Intro to Computer Science
CS-UH 1001, Spring 2022

Lecture 0 – Introduction

Course/Slide Credits

- Note: Most lecture slides are based on those developed by Prof. Yasir Zaki (NYUAD, Department of Computer Science)

Today's Lecture

- Why Computer Science?
- Course logistics
- Computer basics
- Computer setup

Hello World from us!

- Your lecturer:
  - Thomas Pötsch
  - thomas.poetsch@nyu.edu

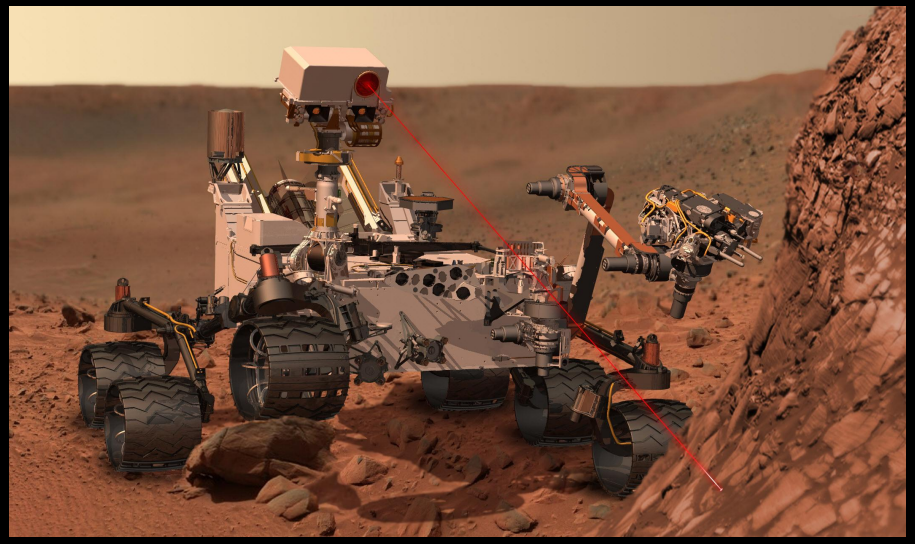
- Your teaching assistant:
  - Shan Randhawa
  - smr693@nyu.edu

Introduction Round

- Your name
- Your computer skills?
  - Do you have any programming experience?
  - Why are you taking this class?
- What is Computer Science to you?
- What do you hope to gain from this course?

Why Computer Science?

- Computer science is all around us
- It's a growing field with excellent job opportunities
- You can do lots of creative and challenging things

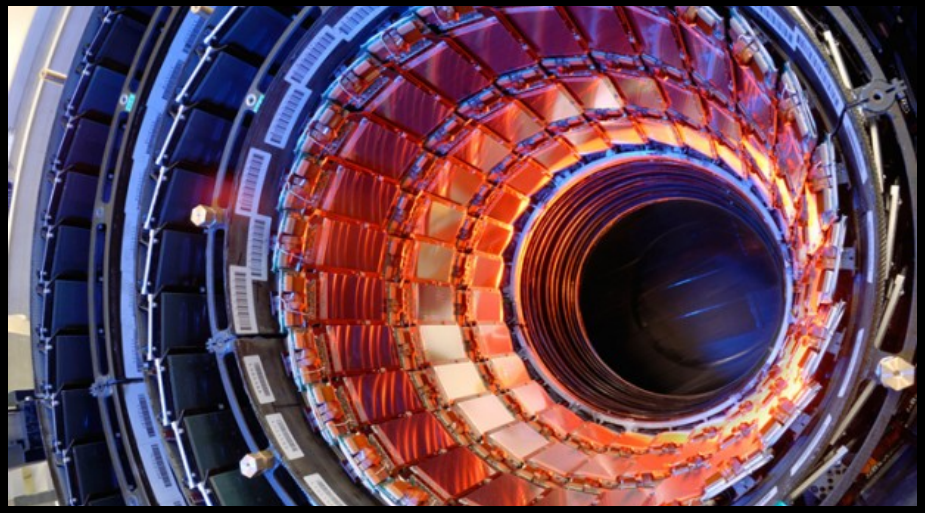


Space

Movies

Why Computer Science?

Why Computer Science?



