CCPS109 Computer Science I

Lecturer: Nhan Tran

n3tran@Ryerson.ca

Agenda

Introduction
Course Management Form

Lecture:

History of Computer Science

Programming Languages

Python Intro

Computers:

- Perform an input by clicking an icon, or perform some gesture
- Immediate output with display, sounds, etc…

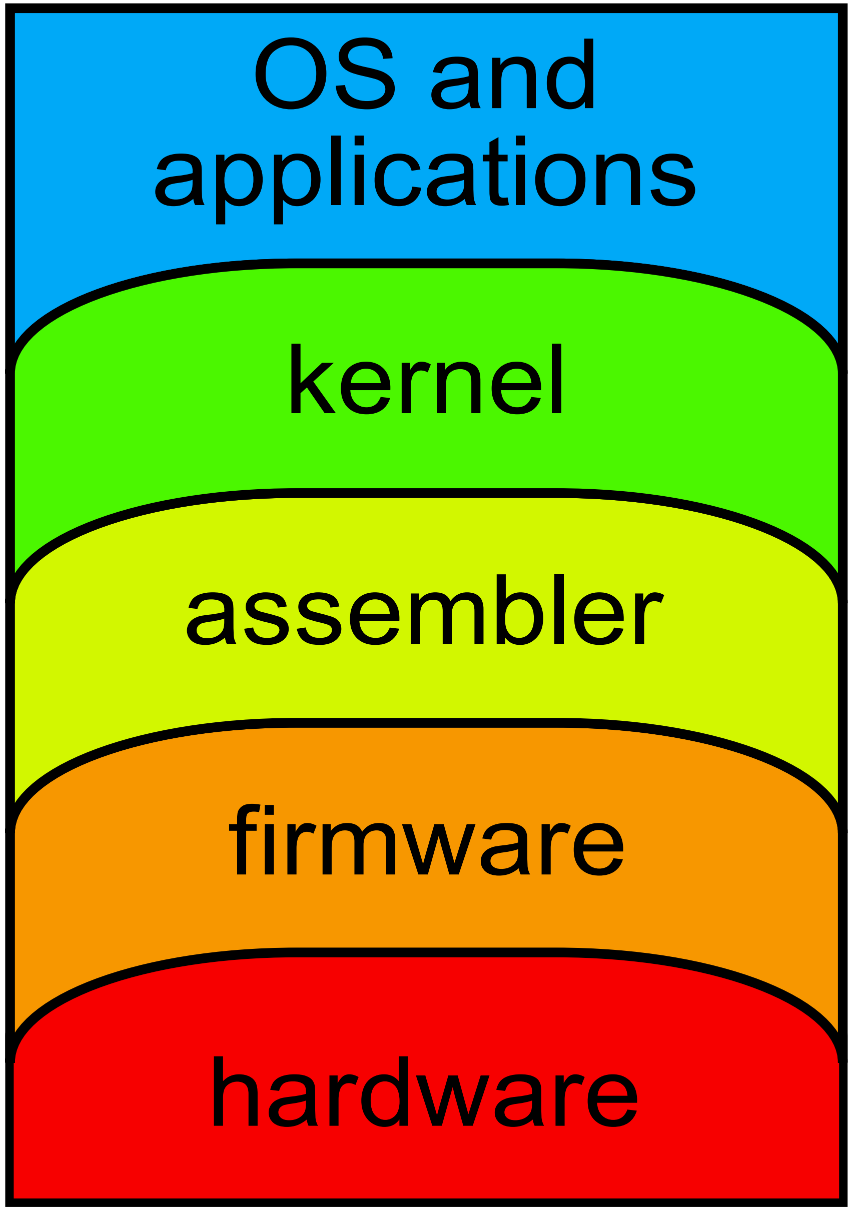
Ex: Apps, games, commands

Generally Computer :

calculate and store data

Give output (when asked)

!note: *Much more complicated in the background*



https://upload.wikimedia.org/wikipedia/commons/0/03/Computer_abstraction_layers.svg

Brief History of Computer Science

Computers were invented to replace “*Computers*”

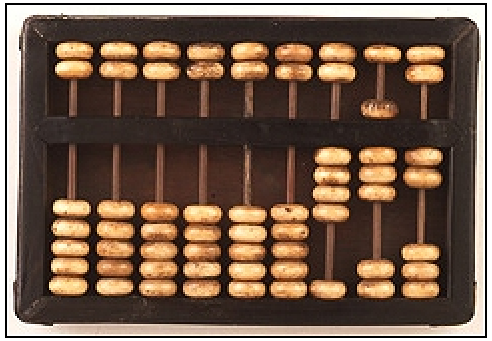
- “*Computer*” were job tittle given to human beings whose performed repetitive calculations:
  - Navigation tables
  - Tidal charts,
  - Astronomical almanacs (planetary positions, star chart)
  - Etc.

