Programming with C++

COMP2011: Introduction

Cecia Chan
Brian Mak
Dimitris Papadopoulos
Pedro Sander
Charles Zhang

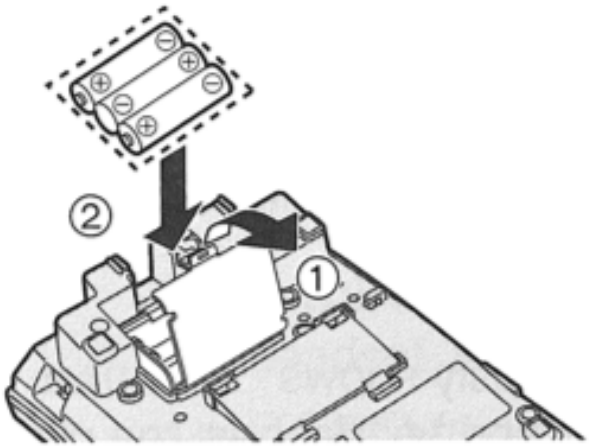
Department of Computer Science & Engineering
The Hong Kong University of Science and Technology
Hong Kong SAR, China

Course Objectives

- To learn how to solve problems by writing computer programs.
- To learn how to design a computer program.
- To learn how to program in C++.
- To learn how to debug a computer program.
- To learn object-oriented programming.
- To prepare you for COMP2012 (OOP & Data Structures), etc.

Question: *computer science* = *programming*?

Installing the Batteries



1 Press down in the direction of the arrow and open the cover (①).

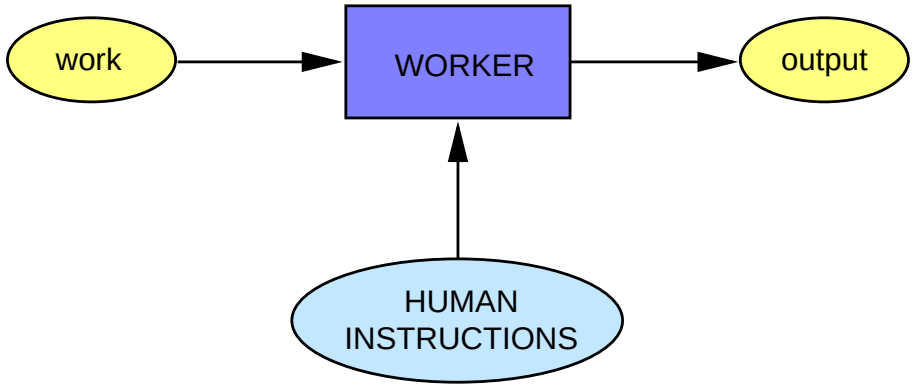
2 Install the batteries in the proper order as shown (②), matching the correct polarity.

3 Close the battery cover.

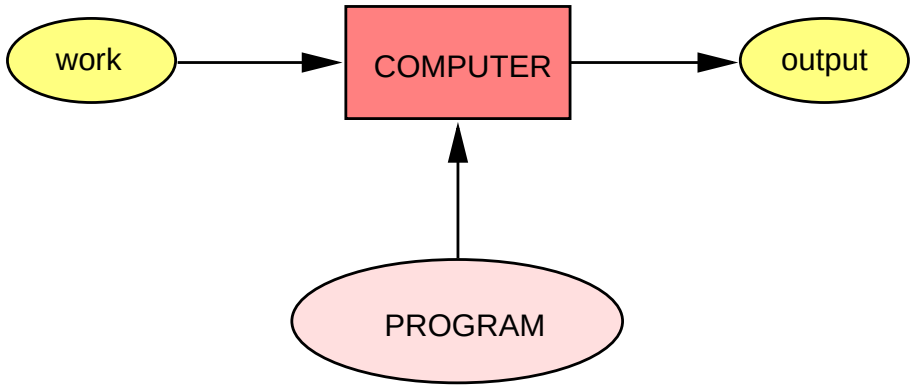
- Batteries are not included in the unit.
- Install three high quality "AA" size Alkaline (LR6) or Manganese (R6, UM-3) batteries. We recommend to use Alkaline batteries.
  Battery life is: —about six months in use of Alkaline batteries.
  —about three months in use of Manganese batteries.
  Battery life may depend on usage conditions and ambient temperature.

