LECTURER: TAI LE QUY

# INTRODUCTION TO COMPUTER SCIENCE

INTRODUCTORY ROUND

# Who am I?

- Name: Tai Le Quy
- PhD candidate at L3S Research Center – Leibniz University Hannover
  - Topic: Fairness-aware machine learning in educational data mining
  - Project: LernMINT (lernmint.org)
- MSc in Information Technology at National University of Vietnam
- Profile: tailequy.github.io
- Email: tai.le-quy@iu.org
- Additional materials: https://github.com/tailequy/IU-IntroductionComputerScience

**INTRODUCTORY ROUND**

# Who are you?

- Name
- Employer
- Position/responsibilities
- Fun Fact
- Previous knowledge? Expectations?

TOPIC OUTLINETOPIC OUTLINE

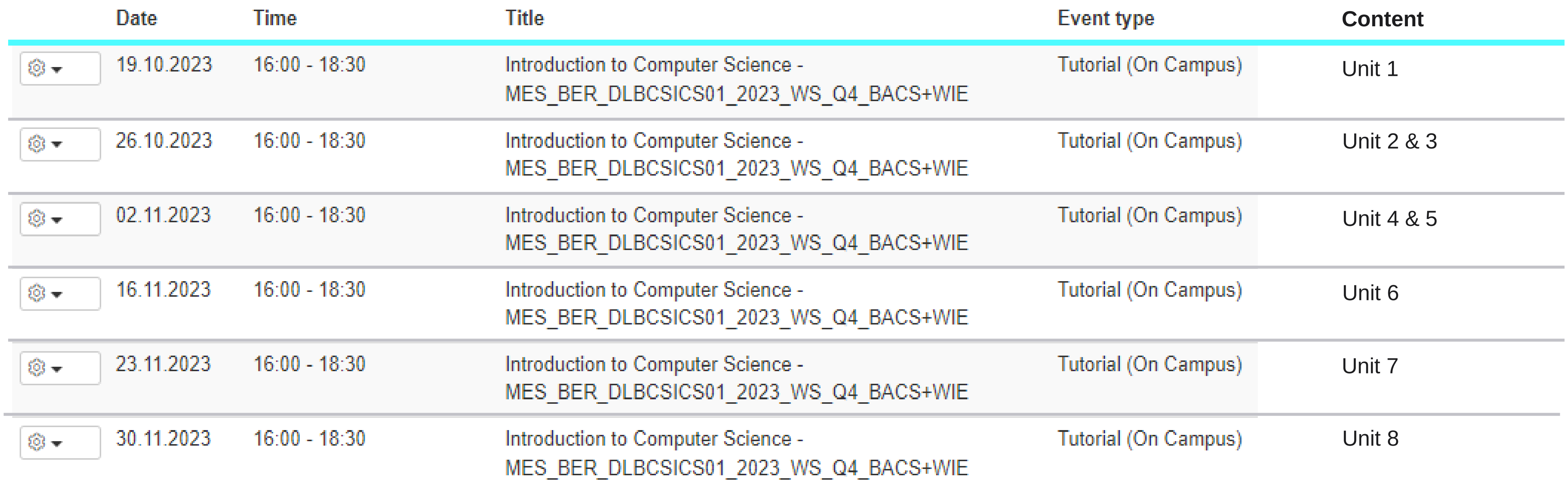
| | |
|---|---|
| Basic Concepts of Data Processing | 1 |
| Information Representation | 2 |
| Algorithms and Data Structures | 3 |
| Propositional Logic, Boolean Algebra and Circuit Design | 4 |
| Hardware and Computer Architectures | 5 |

INTRODUCTION TO COMPUTER SCIENCE
TOPIC OUTLINE

| | |
|---|---|
| **Networks and the Internet** | **6** |
| **Software** | **7** |
| **Computer Science as a Discipline** | **8** |