  Predominantly women – earned much less than men computers

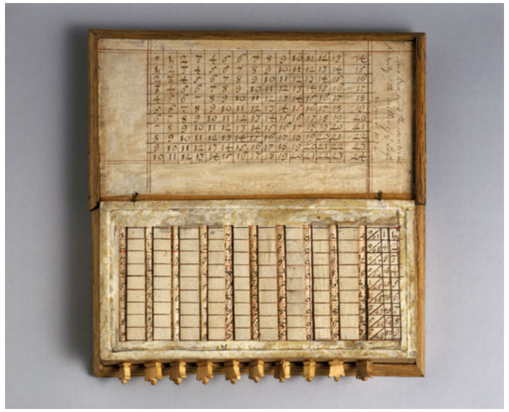
Problems: mistakes from carelessness and boredom
As data sets become bigger and need for faster solutions inventors search new ways to mechanize these tasks.

Brief History of Computer Science

- Abacus
    - Sumerian abacus(~2700BCE)
    - Earliest tool for computation
        - Picture of Chinese abacus



- Antikythera Mechanism 205-100BCE
    - First analog computer
    - Astronomical position, eclipses Olympic games cycles



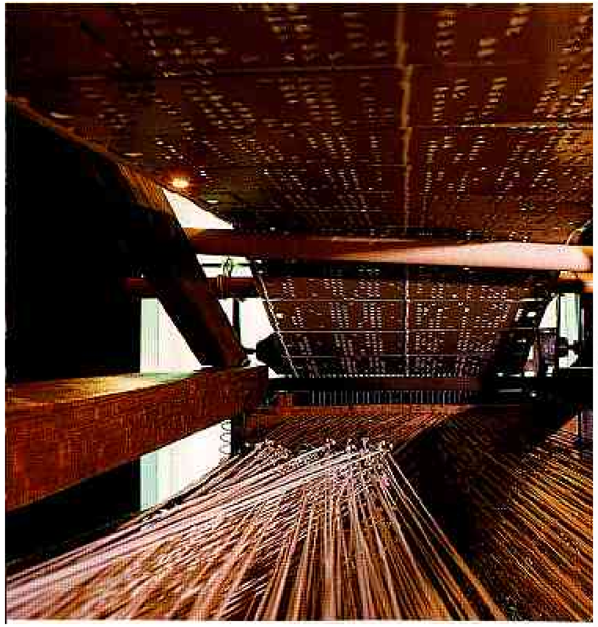
- John Napier (early 17 century)
    - Discover Logarithms
    - Napier's Bones (multiplication & division)

Brief History of Computer Science

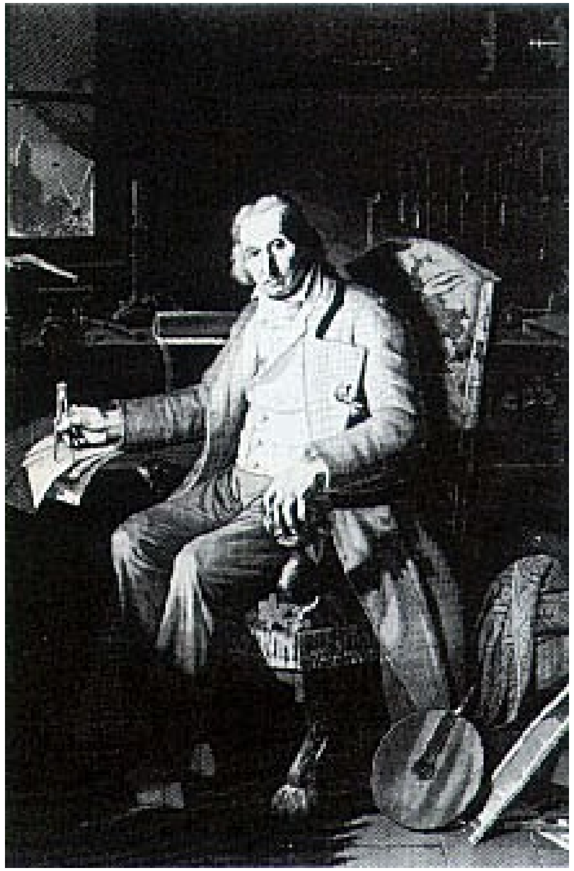
- First slide rule - 1632
  - Used in 1960’s by NASA for space programs

- Blaise Pascal 1642
  - Gear driven one function calculator
    - Pascaline
- Gottfried W. Leibnez
  - Devised modern binary system (1679)
  - Stepped reckoner- calculator (1672-1694)
  - Note: Binary system have been in use by ancient Egyptian and Chinese ,~1600BC and ~900BC respectively.
- Joseph M. Jacquard
  - Jacquard’s Loom (1801)
  - Use wooden punch cards to program fabric patterns

Brief History of Computer Science



[Jacquard-card Making.]