What's a Computer Program? ..

Human work model

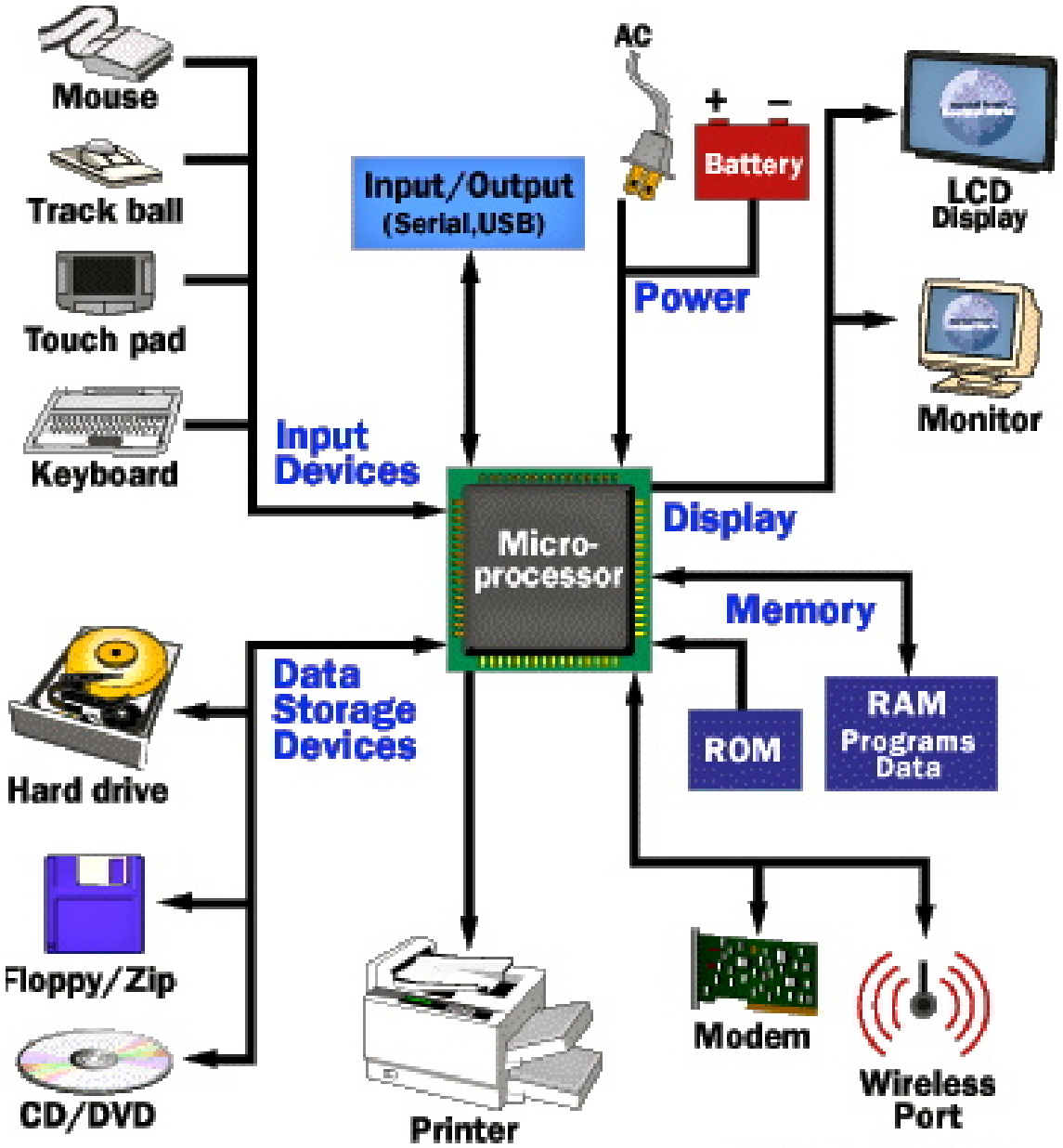


Computer work model

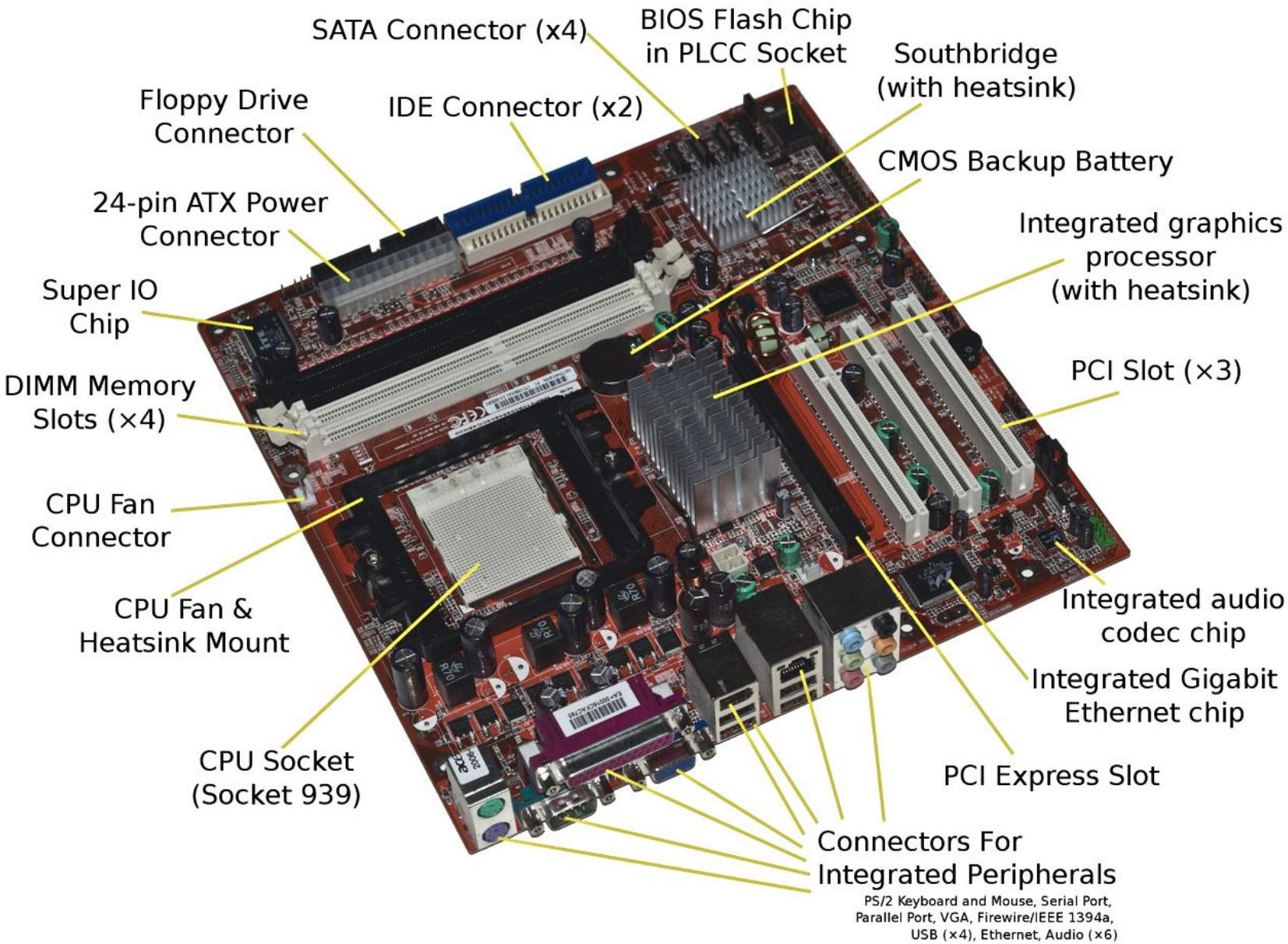


- A computer program is a set of machine-readable instructions that tells a computer how to perform a specific task. (During the execution of the program, it may interact with the users and its environment.)