CERN Particle Accelerator

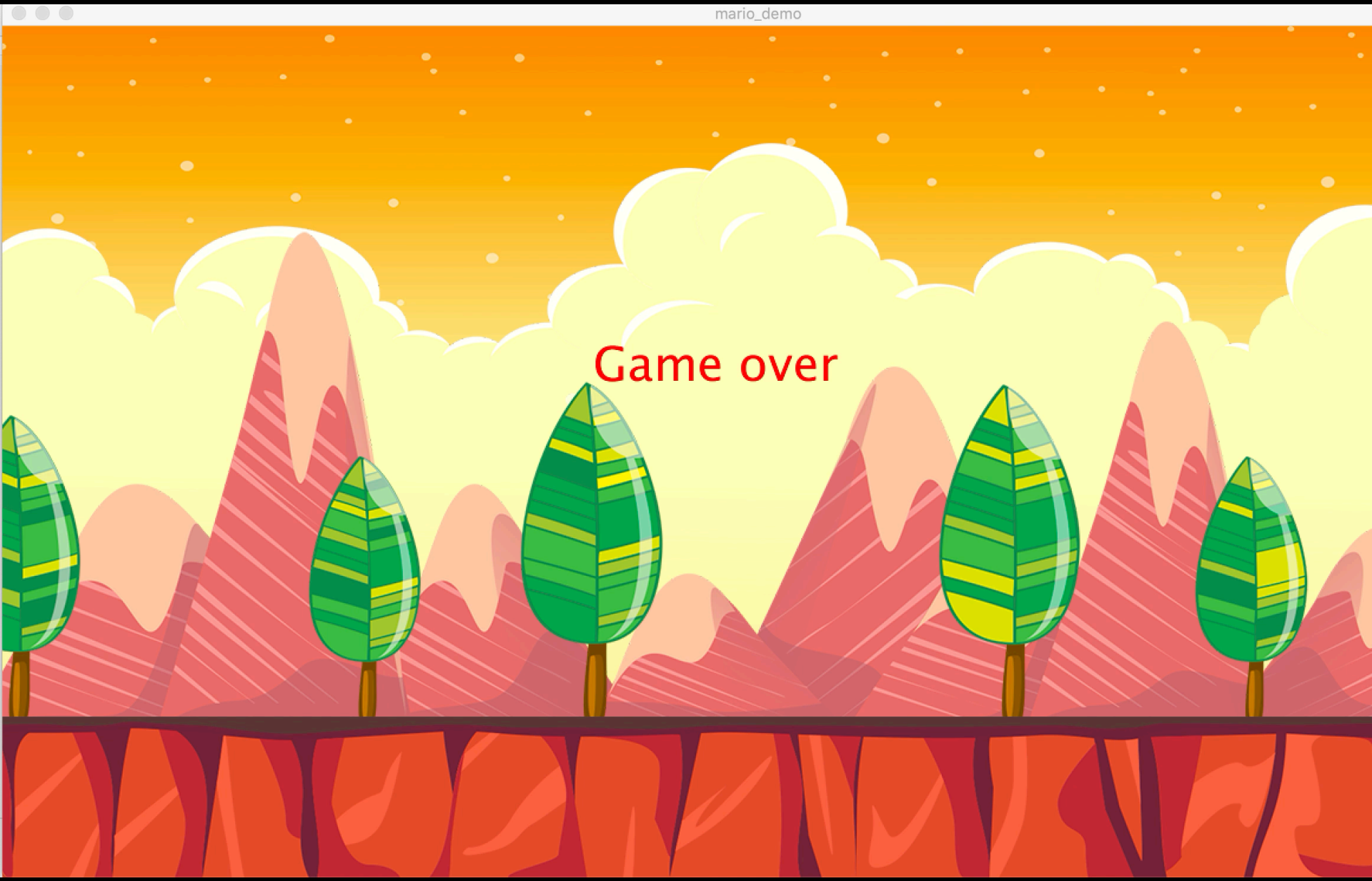
Computer Science and Programming

- The most important skill for a computer scientist is **problem solving**
- Programming is the basic tool in Computer Science to solve problems