## TUTORING SCHEDULE

- 6 weeks (sections)
- Buildings: Berlin Frankfurter Allee
- Rooms: BER 2.12 Spandau (Eingang 73 A)



| | Date | Time | Title | Event type | Content |
|---|---|---|---|---|---|
| ⚙▾ | 19.10.2023 | 16:00 - 18:30 | Introduction to Computer Science - MES_BER_DLBCSICS01_2023_WS_Q4_BACS+WIE | Tutorial (On Campus) | Unit 1 |
| ⚙▾ | 26.10.2023 | 16:00 - 18:30 | Introduction to Computer Science - MES_BER_DLBCSICS01_2023_WS_Q4_BACS+WIE | Tutorial (On Campus) | Unit 2 & 3 |
| ⚙▾ | 02.11.2023 | 16:00 - 18:30 | Introduction to Computer Science - MES_BER_DLBCSICS01_2023_WS_Q4_BACS+WIE | Tutorial (On Campus) | Unit 4 & 5 |
| ⚙▾ | 16.11.2023 | 16:00 - 18:30 | Introduction to Computer Science - MES_BER_DLBCSICS01_2023_WS_Q4_BACS+WIE | Tutorial (On Campus) | Unit 6 |
| ⚙▾ | 23.11.2023 | 16:00 - 18:30 | Introduction to Computer Science - MES_BER_DLBCSICS01_2023_WS_Q4_BACS+WIE | Tutorial (On Campus) | Unit 7 |
| ⚙▾ | 30.11.2023 | 16:00 - 18:30 | Introduction to Computer Science - MES_BER_DLBCSICS01_2023_WS_Q4_BACS+WIE | Tutorial (On Campus) | Unit 8 |

UNIT 1

# BASIC CONCEPTS OF DATA PROCESSING

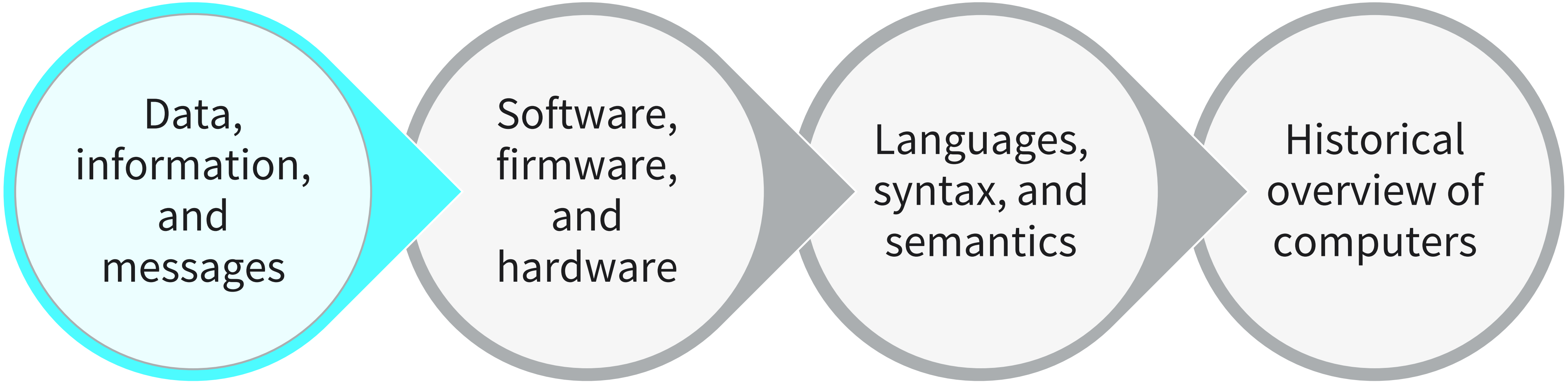
## STUDY GOALS

- Understand the **concepts** of data, information, and computer messaging.
- Learn about the **difference** between hardware, firmware, and software.
- Know the **basics** of binary and data interpretation.
- Understand what syntax and semantics refer to in **programming languages.**
- Know the history of **computers and data processing**.

