Introduction to Computer

Heeseung Jo

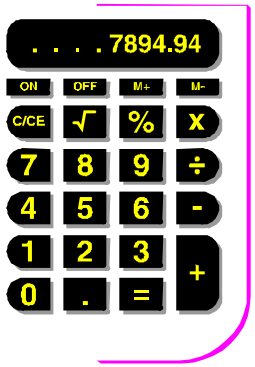
Overview of Discussion

What is computer science?

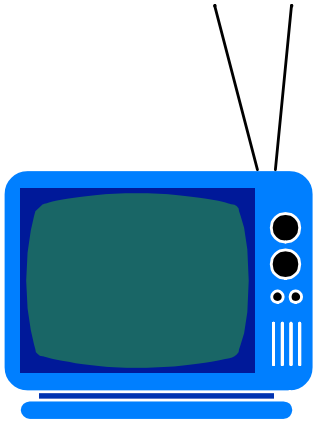
- What is a computer?
- What can computers do?
- How do computers solve problems?

Which one is the computer?

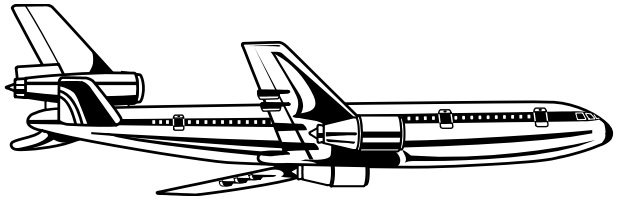
Rock



Calculator



Television



Modern Airplane

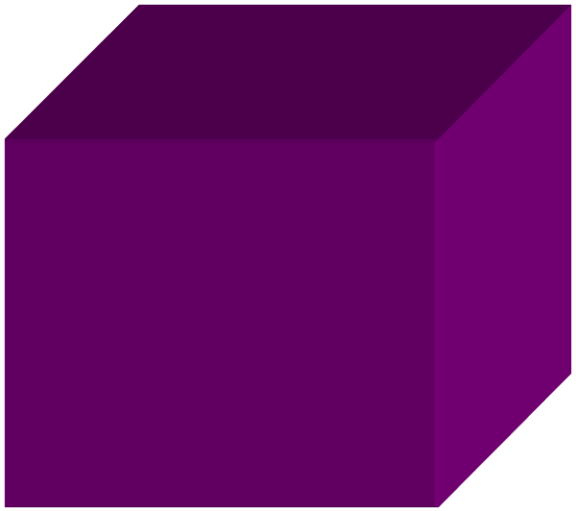
Washing Machine

Computer Workstation

Is it a Computer?

What questions would you ask?

What experiments would you run?



Is a rock a computer?

Computers must be able to handle input and output

Does not act or process

Takes no input and
produces no output

Is a washing machine a computer?

Computers input and output information

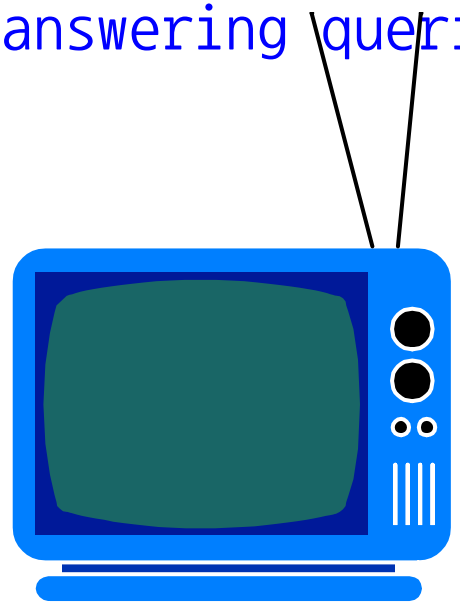
Input: dirty clothes

Output: clean clothes

Does not handle information

Is a television set a computer?