Jacquard portrait program using 10000 punch cards.

Brief History of Computer Science

- Charles Babbage (1882)
  - Analytic Engine
  - Patterns of presence or absence holes may be use to represent data, abstraction of problem statement.
  - Realized punched paper could be used for storage of computed intermediate result.
  - Called the two main part of his machine the “Store” and the “Mill”
    - They are equivalent to **memory** and **central processing unit (CPU)**.

- Ada Augusta Byron (became the Countess Lady Lovelace)
  - Wrote notes detailing sequences of instruction Analytic Engine
  - Invented the looping subroutine and first to recognized its important

History recognize Ada as the first computer programmer.

Brief History of Computer Science

- Herman Hollerith
  - Hollerith desk (used for 1890 U.S census)
    - Ideas adopted Jacquards’ punch card for computation.
    - Card reader, gear mechanism for counting
    - Dial indicators to display count results.
    - Use punch card for read and write (unknown of Babage’s Work)

- 1890 Census was completed in only 3 years saving 5 million dollars vs. 1880 census which took 7.5 years.
- Hollerith built the Tabulation Machine Company
  - Became IBM
  - Punch cards technology became ubiquitous: pay slip, record, bills, test (scantron) etc..

Brief History of Computer Science

- Alan Turing (1936) –Father of Computer Science
  - Construct a formal model of a computer- The Turing Machine
    - A finite state “machine” : an abstract machine that manipulates symbols based on a table of rules.
    - Such machine Store both data and instructions, scanner moves to sequentially read and write symbols
    - Insert different instruction (program) the machine carry out different kind of computation.
    - A power construct where machine’s operation is controlled by set of coded instructions stored in memory.
  - Example of a Universal Turing machine
    - https://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/turing-machine/one.html

Brief History of Computer Science

- Konrad Zuse
  - Z3 (1941)
    - first fully functional, general purpose, program-controlled computer
- Harvard Mark1
  - First programmable digital computer
    - Composed of switches, relays, rotating shafts and clutches.
    - Weight 5 tons, 500miles of wire: 8ft tall by 51ft long.
    - Grace hopper found credited with the word “debugging”

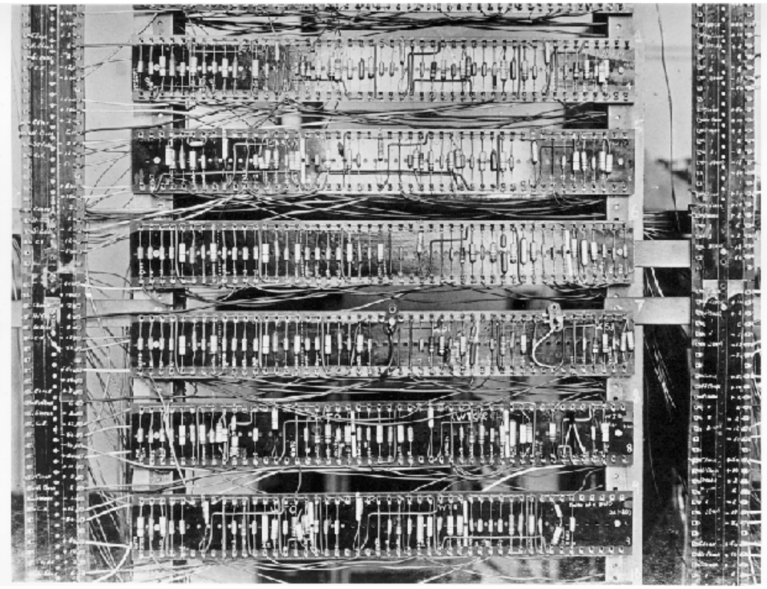
- ENIAC- Electronic Numerical Integrator and Calculator (1945)
  - 20x40ft, weight 30 tons
  - Used 18000 vacuum tubes use 174 kilowatts.

- Manchester baby (1948)
  - First electronic digital (software controlled) computer that ran store programs

Brief History of Computer Science

- Transistor (1948)
  - By John Bardeen and Walter Brattain.