**EXPLAIN SIMPLY**

1. What is the syntax of a language?
2. What is the concept of a message?
3. Why do computers need an operating system?

# BASIC CONCEPTS OF DATA PROCESSING



Data, information, and messages

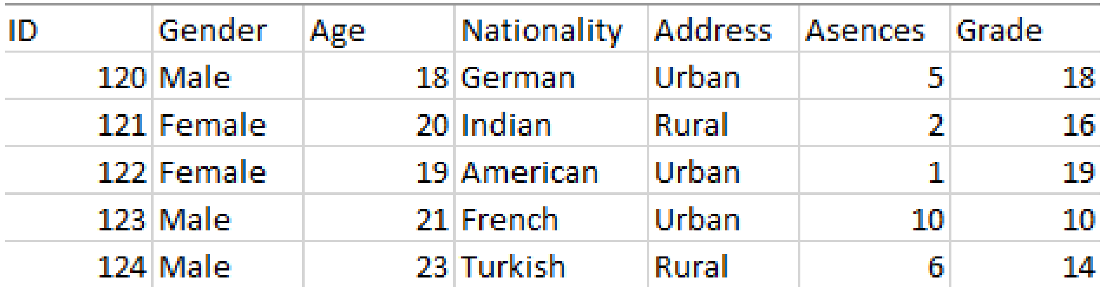
Software, firmware, and hardware

Languages, syntax, and semantics

Historical overview of computers

## WHAT IS THE DIFFERENCE BETWEEN DATA AND INFORMATION?

- **Data** in its most basic, standalone digital format **does not provide information**.
  - Data is a collection of discrete values that convey information, describing the quantity, quality, fact, statistics, other basic units of meaning, or simply sequences of symbols that may be further interpreted.



| ID | Gender | Age | Nationality | Address | Asences | Grade |
|---|---|---|---|---|---|---|
| 120 | Male | 18 | German | Urban | 5 | 18 |
| 121 | Female | 20 | Indian | Rural | 2 | 16 |
| 122 | Female | 19 | American | Urban | 1 | 19 |
| 123 | Male | 21 | French | Urban | 10 | 10 |
| 124 | Male | 23 | Turkish | Rural | 6 | 14 |

An example of tabular data

## WHAT IS THE DIFFERENCE BETWEEN DATA AND INFORMATION?

– When **combined with other data** or manipulated in some way, the organization derives value from it.

– **Information is data with a context.**

➔ **Information** then leads **to knowledge**.

## DIFFERENT TYPES OF DATA

Data processed by software solutions can be:

