Intro to Python (Class 1)

Ben Bettisworth

1. Introduction

2. Theory

3. Python

Section 1

Introduction

Who are we?

- Ben Bettisworth
  - Computer scientist
  - I write bioinformatics tools for biologists
  - email: bbettis@ics.forth.gr
- Alexandros Pittis
  - Biologist
  - alexandros.pittis@gmail.com
- Panos Ioannidis
  - Biologist
  - pioannidis@ics.forth.gr
- Georgios Koustovoulos
  - Biologist
  - gkoutsov@ics.forth.gr

Grading

- 40% Attendance
  - 2 Free Skips
- 40% Exercises
- 20% Final project

What is not allowed

- LLMs (ChatGPT, Gemini, etc.)
- Third party libs (at first)

Useful resources

- The official docs:
  - https://docs.python.org/3/index.html

Section 2

Theory

Theory

- What is a Computer?
- What is a Program?
- What is a File?

What is a computer?

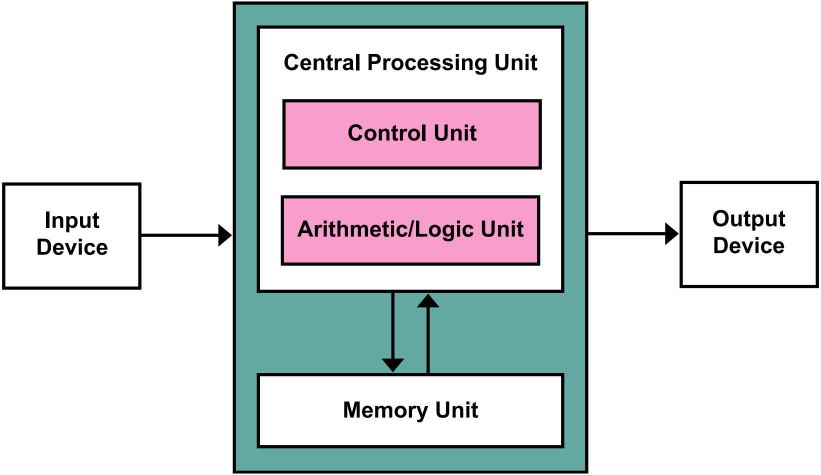
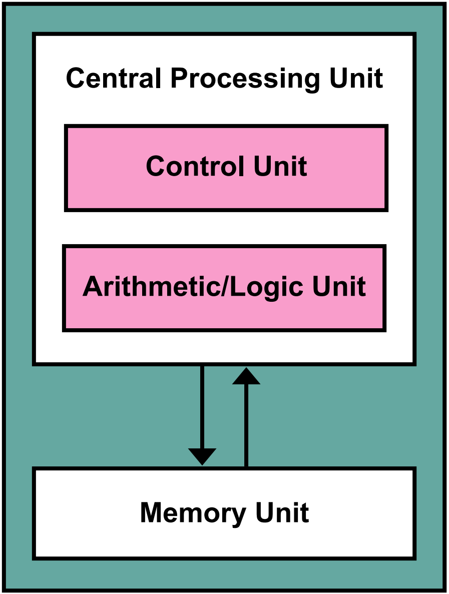
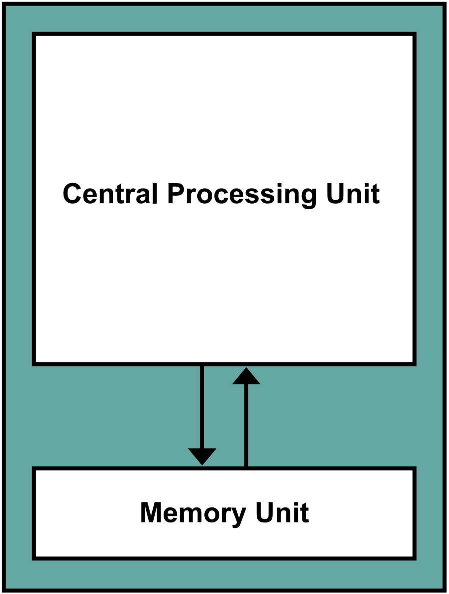


Figure 1: Schematic of a Computer (Kapooht, CC BY-SA 3.0)

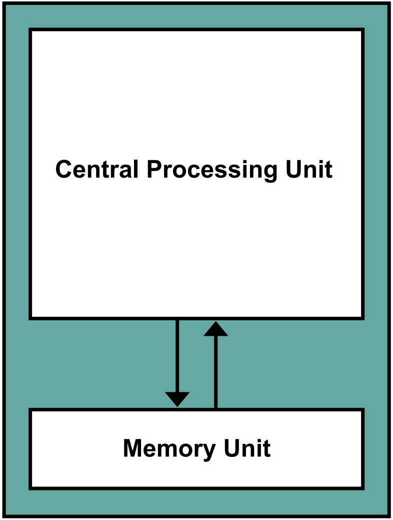
What is a computer?