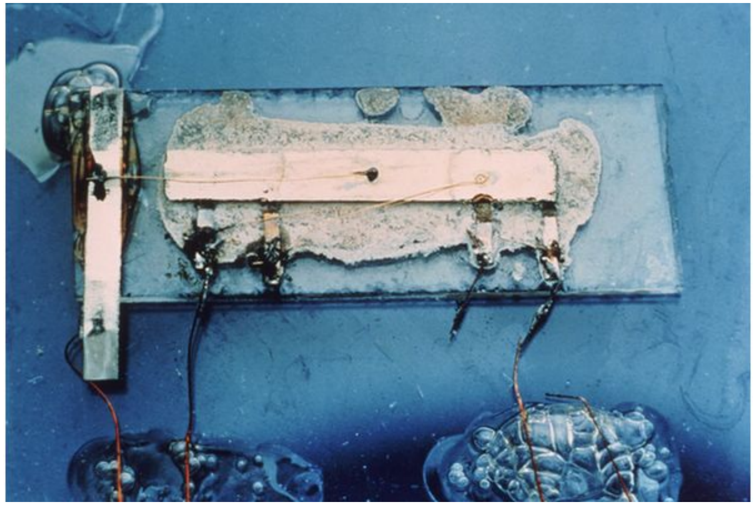
- First Transistor computer (1953)
  - University of Manchester.

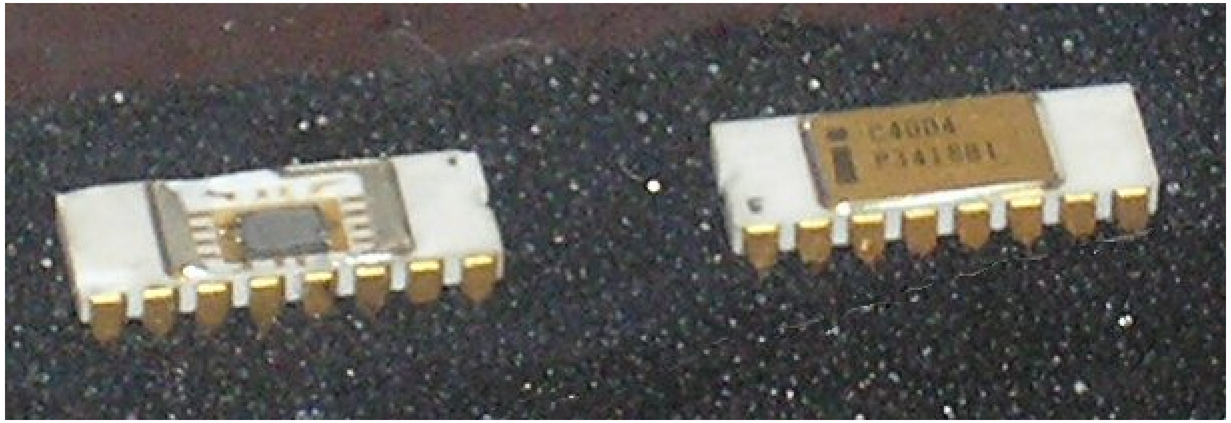


Brief History of Computer Science

- Integrated circuit
  - Successful demonstration (1958)
    - Jack Kilby

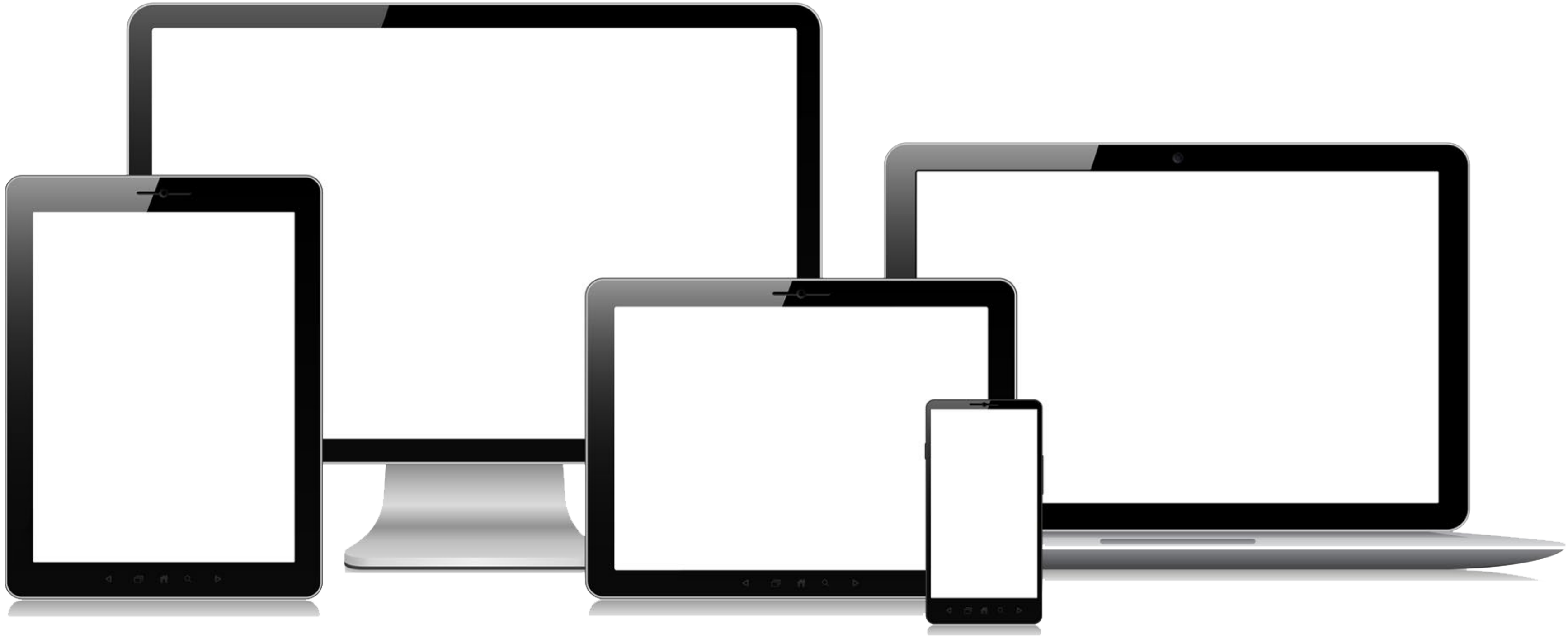
- Intel 4004 (1971)
  - Firs commercial microprocessor
  - 4- bit cpu
  - 256- byte rom
  - 40-byte ram
  - 740khz