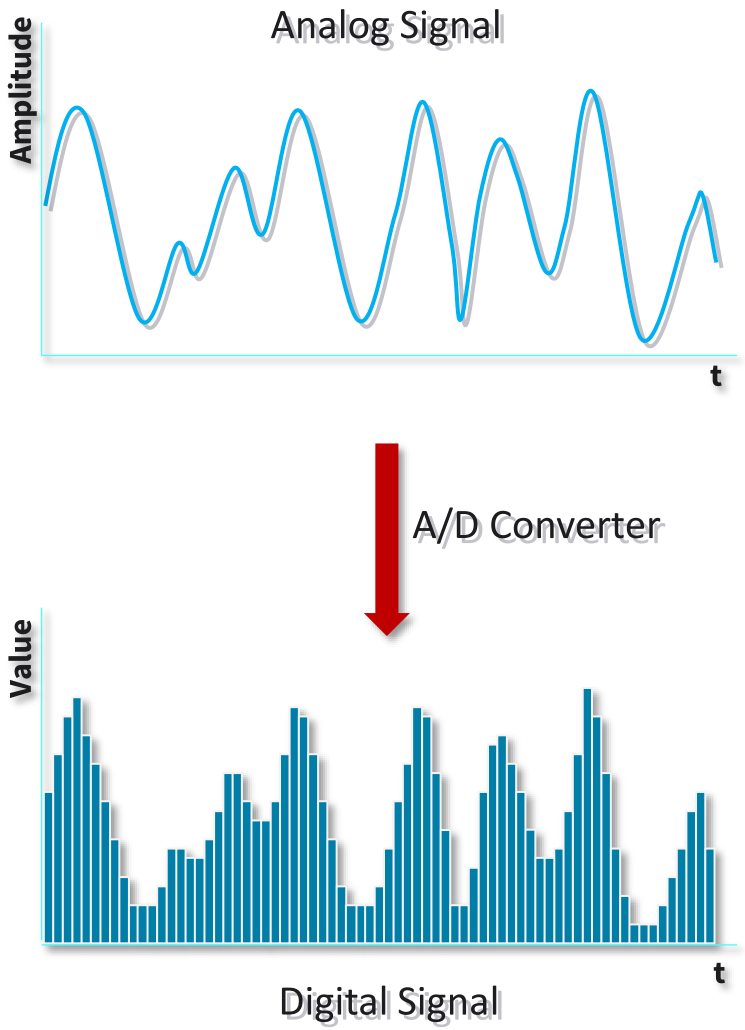
human-generated

machine-generated

Although it is ultimately the responsibility of machines to generate the processed results.
Human-generated data is the result of human interaction with systems, such as online services and digital devices.

## ANALOG VS. DIGITAL DATA

- **Analog data** is temporally continuous. In analog technology, raw data is transformed into electric pulses of varying amplitude.
- **Digital data** is temporally discrete. In digital technology, analog data is transformed into binary representations.
- The conversion of analog data to digital data is called **digitization**.*
- Analog signals are sampled rapidly over time, storing the value of the amplitude a binary number.



* **Digitalization ≠ Digitization**

Digitalization which refers to the operational application of digital technologies and **digital transformation**, the strategic approach.

## MESSAGES VS. EVENTS VS. COMMANDS

**Communication** is key for a distributed environment and software services.

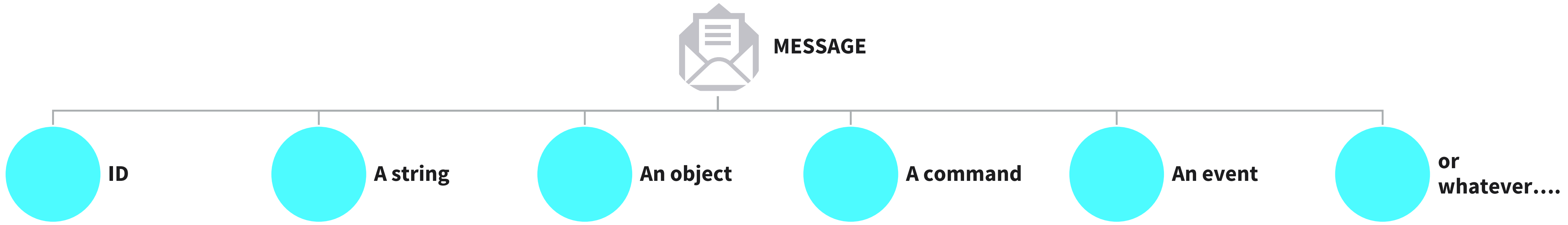
One of the most popular ways of communication is messaging.

There are 3 basic concepts:

- **messages**
- **events**
- **commands**

## MESSAGES

- Message brokers deal with messages.
- Messages are the **basic unit of communication** and **can literally be anything**: an ID, a string, an object, a command, an event or whatever.
- **Messages have no special intent**.



