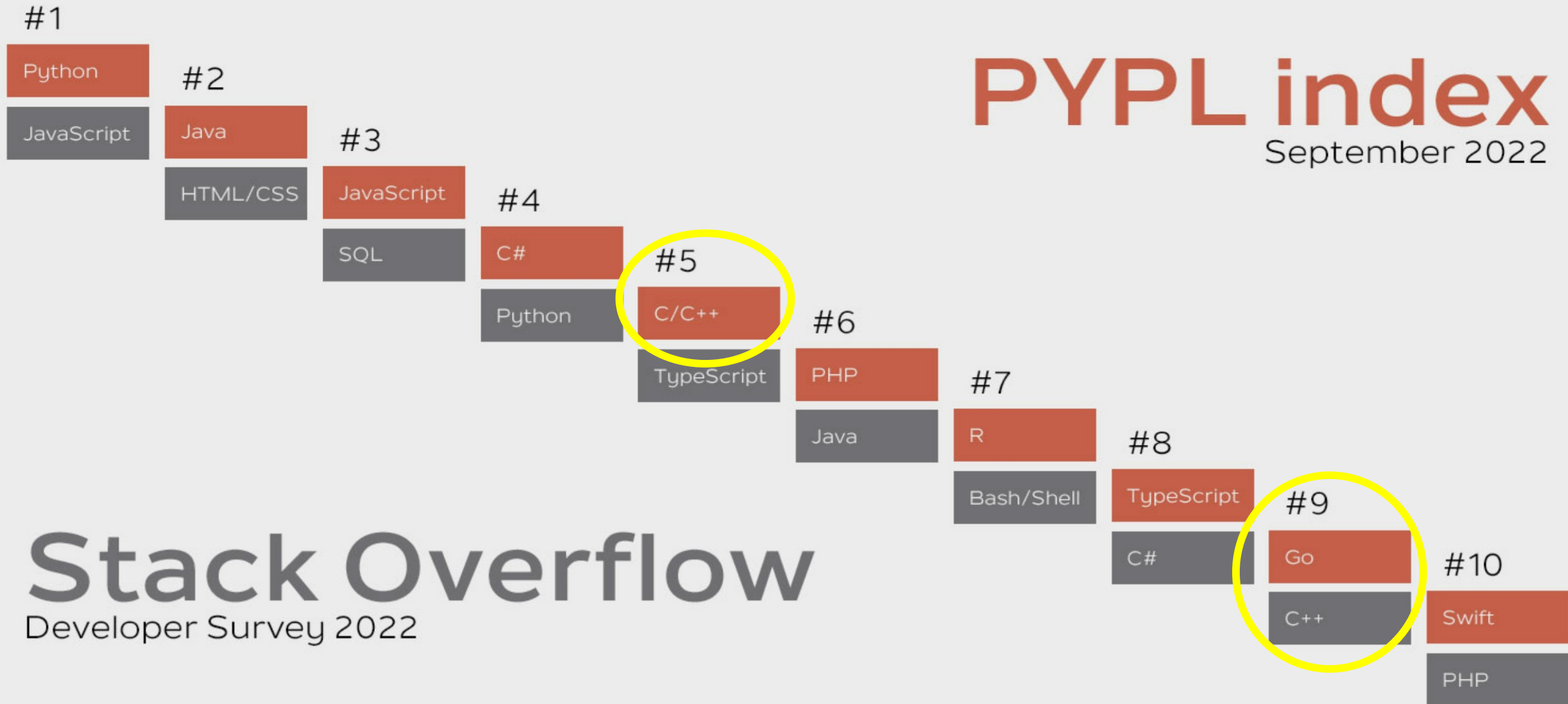
- C is a general-purpose programming language that was originally designed by Dennis Ritchie of Bell Laboratories and implemented there in 1972.
- It was first used as the systems language for the UNIX operating system.

Is CSE 3100 just a course to learn a “dead” language from the 70s???



Most popular programming languages 2022

(<https://www.stackoverflow.com/blog/most-popular-programming-languages/>)

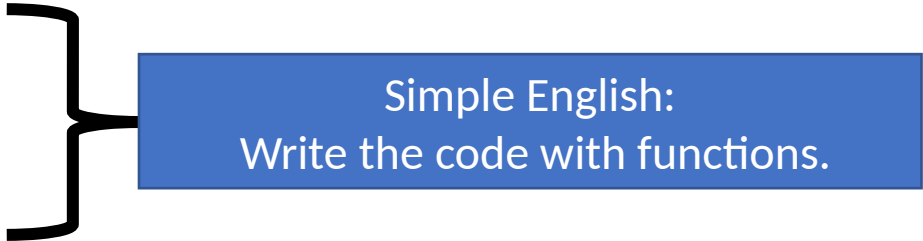


C Design Principles 1

- C should be...
 - Simple
 - Easy to compile
 - Typed (as in we define the data types for variables)
 - Support low-level memory access
 - Ideal for embedded controller, OS, ...
- Yet...
 - C is *powerful*
 - C is *fast*

C Design Principles 2

- C is a purely **procedural** language
 - No object-orientation whatsoever
- Central Dogma
 - Object of interest: Computations
 - Main abstraction tool: **Procedures/functions**
 - Caller / Callee
 - Abstracts away “How things are done”
 - Programming means
 - Organizing processes as procedures
 - Composing processes through procedure calls



Simple English:
Write the code with functions.