- Modern computers
  - Ghz
  - multicore
  - Tb

TFLOPS



Modern Software

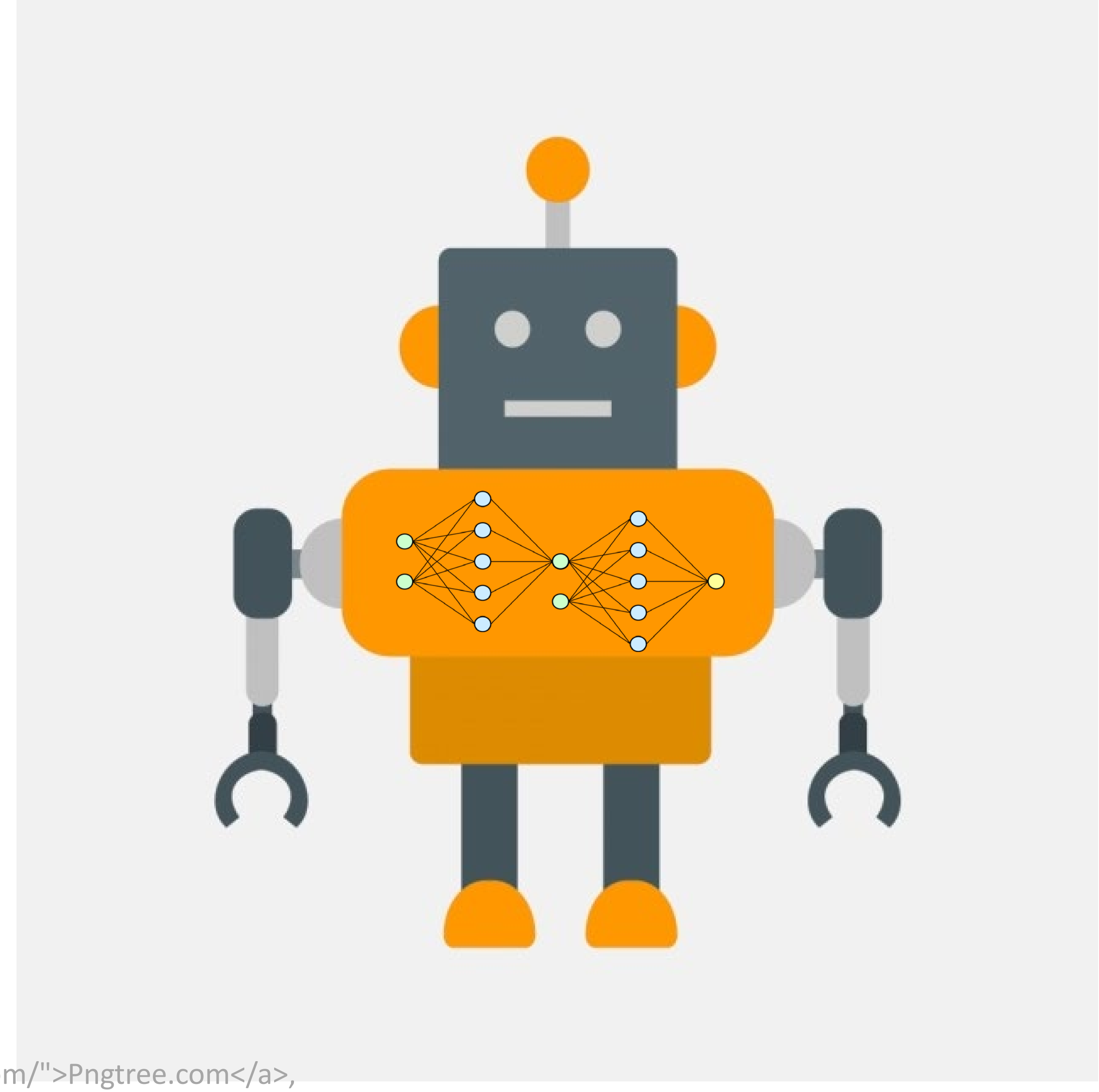


Figure: https://www.pxfuel.com/en/free-photo-qlnan,
android icon PNG Designed By IYIKON from <a href="https://pngtree.com/">Pngtree.com</a>,
https://upload.wikimedia.org/wikipedia/commons/thumb/3/3d/Neural_network.svg/800px-Neural_network.svg.png

What’s this course about?

Computational Thinking

How to solve problems via programs

Concepts and Skills that’s transferable to any programming language

Computational Thinking

Declarative knowledge

- statement of facts
- Information that can explicitly stored and retrieved

Ex: sqrt(25) is 5

How to find a sqrt?