MESSAGE

ID

A string

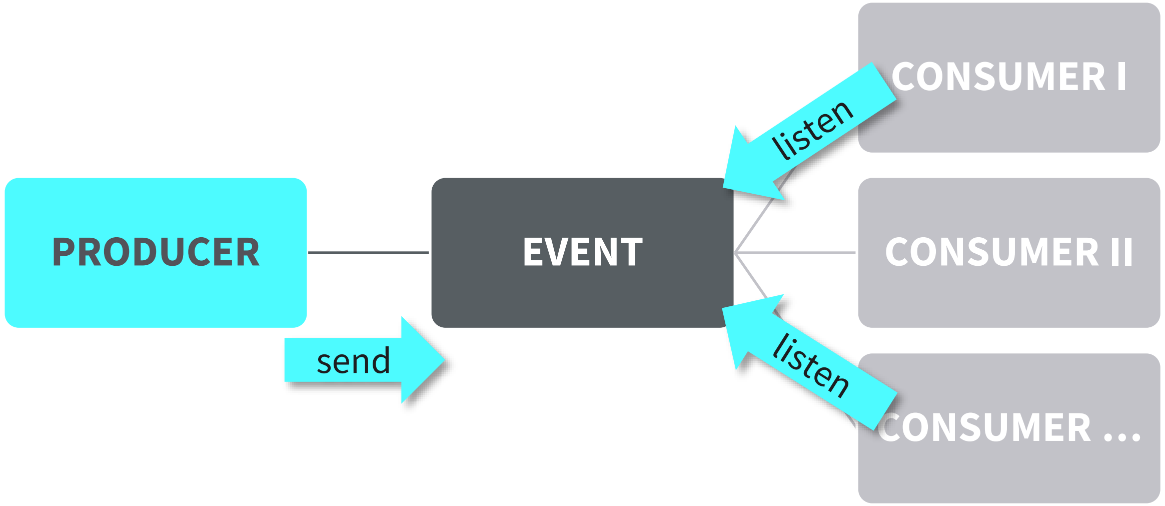
An object

A command

An event

or whatever....

- **An event is a message which informs** various listeners **about something which has happened**.
- It is sent by a producer that doesn’t know and doesn’t care about the consumers of the event.



**Example:**

A typical example would be an online shop.

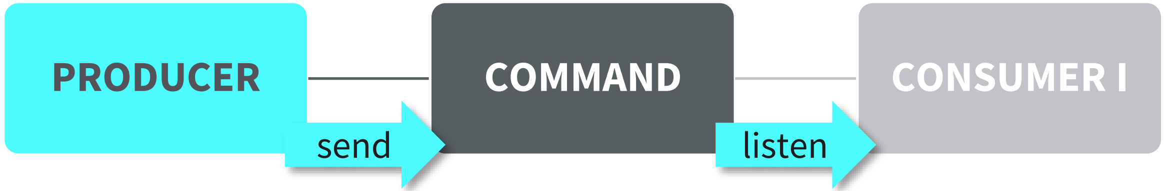
Whenever an order is placed, the shop would publish an *OrderSubmittedEvent* to inform other systems (e.g., the logistics system) about the new order.

However, the shop doesn’t care – or even know – about the consumers.

If the logistics system isn’t interested in the events anymore, it just unsubscribes from the list of consumers.

## COMMANDS

- A **command** on the order side is much more **specific**.
- It’s a one-to-one connection between a producer (who sends the command) and a consumer (who takes and executes the command).



**Example:**

In case of an online shop, one such example could be the *BillCustomerCommand*.