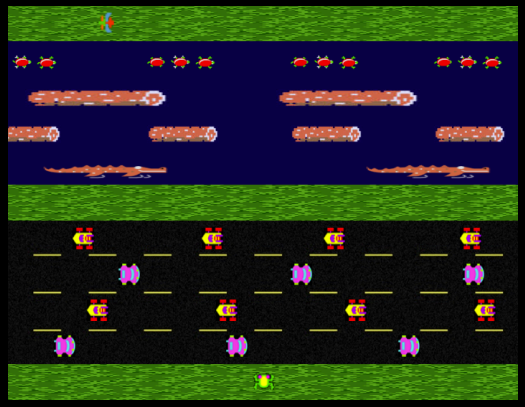
- Good news!
  - For this course, no prior programming knowledge is required

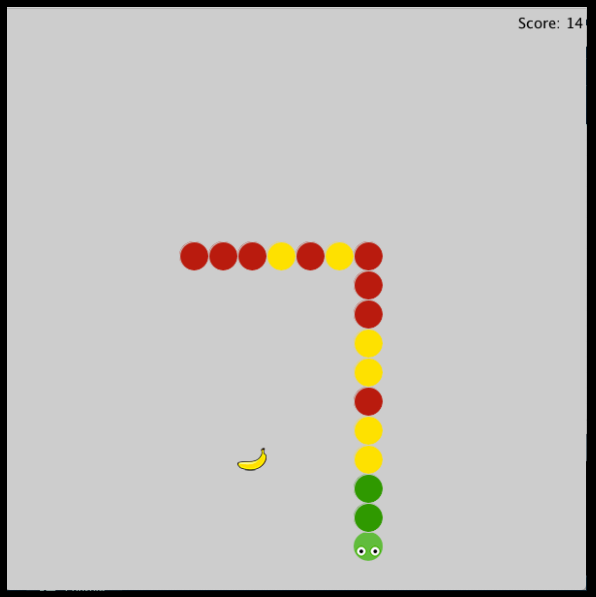
Course Goals

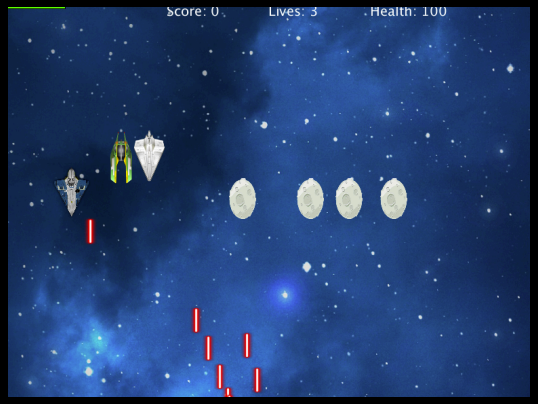
- Provide an introduction to computer science and programming
- Learn how to approach and solve computer science problems
  - Obtain the ability to translate problems into code
- Learn the basic concepts and fundamentals of computer programing
  - Obtain the necessary vocabulary to read code
- Learn how to write computer programs
  - Accumulated enough knowledge to write code
  - Build arcade like games (Super Mario, Tetris, Pac-man, etc.)

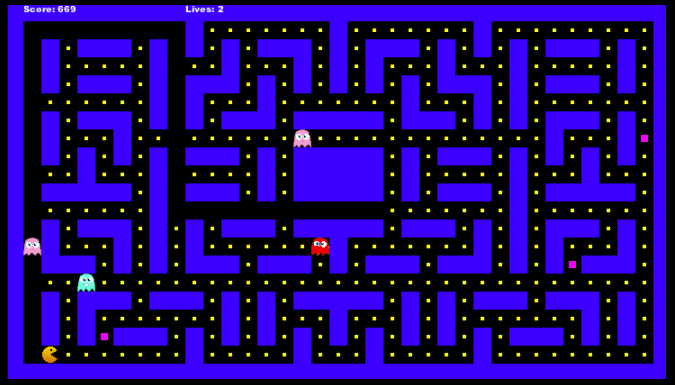

mario_demo
Game over

Games from Previous Years









Class Structure

- We will meet twice a week for 2:40h
- This class has a built-in lab component
- Interactive lecture:
  1. Introduce a concept (~30 min)
  2. Practice the concept during the lab (hands-on sessions) (~30 min)
  3. Solve the exercise together (~15 min)
  4. Break (~10 min)
  5. go back to 1.

Online Resources

- The course website is on Brightspace

- There you will find:
  - Course Syllabus
  - Announcements
  - Lecture slides (uploaded before class)
  - Solved exercises (uploaded after class)
  - Homework assignments
  - All your grades (check them regularly!)