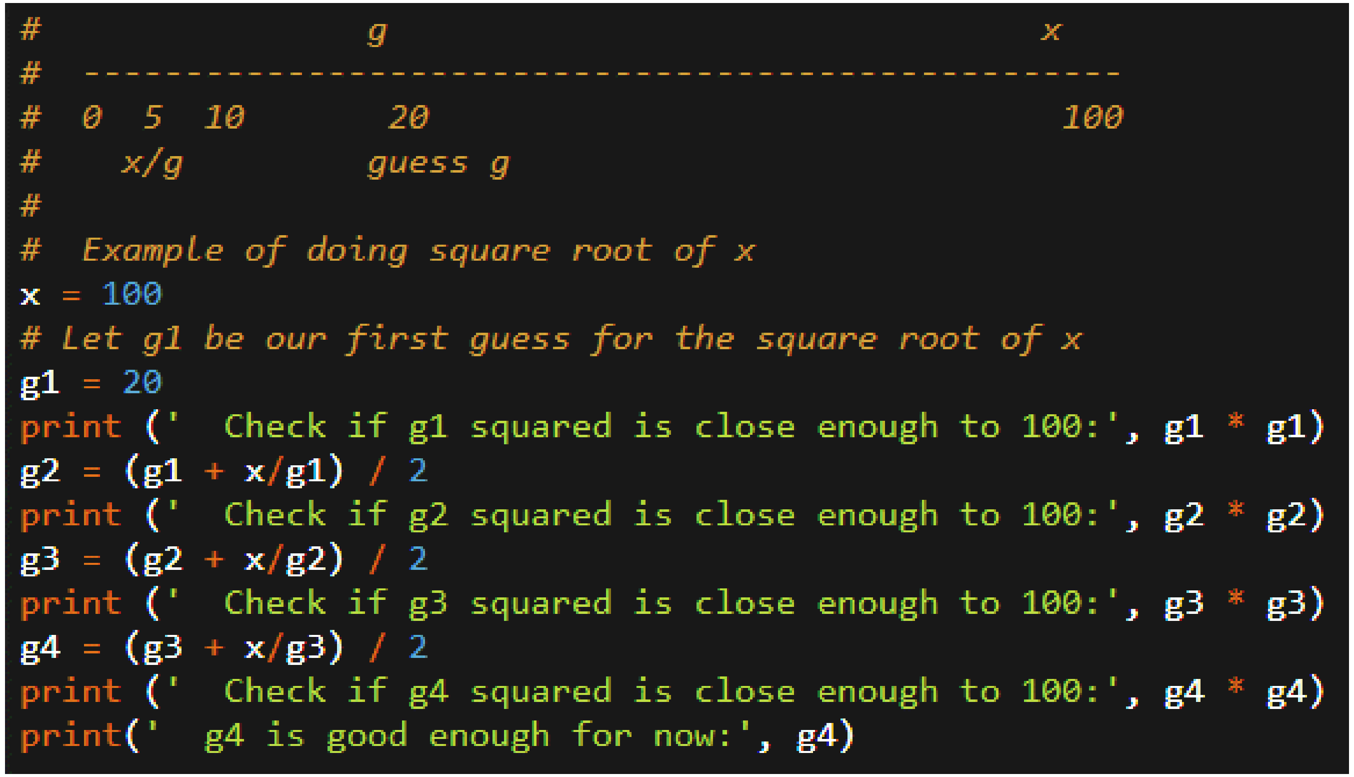
Imperative Knowledge

- how to knowledge
- Recipes for deducing information

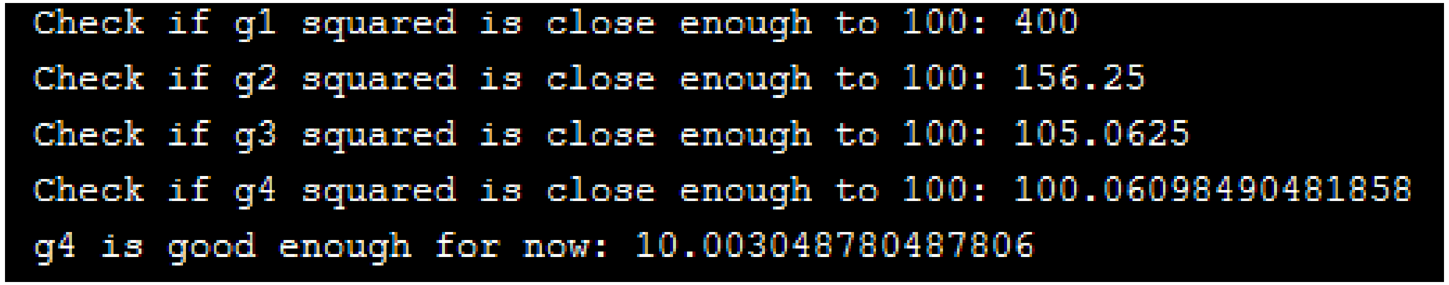
Heron of Alexandria

first to document a way to compute sqrt of a number.

Heron of Alexandria



```
#                g                                x
#  ---------------------------------------------------
#  0  5  10       20                               100
#    x/g         guess g
#
#  Example of doing square root of x
x = 100
# Let g1 be our first guess for the square root of x
g1 = 20
print ('  Check if g1 squared is close enough to 100:', g1 * g1)
g2 = (g1 + x/g1) / 2
print ('  Check if g2 squared is close enough to 100:', g2 * g2)
g3 = (g2 + x/g2) / 2
print ('  Check if g3 squared is close enough to 100:', g3 * g3)
g4 = (g3 + x/g3) / 2
print ('  Check if g4 squared is close enough to 100:', g4 * g4)
print('  g4 is good enough for now:', g4)
```



```
Check if g1 squared is close enough to 100: 400
Check if g2 squared is close enough to 100: 156.25
Check if g3 squared is close enough to 100: 105.0625
Check if g4 squared is close enough to 100: 100.06098490481858
g4 is good enough for now: 10.003048780487806
```

Algorithm

“an algorithm is a finite list of instructions that describe a computation that when executed on a set of inputs will proceed through a set of well-defined states and eventually produce an output”.

Similar to cooking recipes

Quote: Introduction to Computation and Programming using Python, with Application to Understanding Data, Second Edition, by John Guttag pg2