What is a computer?

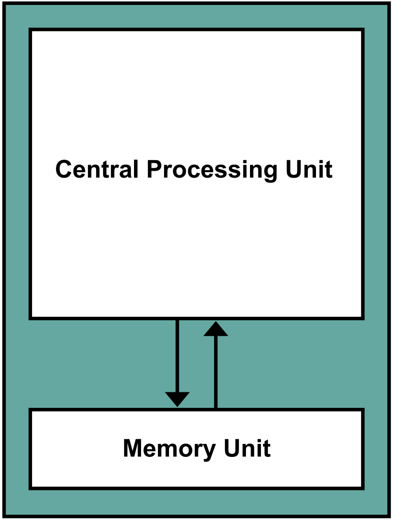


What is a computer?



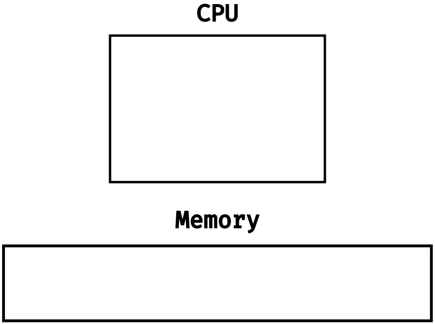
- CPU reads from memory (instructions and data)
- CPU executes instructions
- CPU writes results to memory

What is a program?



- A program is a list of instructions
- Each instruction modifies the state of the memory
- Therefore, each instruction modifies the state left by the previous instruction

What is a program? (example)



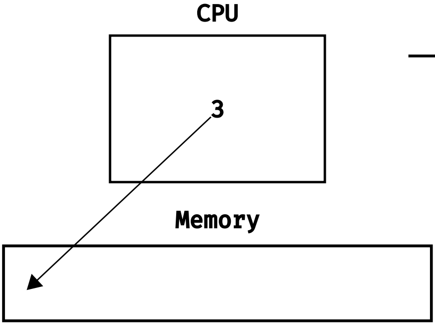
Program

- Store 3 in memory location 1

- Read value at memory location 1, and multiply it by 2

- Store result in memory location 2

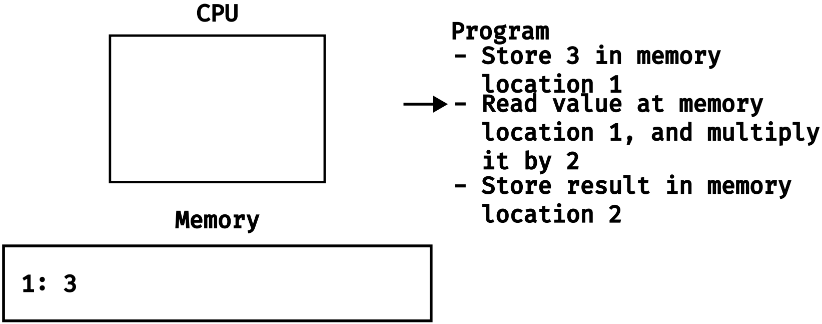
What is a program? (example)



Program

- Store 3 in memory location 1
- Read value at memory location 1, and multiply it by 2
- Store result in memory location 2

What is a program? (example)