Computers process information by computing new results and answering queries



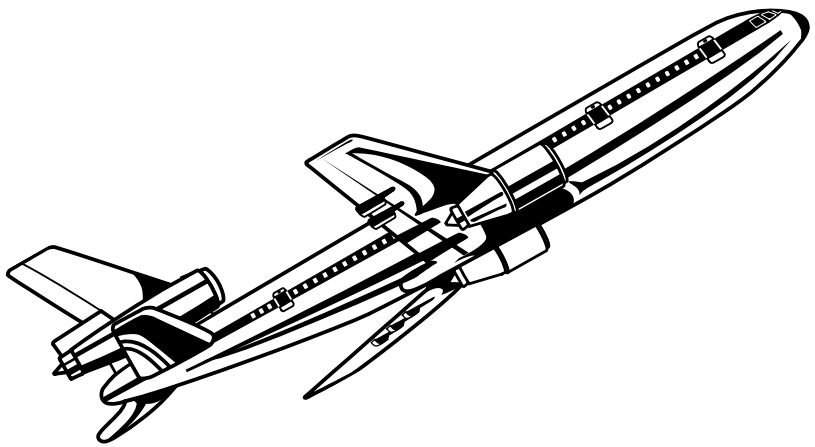
Input: information from cables or radio waves

Output: information as sound and picture

Does not process information

Is a modern airplane a computer?

Computers are general purpose because they can perform many different tasks



Input: information from radio waves

Output: manipulations to the airplane

Can only handle specific information necessary for flight control

Is an ordinary calculator a computer?

Computers are programmable so they can remember sequences of operations

Input: numbers and mathematical operations

Output: answer

Handles any numeric task

Cannot remember which buttons are pressed

Definition of a Computer

a general purpose,

programmable,

information processor

with input and output

What can computers do - today?

Business productivity managers

Personal information managers

Spreadsheets

Database software

Desktop publishing

Multimedia encyclopedias

Simulate the physical world

Produce a music video

What might computers do - tomorrow?

Diagnose diseases

- MYCIN captures medical knowledge in rules that allowed a computer to identify an ailment based on symptoms

Control robots that walk, talk, and learn

- CMU created a program that drove a van from Pittsburgh to D.C. using cameras for eyes

Compose music and create art

How do computers solve problems?

Humans deconstruct problems into small operations that a computer can carry out

- Writing an algorithm

Solve a problem by computer requires

- State the problem clearly in a problem statement
- Solve the problem with an algorithm that gives clear instructions
- Use a computer to carry out the instructions

Stating the problem clearly

Describes what to do, not how to do it

- How do I get from Seoul to Busan?

Solve general classes of problems

- How do I get from point A in Seoul to point B?
- What is the square root of y?

Specifying a problem

Clear problem statement is called the specification

- What information can we use as input
- What the output, or solution, to our problem should look like

Specification for the square root problem

- Input: A positive number $y > 0$
- Output: A positive number x such that $x^2 = y$

Make sure specification is not ambiguous

Solving the problem using an Algorithm

Algorithm - a clear sequence of instructions for performing a task

- a well-ordered sequence
- of well-defined,
- feasible operations
- that takes finite time to carry out

Almost Algorithms

To shampoo your hair

1. Rinse
2. Lather
3. Repeat

To set the time on the VCR

1. Open the front panel
2. Push the button
3. Set the hours, then the minutes

To write the Great American Novel

1. Get paper and pencil
2. Sit down
3. Write word on paper
4. If novel is great, quit. Otherwise, go back to step 3.

Necessity of artificial languages

Problems with natural languages (like English)