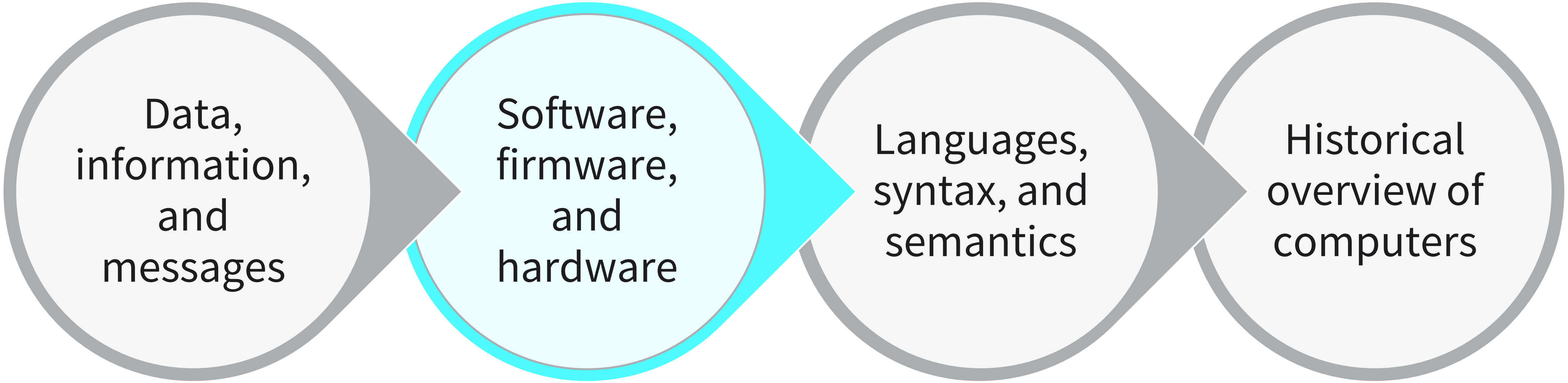
After an order is placed, the online shop sends this command to the billing system to trigger the invoice.

## THE INTENT

- The **difference** between messages, events and commands **lies in their intent.**

- While **messages** have **no special intent** at all, **events inform about something** which has happened and is already completed (in the past).

- **Commands trigger something** which should happen (in the future).



Data, information, and messages

Software, firmware, and hardware

Languages, syntax, and semantics

Historical overview of computers

## SOFTWARE

- **Instructions** that tell a computer what to do.
- Software comprises the entire set of **programs, procedures and routines** associated with the operation of a computer system.
- The term was coined to **differentiate these instructions from hardware**—i.e., the physical components of a computer system.
- **A set of instructions** that directs a computer's hardware to perform a task is called a **program or software program**.

```
    //fires the appear event when appropriate
    var check = function() {
        //is the element hidden?
        if (!t.is(':visible')) {

            //it became hidden
            t.appeared = false;
            return;
        }

        //is the element inside the visible window?
        var a = w.scrollLeft();
        var b = w.scrollTop();
        var o = t.offset();
        var x = o.left;
        var y = o.top;

        var ax = settings.accX;
        var ay = settings.accY;
        var th = t.height();
        var wh = w.height();
        var tw = t.width();
        var ww = w.width();

        if (y + th + ay >= b &&
            y <= b + wh + ay &&
            x + tw + ax >= a &&
            x <= a + ww + ax) {

                //trigger the custom event
                if (!t.appeared) t.trigger('appear', settings.data);

            } else {

                //it scrolled out of view
                t.appeared = false;
            }
        };

        //create a modified fn with some additional logic
        var modifiedFn = function() {

            //mark the element as visible
            t.appeared = true;

            //is this supposed to happen only once?
            if (settings.one) {

                //remove the check
                w.unbind('scroll', check);
                var i = $.inArray(check, $.fn.appear.checks);
                if (i >= 0) $.fn.appear.checks.splice(i, 1);
            }

            //trigger the original fn
            fn.apply(this, arguments);
```

Photo: Markus Spiske on Unsplash

# HARDWARE

- **computer machinery and equipment**, including memory, cabling, power supply, peripheral devices, and circuit boards
- Computer **operation requires** hardware and software.
- Hardware design specifies a computer’s capability; software instructs the computer on what to do.
- The advent of microprocessors in the late 1970s led to much smaller hardware assemblies and accelerated proliferation of computers.
- Modern personal computers are as powerful as early mainframes, while mainframes are smaller and have vastly more computing power than early models.