Programming Language:

- Church - Turing Thesis: all things computable are computable with a Turing machine
- Halting Problem: Given a program paired with an input, determine if the program will finish running or go into an infinite loop.
- Turing completeness: a programming language is said to be Turing complete if it can simulate UTM
- All modern PL's are Turing Complete
  - Anything programmed in one programming language can be programmed in any other language

Programming Language:

There are MANY (100+):

MATLAB	matrices, vectors
C/C++	networking, systems
PHP, JavaScript	web dev.
Python	general purpose prog. lang.

Programming Language (PL)

- PL can be low-level
  - assembler for hardware of machine
  - machine language#

or

- high-level -- C, Java, Python

- PL can be general
  - python, Java

or

- Application domain specific
  - SQL is for databases

Programming Language (PL)

- PL can be interpreted
  - Python: each instruction is changed to machine code as you come to it

or

- Compiled
  - like C (gcc compiles) and Java (javac)#
  - where you get machine code for your whole program

Characteristic of programming language

- Primitive constructs: words (aka keywords,variables)

- Syntax: sentences (which strings of character and symbols are well formed)

  ```
  k=j++
  ```

- Static Semantics: syntactically valid strings that have a meaning

  ```
  y=a/b
  y=x/b
  ```

Characteristic of programming language

- Semantics meaning of the sentence
    - Natural language have multiple meaning and may be ambiguous

    “I cannot praise this student too highly,”