- Flexible
- Often ambiguous

Computers use artificial languages with precise meanings

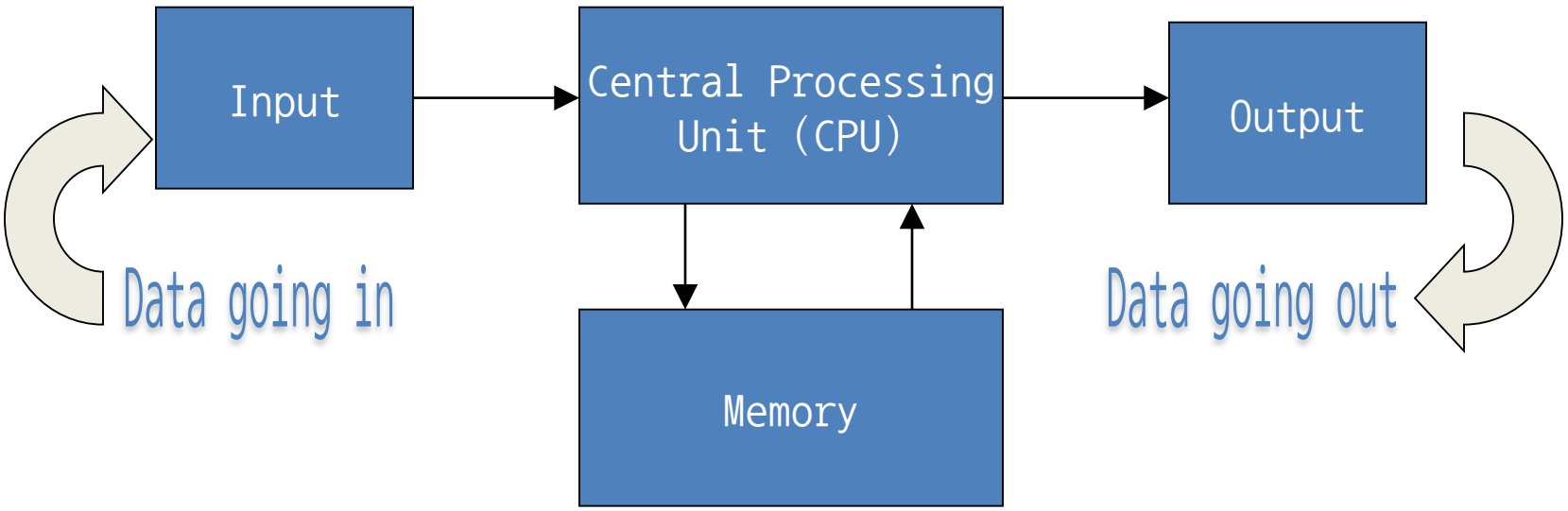
- mathematical equations, music notation, programming languages

Programming languages define primitive operations computing agents understand

von Neumann Machine

Store programs in electronic memory along side the data (1943)

- Move and manipulate a program like data
- Enabled high-level programming languages



Machine Languages

Only language computers directly understand

- "Natural language" of computer

Defined by hardware design

- Machine-dependent

Generally consist of strings of numbers

- Ultimately 0s and 1s

Instruct computers to perform elementary operations

- One at a time

Cumbersome for humans

Example:

```
+1300042774
+1400593419
+1200274027
```

Assembly Languages

English-like abbreviations representing elementary computer operations

Clearer to humans

Incomprehensible to computers

- Translator programs (assemblers)
  - Convert to machine language

Example:

- LOAD BASEPAY
  ADD OVERPAY
  STORE GROSSPAY

High-level Languages

Similar to everyday English, use common mathematical notations

Single statements accomplish substantial tasks

- Assembly language requires many instructions to accomplish simple tasks

Translator programs (compilers)

- Convert to assembly language

Interpreter programs

- Directly execute high-level language programs

Example:

- grossPay = basePay + overTimePay

Programming Approaches

Structured programming (1960s)

- Disciplined approach to writing programs
- Clear, easy to test and debug, and easy to modify
- Focus on what the program does

Object Oriented programming

- Object is an entity characterized by a state and a behavior
  - state is encoded in the computer program as data
  - behavior is encoded as methods

Basics of a Typical C Environment

C systems

- Program-development environment
- Language
- C Standard Library

Basics of a Typical C Environment

Phases of C Programs:

1. Edit
2. Preprocess
3. Compile
4. Link
5. Load
6. Execute