Photo: Niclas Illg on Unsplash

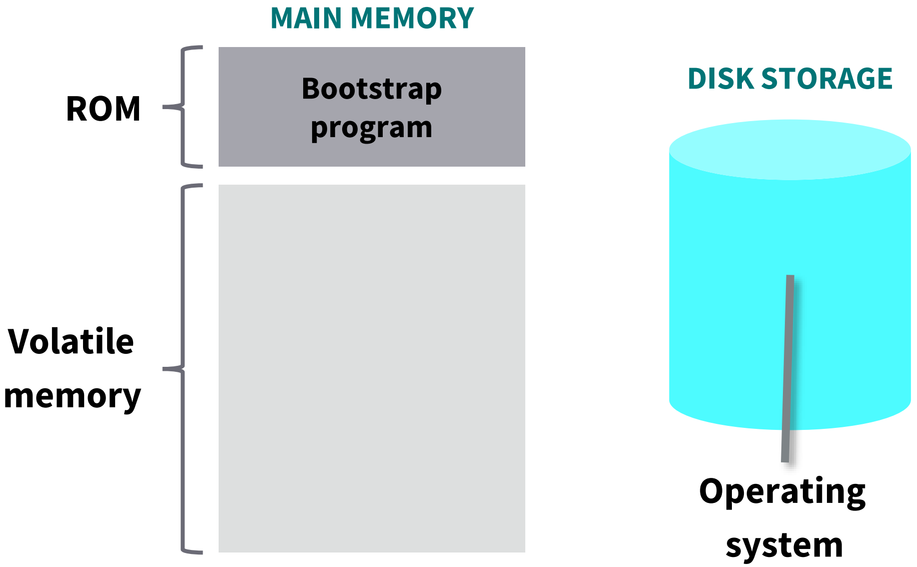
## FIRMWARE

- **Firmware** is a specific **class of computer software** that provides the low-level control for a device's specific hardware.
- Firmware can either **provide a standardized operating environment** for more complex device software (allowing more hardware-independence), **or**, for less complex devices, **act as the device's complete operating system,** performing all control, monitoring and data manipulation functions.
- Typical examples of devices containing firmware are embedded systems, consumer appliances, computers, computer peripherals, and others.
- Almost all electronic devices beyond the simplest contain some firmware.

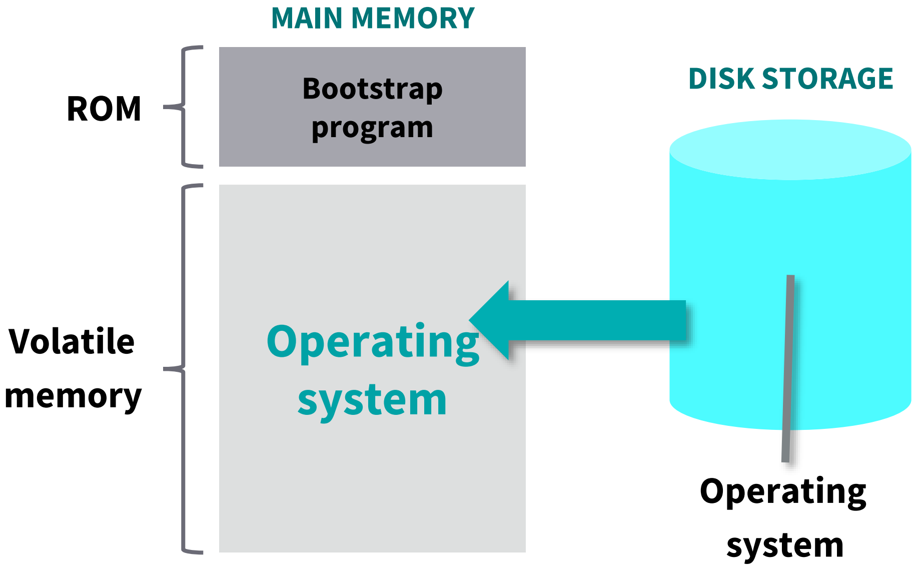
Photo: Denny Müller on Unsplash

**Bootstrap:** Program in ROM (example of firmware)

- run by the CPU when power is turned on
- transfers operating system from mass storage to main memory
- executes jump to operating system