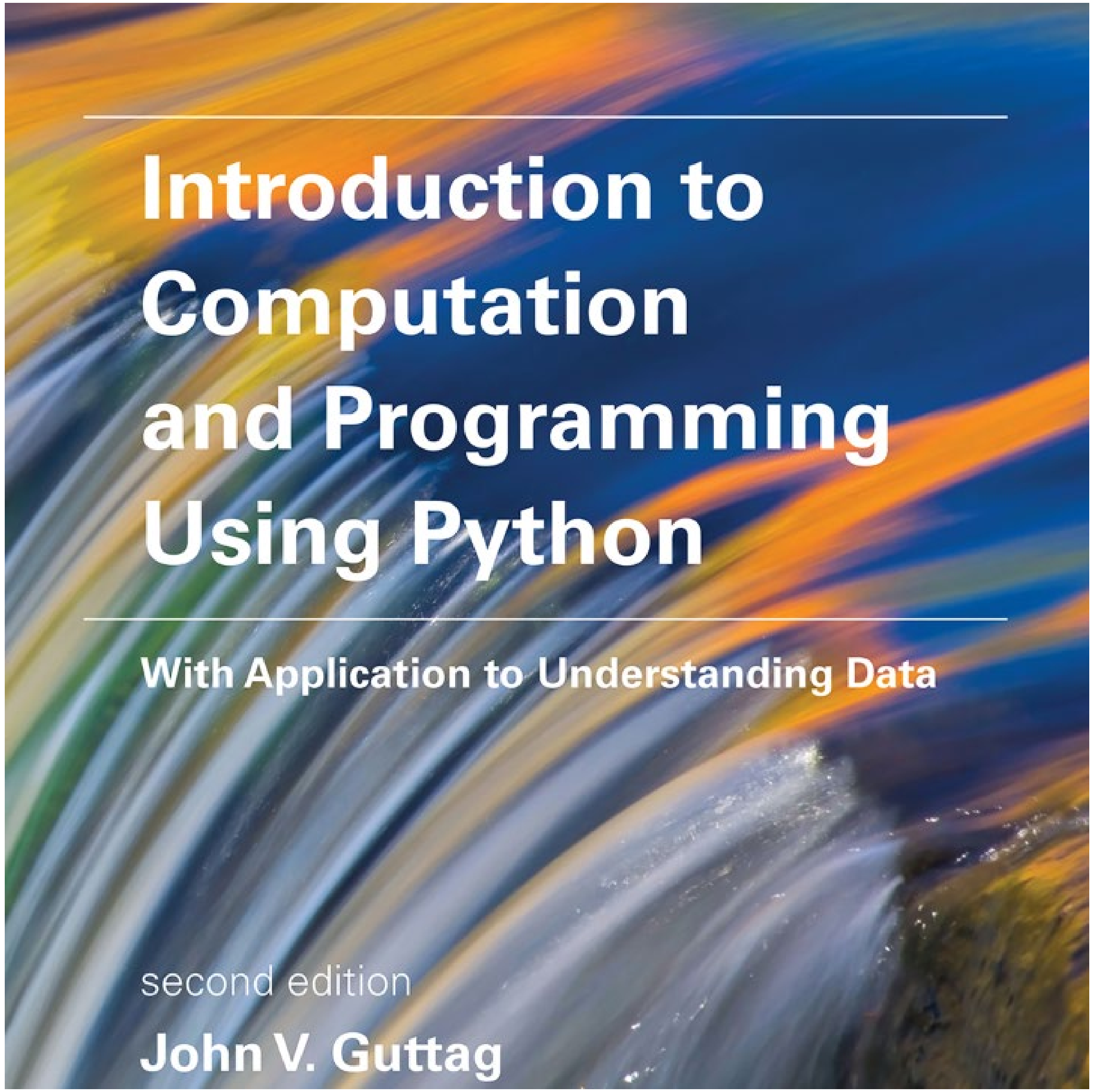
- Programming Language has only one meaning for any legal program

Possible Programming errors

- Syntax
- Static Semantics
- Run time- Possible results of run time error
  - Crash
  - freeze, never stop
  - give you the wrong answer (logic error)

Intro to Python

Textbook



Introduction to Computation and Programming Using Python

With Application to Understanding Data

second edition

John V. Guttag

Basic Element of Python

- Python program (aka script)
  - A sequence of definitions and commands

- Python Interpreter/Shell:
  - Evaluates definitions and executes commands

Commands are statements instruct the shell to do things:

```
print(‘Hello CCPS109!’)
```

Basic Element of Python

Commands are statements instruct the shell to do things:

```
print(‘Hello CCPS109!’)
print(‘hello’ , ‘ world;’ )
```

Defintion:

```
# this is a comment
def hello() :
    print('hello’)
    print('goodbye’)
hello()   #this run the function
```

Objects

- Things that python program manipulates
- Two types:
- Scalar: indivisible, atomic
  - (four kinds): int, float, bool (True, False), Nonetype(3)
- NonScalar: are composite, like strings, have internal structure
- Literals represent objects in the program
  - Text 2 🡸 a literal representing number
  - Text ‘abc’ 🡸 a string literal

Demos/ code along

Python’s Keywords

False
None
True
and
as
assert
break
class
continue
def
del
elif
else
except
finally
for
from
global
if
import
in
is
lambda
nonlocal
not
or
pass
raise
return
try
while
with
yield