Text Books

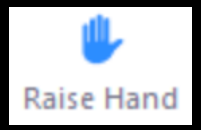
- **"Starting out with Python"** by Tony Gaddis, Global Edition
  - You should have received a copy from the book store
  - https://mlm.pearson.com/northamerica/myprogramminglab/
  - Select "I'm outside the US and Canada" after login
  - Access Code: **PEAR-52218-PVCC-51**

- **"How to Think Like a Computer Scientist: Learning with Python 3"**, by Peter Wentworth, Jeffrey Elkner, Allen B. Downey and Chris Meyers
  - You can download it here:

  www.openbookproject.net/thinkcs/

Guidelines for Online Classes

- Please turn on your camera!!
- If you have questions during the lecture:
  - "Raise hand"



- If you have questions during the lab/exercises:
  - Use the chat to send us a message and we will assist you (in breakout rooms if needed)
- Remember:
  - Any form of recording is not permitted!

Graded Activities

Type	Overview	Weight
Class participation	Interaction, exercises, etc.	5 %

- If you miss a class, you might miss a quiz
  - Be on time, otherwise you might miss a quiz
  - There will be no retakes and no late hand outs for the quizzes

Graded Activities

Type	Overview	Weight
Class participation	Interaction, exercises, etc.	5 %
Random Quizzes	There will be random pop quizzes based on the previous class material	10 %
Assignments	3 in total	15 %
Final project	Group based (last 2-3 weeks)	15 %
Mid-term exam	Last lecture before break	25 %
Final exam	Date as advertised on Albert	30 %

- If you miss a class, you might miss a quiz
  - Be on time, otherwise you might miss a quiz
  - There will be no retakes and no late hand outs for the quizzes

Class Participation

- Attend the class (on time)
  - and show us that you are there (don't "hide" behind your screen)

- Actively contribute to the class
  - Verbally, not only on the chat
  - Ask questions at any time (and ask many!)
  - Respond to questions asked

- Warning: On low class participation, we will ask random students to present their exercise solution in front of the class

Homework Assignments

- We will have 3 homework assignments throughout the semester
- Each is due after 10 days
  - Usually at exactly 11:59 pm
  - No extensions will be granted
- Late submissions:
  - 1 sec = 1 minute = 1 hour = 1 day
  - -20% penalty for each day late
  - submission via email are not accepted

Agenda until Fall Break*

Week	Topic	Assignments
1	Command line terminal, Programming basics	
2	Input and output, strings and sequences	
3	Boolean logic, decision structures, loops	HW 1 release
4	Functions and modules	
5	Dictionaries, File I/O	HW 2 release
6	Exception handling, Scientific data analysis	
7	Midterm review, Midterm exam (last class before break)	

*subject to change

Grade Distribution

Score	Grade
95 – 100	A
90 – 95	A-
87 – 90	B+
83 – 87	B
80 – 83	B-
77 – 80	C+
73 – 77	C
70 – 73	C-
67 – 70	D+
63 – 67	D
< 63	F

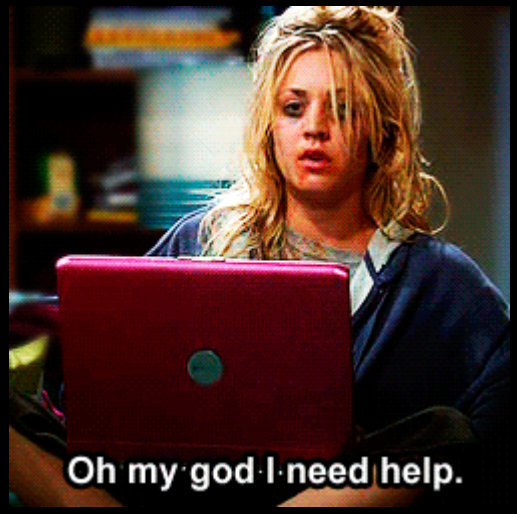
No curving and no extra credit!

How to succeed in this course?