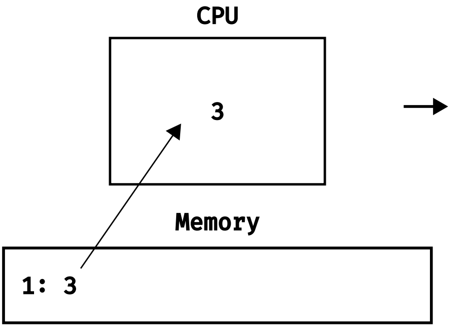
CPU

Memory

1: 3

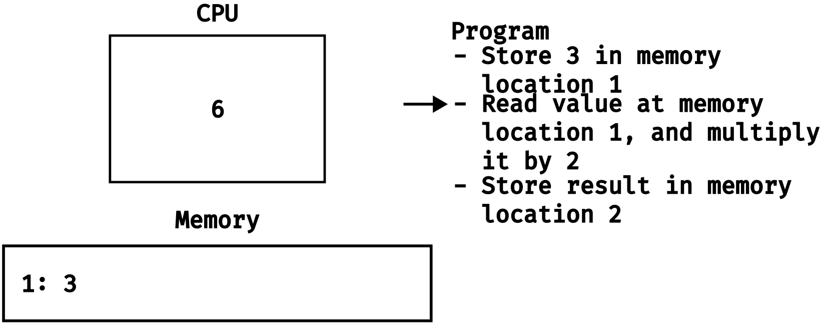
```
Program
- Store 3 in memory
  location 1
- Read value at memory
  location 1, and multiply
  it by 2
- Store result in memory
  location 2
```

What is a program? (example)



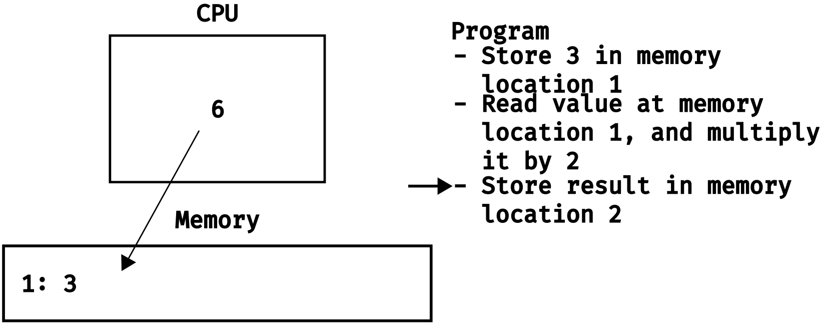
Program
- Store 3 in memory location 1
- Read value at memory location 1, and multiply it by 2
- Store result in memory location 2

What is a program? (example)



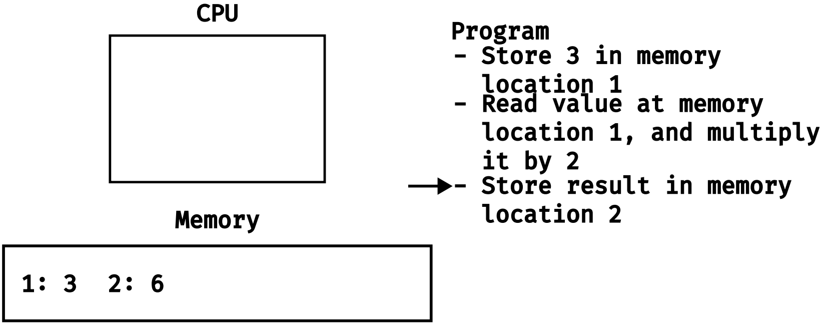
Program
- Store 3 in memory location 1
- Read value at memory location 1, and multiply it by 2
- Store result in memory location 2

What is a program? (example)



Program
- Store 3 in memory location 1
- Read value at memory location 1, and multiply it by 2
- Store result in memory location 2

What is a program? (example)



CPU

Memory

1: 3   2: 6

Program
- Store 3 in memory location 1
- Read value at memory location 1, and multiply it by 2
- Store result in memory location 2

Upshot

A program should be thought of as a series of steps which evolve the state of the computer's memory.

What is a file?

- Files are chunks of memory stored in a file system
- These files are organized into a directory structure
- A file is identified by a *unique* path.
  - A path is the list of directories plus the files name
  - e.g. `/foo/bar/a` or `C:\Users\foo\bar\a.txt`
- Files are used to store state between program runs

But what is a file?

- That previous definition is true, but practically useless.
- Imagine two programs, a and b, one outputs a file and the other reads that file.
- If it was *just* memory, then a and b would have to have the same memory layout
- In practice, files are *serialized* into some format

Section 3

Python

What is python

- Python is an *interpreted*, *dynamically typed*, *garbage-collected* language.
- *Interpreted* means that the program does not compile to a binary, but instead another program interprets the source code.
- *Dynamically typed* means that the operation applied to values is determined at runtime.
- *Garbage-collected* means that the memory allocated by the language is handled automatically by a *garbage collector*.

Interpreted Languages

```
a = 3
b = 4
c = (a + b) ** 2
print(c)
```

- Each line of the program (script) is read by the interpreter
- The interpreter translates the line into machine code, and executes it
- The interpreter continues to the next line, and repeats

Hello world

```
#!/usr/bin/env python3

print("hello world")
```