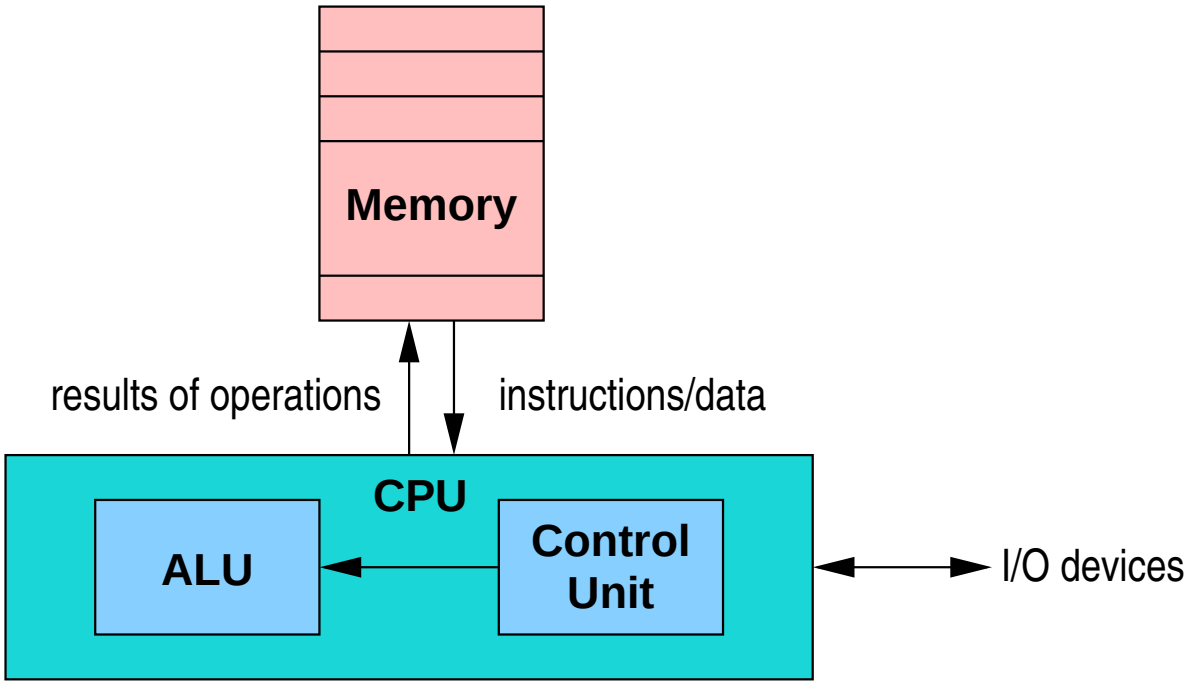
Schematic Diagram of a Personal Computer