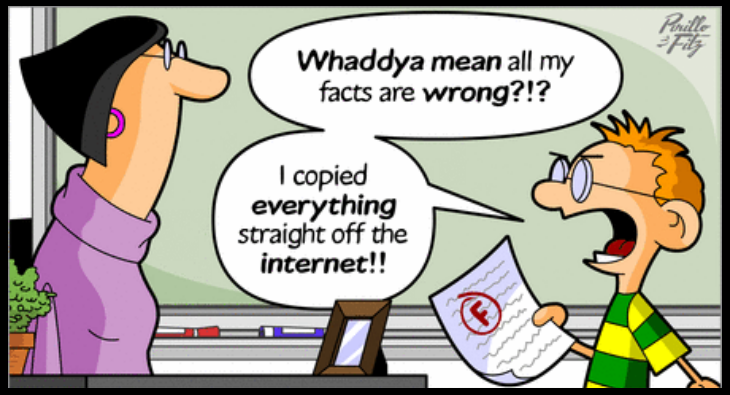
- Attend and participate in class
  - Many of the exam questions are related to discussions in class
- Do ALL exercises and assignments yourself!
- Practice makes perfect – study as many examples as possible
  - Understand and enhance the examples provided in class/labs as well as the ones in the textbook
- Seek help when needed and ask questions (and do it EARLY)
  - During lectures or come to office hours

(Virtual) Office Hours

- If you have questions or need help, please come during office hours or schedule an appointment by email
  - My office hours:
    - By appointment
    - tp53@nyu.edu
  - Shan's office hours:
    - By appointment
    - smr693@nyu.edu
- We are happy to help!



Academic Integrity



Academic Integrity and Dishonesty

- Academic integrity means
  - Responsibility
  - Honesty
  - Respect

- Academic dishonesty and integrity violations include
  - Plagiarism
    - Submitting work of another person
    - Submitting work previously used/submitted
    - Submitting work from other sources
  - Seeking unauthorized guidance
  - etc

Academic Integrity Violations

- Academic integrity violations **WON'T** be tolerated in this course:
  - **When** you are discovered, both will get a 0
  - **Every** incident will be reported according to the policies of NYU Abu Dhabi*
  - Absolutely **NO** second chances

*https://students.nyuad.nyu.edu/academics/registration/academic-policies/academic-integrity/

How to Avoid Integrity Violations?

- Submit your own work
  - Do not copy work from others, in parts or fully
  - Do not share your work with others
  - Do not complete work for others
  - Avoid working in groups
- Most importantly: Don’t be afraid to ask us for help
- You are allowed to use work from the slides or exercises!


Let's Get Started

Computer Basics

- A computer consists of the following main components:
  - Input (e.g. keyboard, mouse)
  - Output (e.g. screen)
  - Central Processing Unit (CPU)
  - Memory (RAM)
  - Storage unit (HDD)

- Computer execute instructions, called programs

What is a Program?

- It's a collection of instructions that perform a specific task when executed by a computer
- Programs are written in programming languages
- A collection of programs are referred to as software

Programming Languages

- A programming language is used to interact with a computer
- A set of rules and symbols used to construct a computer program

- Some of the instructions that almost appear in every language:
  - Input
  - Output
  - Computation

Programming Languages

- There are two kinds of programming languages:
  - Low level (Machine language, Assembly)
    - Mostly binary-code, machine-dependent and not portable
  - High level (Python, C, C++, Java, etc.)
    - Use English-like commands
    - Machine independent
    - Portable (must be compiled for different platforms)

Machine Language

- A representation of a computer program which is actually read and understood by the computer
  - A program in machine code consists of a sequence of machine instructions
- Instructions:
  - are in binary code (0 and 1)
  - specify operations and memory cells involved in the operation

Hello World in Machine Language

0010 0001 0000 1010 0000 0000 0000 0000

```
21 0a 00 00    #moving "!\n" into eax
0c 10 00 06    #moving eax into first memory location
6f 72 6c 64    #moving "orld" into eax
08 10 00 06    #moving eax into next memory location
6f 2c 20 57    #moving "o, W" into eax
04 10 00 06    #moving eax into next memory location
48 65 6c 6c    #moving "Hell" into eax
00 10 00 06    #moving eax into next memory location
00 10 00 06    #moving pointer to start of memory location
10 00 00 00    #moving string size into edx
01 00 00 00    #moving "stdout" number to ebx
04 00 00 00    #moving "print out" syscall number to eax
80             #calling kernel to execute the print to stdout
01 00 00 00    #moving "sys_exit" call number to eax
80             #executing it via linux sys_call
```