Procedural Programming in C

- Generates very efficient code.
- Exposes as many low-level details you wish to see.
- Provides full control over memory management (no Garbage Collection!)
- The Programmer is fully in charge.

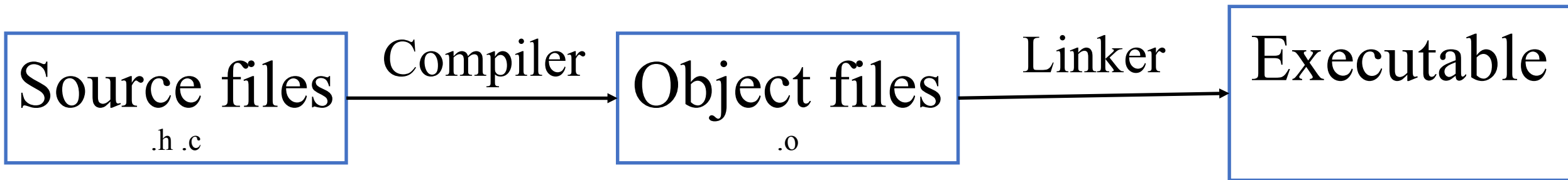


Pictured: How it should feel to program in C.

What do we need to program in C?

1. Use text editor to edit source files
2. Use compiler to generate .o files
3. Use linker to link multiple .o files into executables

Text editor



Example C Code

```
#include <stdio.h>

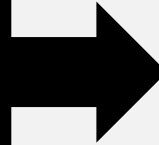
/* comments */
// single line comments

int main(void)
{
    printf("Hello, world!\n");
    return 0;
}
```

Comments in C

```
#include <stdio.h>
```

```
/* comments */  
// single line comments
```



```
int main(void)  
{  
    printf("Hello, world!\n");  
    return 0;  
}
```

- The compiler ignores everything between “/*” and “*/”
- Comments are meant to be human readable!
- Comments can be multi-line
- Cannot be nested
- Everything starting from “//” is ignored in a line

The 'main' function in C

```
#include <stdio.h>

/* comments */
// single line comments

int main(void)
{
    printf("Hello, world!\n");
    return 0;
}
```



- A special function that defines the entry point for the program.
- This is where the Operating System transfers control once the program starts.
- **void** indicates no arguments
- **int** before 'main' indicates the main function returns an integer.

printf in C

```
#include <stdio.h>

/* comments */
// single line comments

int main(void)
{
    printf("Hello, world!\n")
    return 0;
}
```

- What is 'printf' ?
- A C library function to print on the standard output for a process.
- It takes a string as the argument
- Between double quotations, like "This is a string."
- '\n' is a newline character

Why do we 'return' anything from main() ?

```
#include <stdio.h>

/* comments */
// single line comments

int main(void)
{
    printf("Hello, world!\n");
    return 0;
}
```



- When the process terminates, it can tell the OS how things went.
- This is the way to report back.
- Returning 0 means 'everything went according to plan'.

What's up with #include ?

```
#include <stdio.h>
```



```
/* comments */  
// single line comments  
  
int main(void)  
{  
    printf("Hello, world!\n");  
    return 0;  
}
```

- It “imports” a header file.

What is a header file?

- A file which contains C function declarations and macros to be shared between several source files.

What is a macro?

- A macro is a fragment of code which has been given a name.

Compiling and Executing a C Program

hello.c program

```
#include <stdio.h>

int main(void)
{
    printf("Hello world!\n");
    return 0;
}
```

Terminal

```
$ cc hello.c
$ ./a.out
```

- What is **cc** doing really ?
 - Three steps
 - **preprocesses** hello.c
 - **compiles** hello.c to hello.o
 - **links** hello.o with **libc**
 - **writes** executable file **a.out**
 - You **can** and **often will** separate those steps!
- The name of the executable can be changed
 - a.out is the default
- **./** indicates a.out in the current directory
 - Otherwise the OS searches directories in 'PATH'

Lecture Conclusions

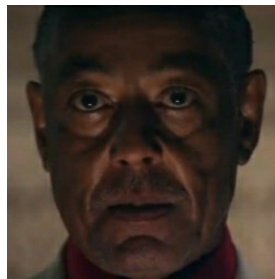


Pictured: How it should feel to program in C.

Week 1



Week 6



- C is a low level but powerful programming language.
- We'll start with basic syntax and programming and will progress to more advance concepts like processes.
- The course is very challenging so follow along closely and seek help when needed at office hours.

Figure Sources

1. https://media.pitchfork.com/photos/626be39b8eeb4ac0c1275b4e/master/w_1280%2Cc_limit/Future-I-Never-Liked-You-2022.jpeg
2. <https://i.ytimg.com/vi/O1hCLBTD5RM/maxresdefault.jpg>
3. <https://1000logos.net/wp-content/uploads/2016/10/apple-emblem.jpg>
4. <https://preview.redd.it/xpcv28tuztqz.png?auto=webp&s=d27bf66316763d9a0cf420f68d34287df7992c7e>
5. <http://www.extremetech.com/wp-content/uploads/2012/02/CPU-Scaling.jpg>
6. <https://i.imgflip.com/5q88rn.png>
7. [https://en.wikipedia.org/wiki/Scheme_\(programming_language\)](https://en.wikipedia.org/wiki/Scheme_(programming_language))
8. <https://images.emojiterra.com/google/android-11/512px/1f631.png>
9. <https://alanzeichick.com/wp-content/uploads/king3.png>