A Typical Motherboard

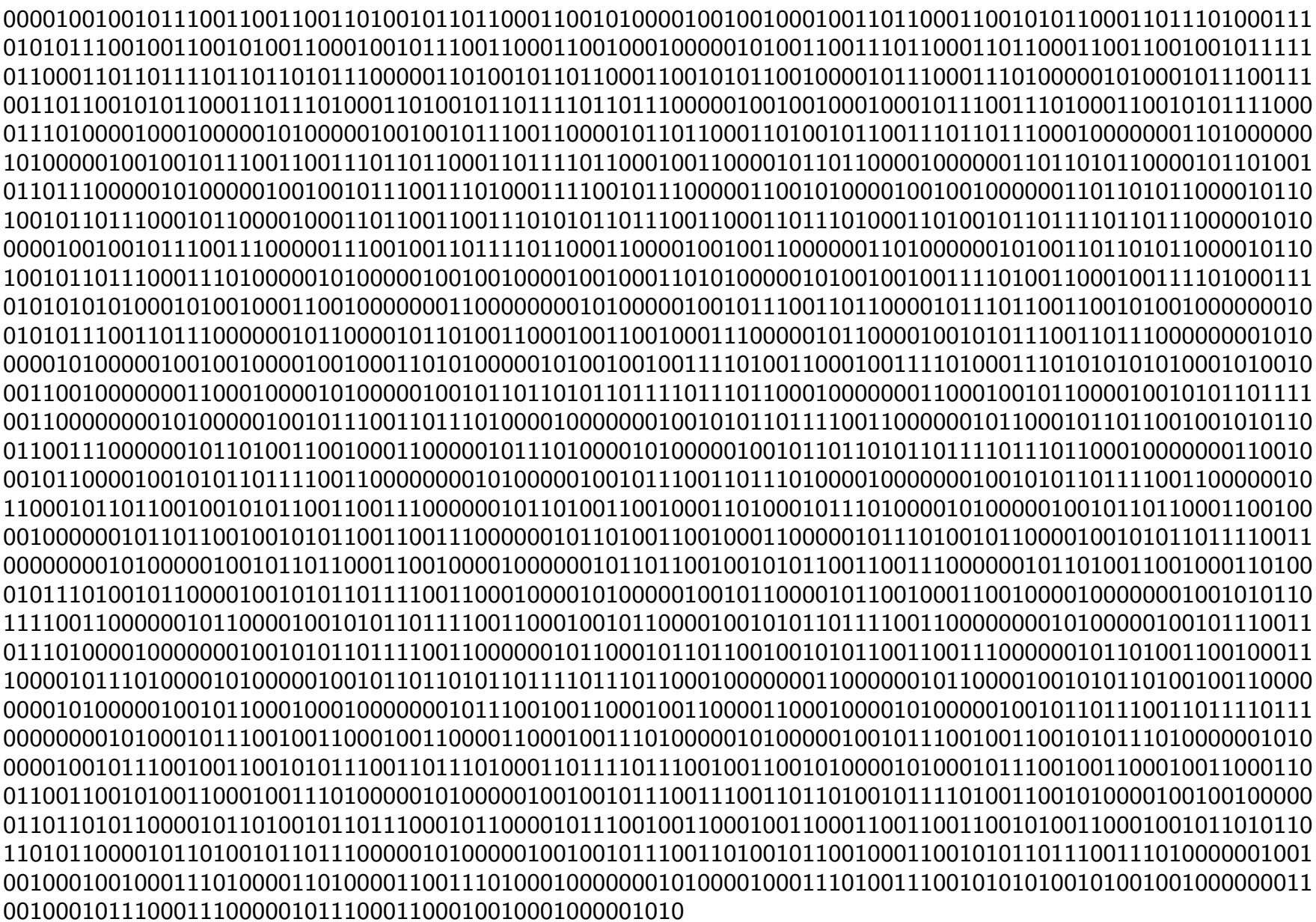


von Neumann Computer Architecture



- Designed by John von Neumann, a mathematician, in 1945.
- It is still today's dominant computer architecture.
- CPU = Central Processing Unit
- ALU = Arithmetic Logic Unit.
- For efficiency, many programming languages, including C++, are designed to take advantage of the architecture.
- More on this in COMP2611 (Computer Organization).

Can You Understand This?



How About This?

```
main:
        !#PROLOGUE# 0
        save %sp,-128,%sp

        !#PROLOGUE# 1
        mov 1,%o0
        st %o0,[%fp-20]
        mov 2,%o0
        st %o0,[%fp-24]
        ld [%fp-20],%o0
        ld [%fp-24],%o1
        add %o0,%o1,%o0
        st %o0,[%fp-28]
        mov 0,%i0
        nop
```

Is This Better Now?

```
int main( )
{
    int x, y, z;

    x = 1;
    y = 2;
    z = x+y;

    return 0;
}
```