Assembly Language

- A symbolic representation of the machine language of a specific processor
- Converted to machine code by an assembler
- Usually, each line of assembly code produces one machine instruction (One-to-one correspondence)
- Programming in assembly language is slow and error-prone but more efficient in terms of hardware performance

Hello World in Assembly

```
section .text
        global _start                   ;must be declared for linker

_start:                                 ;tells linker entry point
        mov  edx,len                    ;message length
        mov  ecx,msg                    ;message to write
        mov  ebx,1                      ;file descriptor (stdout)
        mov  eax,4                      ;system call number (sys_write)
        int  0x80                       ;call kernel

        mov  eax,1                      ;system call number (sys_exit)
        int  0x80                       ;call kernel

section .data
        msg db 'Hello, world!', 0xa  ;string to be printed
        len equ $ - msg                 ;length of the string
```

High-level Language

- A language which use statements consisting of English-like keywords, such as "FOR", "PRINT", “INPUT”, “IF“, … etc.
- Each statement corresponds to several machine language instructions (one-to-many correspondence)
- Much easier to program than assembly language
- Data is referenced using descriptive names (variables)
- Operations can be described using familiar symbols (math symbols)

Programming Syntax

- Syntax is the structure of a program as well as the rules behind the structure
  - Set of legal expressions in that language

- In English, a sentence is composed of a noun + verb + noun
  - John plays chess (noun verb noun)
  - John chess plays (noun noun verb)

- Programming languages also have a syntax
  - 2 + 2 (Operand Operator Operand)
  - 2 2 + (Operand Operand Operator)

Programming Errors

- Syntax error
  - using illegal terms, expressions or rules

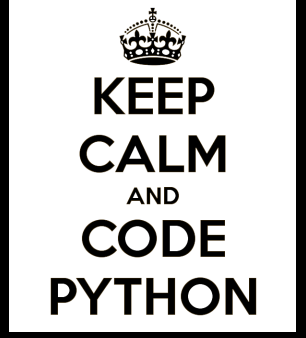
  Example: 2 2 +
- Runtime errors
  - no syntax errors, but the program can't complete execution
    - Division by zero
    - Invalid input data
- Semantic errors (logical errors)
  - program completes execution, but delivers incorrect results
    - Example: 2+4/2

The Python Programming Language

- The name Python has nothing to do with the type of Snake
- Python implementation started in 1989 by Guido Van Rossum
  - he liked Monty Python
- We use Python as a tool to learn programing

Why Python?

- Simple and easy to learn
- Very popular language
- Highly productive as compared to other programming languages like C and Java
- Clear and readable syntax
- Excellent documentation
- It's fun!



Hello World in High Level Language

C

```c
#include <stdio.h>
int main(int argc, char** argv) {
  printf("Hello  World!\n");
  return 0;
}
```

Java

```java
public class HelloWorld {
   public static void main(String[] args) {
      System.out.println("Hello  World");
   }
}
```

Python

```python
print("Hello  World!")
```

Hello World

Python program

```
# My first Python program
# is a Hello World program

print("Hello World")
```

Screen output

```
Hello World
```

Hands-On Session I:
Computer Setup

MacOS

- For this class we will be using **Python 3.X**
- You can download and install the latest version of Python from:

  https://www.python.org/downloads/

- Once you finish installing:
  - sit back and relax !!!

Windows:
Installing the Linux Subsystem

1. Search for “PowerShell” in the Start Menu
2. Right click on the PowerShell icon and choose “Run as Administrator”
3. Type the following command in the PowerShell command line. Then press the Enter key

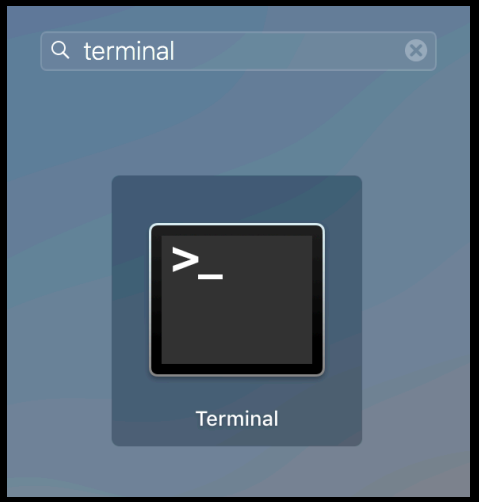
```
Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux
```

4. Press Y, then Enter to restart your computer
5. Navigate to the Windows Store and search for “Ubuntu 18.04” or “Ubuntu 18.04 LTS”
6. Click the “Get” button to download
7. After the download and installation, search for “Ubuntu 18.04” in the Start Menu
8. When asked for a password, make sure you pick an easy one you remember!!!