Example: Write a Program to Sum 2 Numbers

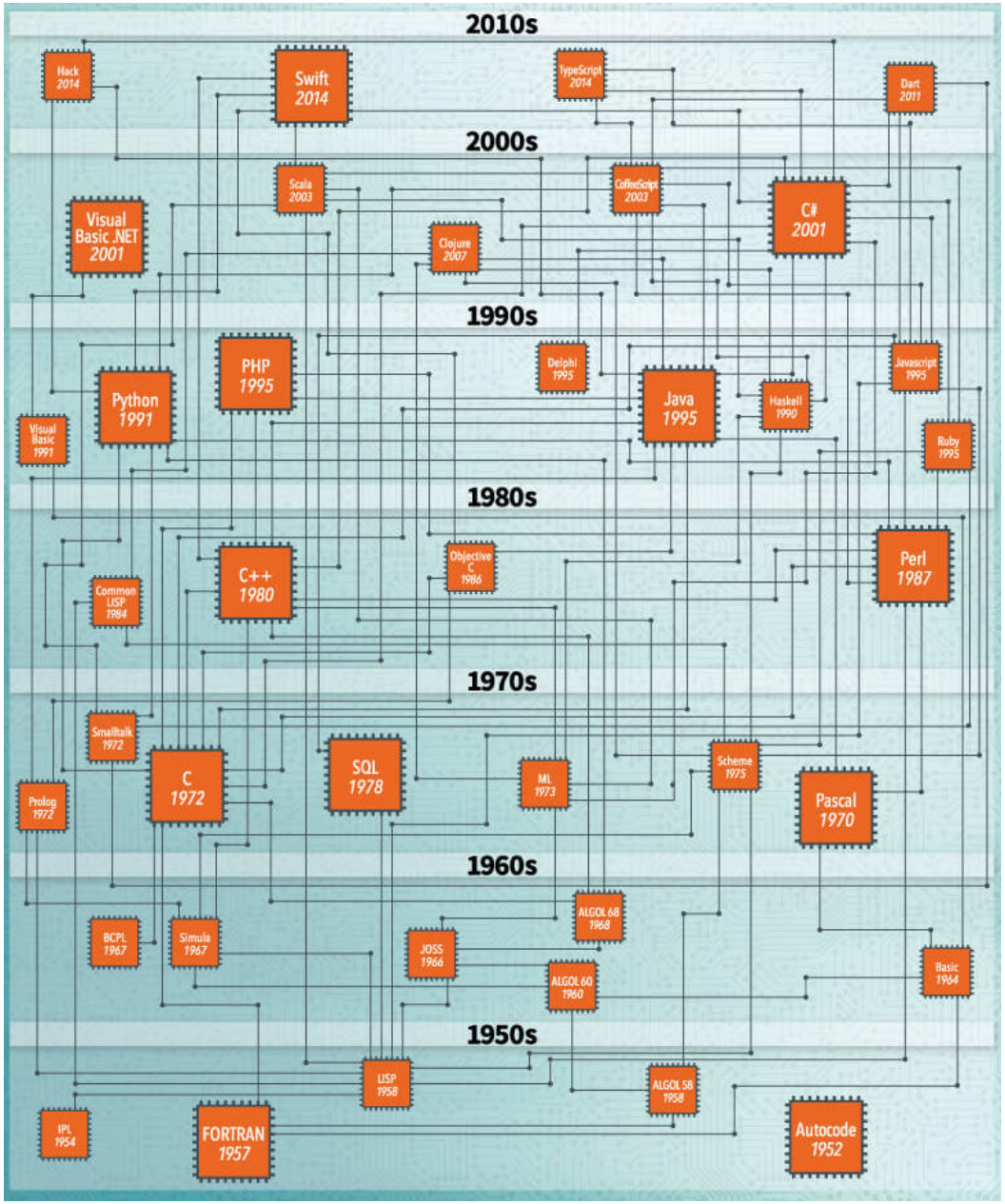
- There are 3 integer-value-holding objects: x, y, and z.
- x and y have the value of 1 and 2 respectively.
- z's value is the sum of x's and y's.

```
int main( )
{
    int x, y, z;

    x = 1;
    y = 2;
    z = x+y;

    return 0;
}
```

Levels of Programming Languages

- machine (binary) language is unintelligible
- assembly language is low level
  - mnemonic names for machine operations
  - explicit manipulation of memory addresses/contents
  - machine-dependent
- high level language
  - readable
    - instructions are easy to remember
    - faster coding
    - less error-prone (fewer bugs?)
    - easier to maintain
  - no mention of memory locations
  - machine-independent = portable

Chronology of Some Programming Languages



More details in this computer history website, and the following infographics.

PYPL PopularitY: Most Popular Programming Languages

Worldwide, Aug 2022 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Python	28.11 %	-2.6 %
2		Java	17.35 %	-0.9 %
3		JavaScript	9.48 %	+0.2 %
4		C#	7.08 %	+0.1 %
5		C/C++	6.19 %	-0.3 %
6		PHP	5.47 %	-0.8 %
7		R	4.35 %	+0.6 %
8	↑↑	TypeScript	2.79 %	+1.1 %
9	↑↑	Swift	2.09 %	+0.5 %
10	↓↓	Objective-C	2.03 %	+0.2 %