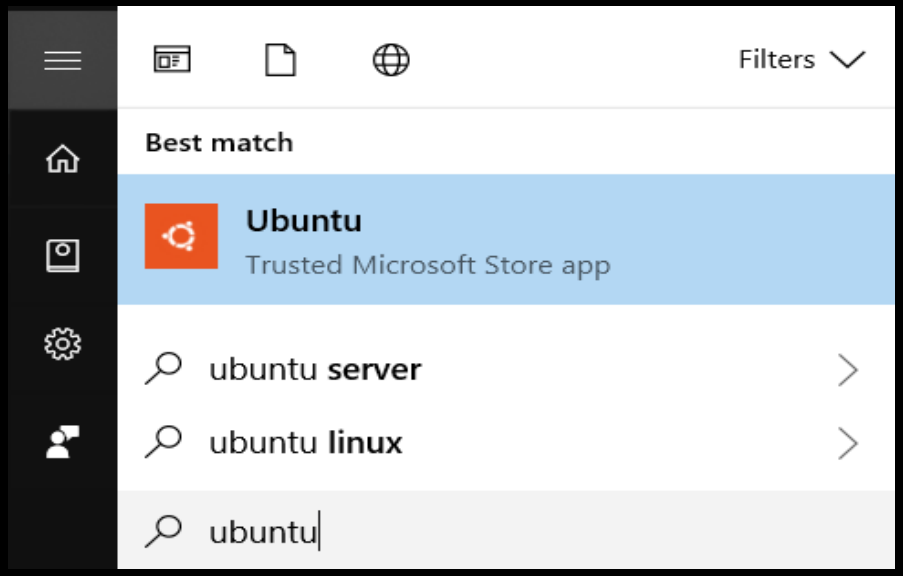
tiny.cc/introCS

Finding the Command Line

Mac OS

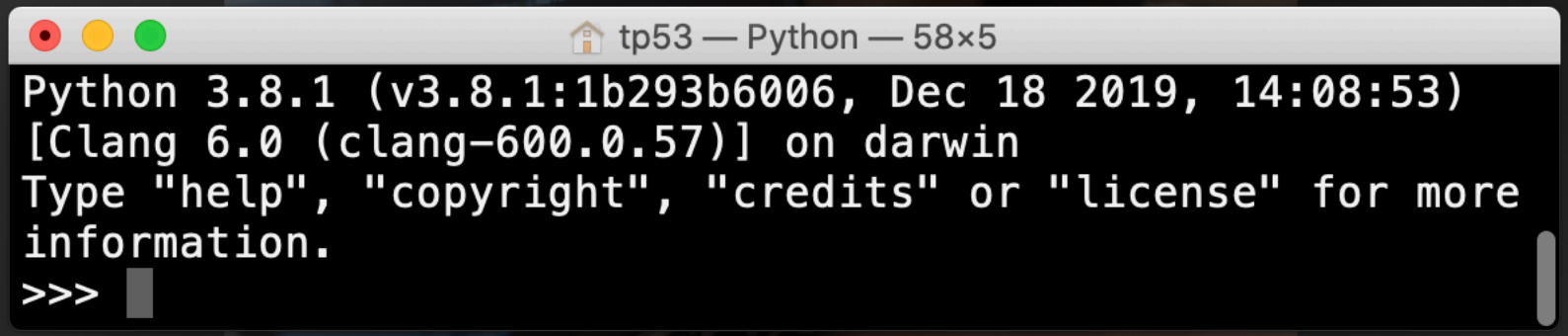


Windows



Open Terminal:

- Once terminal starts, type python3 and hit enter
  - Python interactive mode will start, and you will see a command prompt >>>



- You can try to do some math:

  >>> 3+5

  or

  >>> 20/6

Next Class

- Topics:
  - Terminal/Command Line
  - Values and Data Types
  - Operators and Operands
  - Variables