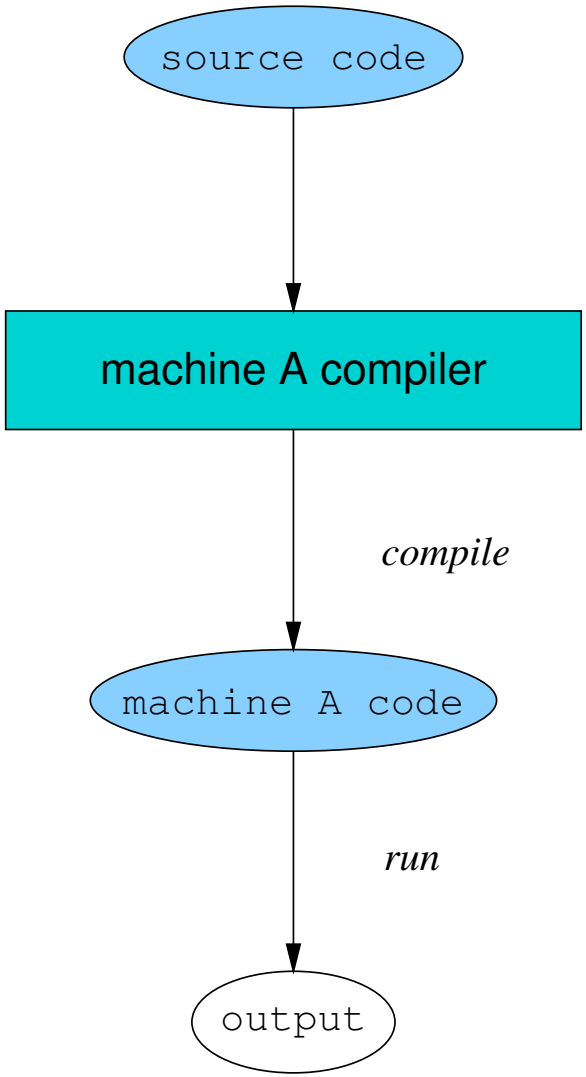
It is based on the number of Google searches on the languages' tutorials.

Mostly Used Programming Languages in Github

Year: 2022 | Quarter: 1

# Ranking	Programming Language	Percentage (YoY Change)	YoY Trend
1	Python	16.689% (+0.061%)	^
2	JavaScript	14.270% (-4.486%)	v
3	Java	13.075% (+1.394%)	
4	TypeScript	9.105% (+2.501%)	^
5	Go	7.885% (+0.056%)	v
6	C++	6.950% (-0.035%)	
7	Ruby	6.179% (-1.408%)	v
8	PHP	5.260% (+0.179%)	
9	C#	3.059% (-0.555%)	
10	C	3.023% (-0.230%)	

Compilation: From Source to Runnable Program



A compiler translates source programs into machine codes that run directly on the target computer.

For example,
a.cpp $\longrightarrow$ a.out (or a.exe).

Some C++ compilers:
gcc/g++, VC++.

- static codes
- compile once, run many
- optimized codes
  $\Rightarrow$ more efficient
- examples: FORTRAN, Pascal, C++

Programming as Problem Solving

- Understand and define the problem clearly.
  - What are the input(s) and output(s)?
  - Any constraints?
  - Which information is essential?
- Develop a solution.
  - Construct an algorithm.
- Translate the algorithm into a C++ program.
- Compile the program.
- Test the program.
- Debug the program.
- Document the program as you write the program.
- Maintain the program
  - modify the codes when conditions change.
  - enhance the codes to improve the solution.

Summary

- Why C++?
  Read the FAQ from the designer of C++, Bjarne Stroustrup.
- Which C++?
  - The language has been evolving:
    C++ 1983 $\Rightarrow$ C++ 1998 $\Rightarrow$ C++ 2003 $\Rightarrow$ C++ 2011 $\Rightarrow \cdots$
  - We will learn C++11 (but not all the new features).
- Which compiler?
  GNU gcc/g++. It is free.
  (The compiler you will use in CSE lab is C++11-compliant.)
- Which IDE (integrated development environment) for writing programs?
  VS Code. It is free and supported by many operating systems such as Windows, Mac OS, and Linux. Other freeware like Eclipse is also fine.