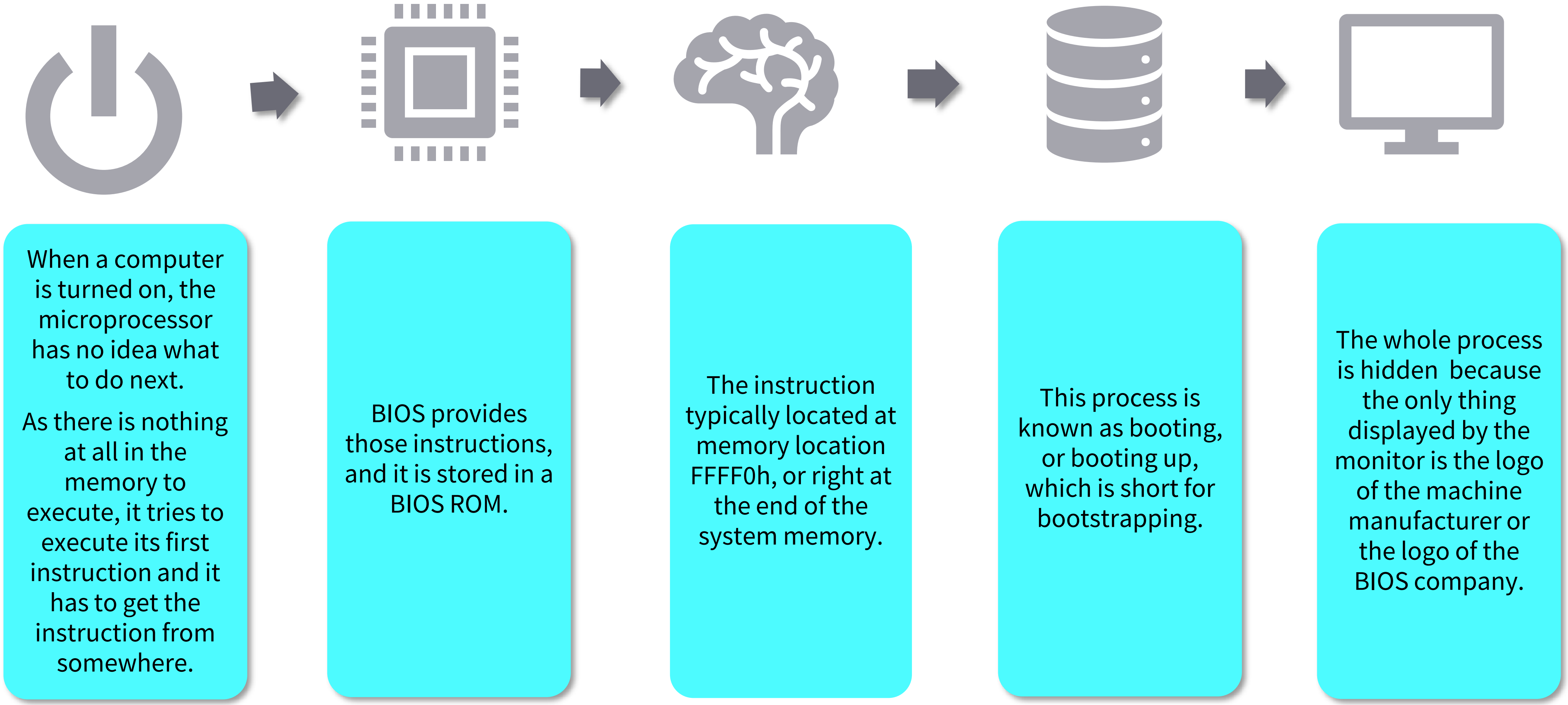
**Step 1:** Machine starts by executing the bootstrap program already in memory . Operating system is stored in mass storage.



**Step 2:** Bootstrap program directs the transfer of the operating system into main memory and then transfers control to it

Source of the text: Brookshea, 2011, p. 121.

# BOOT UP PROCESS



When a computer is turned on, the microprocessor has no idea what to do next.

As there is nothing at all in the memory to execute, it tries to execute its first instruction and it has to get the instruction from somewhere.

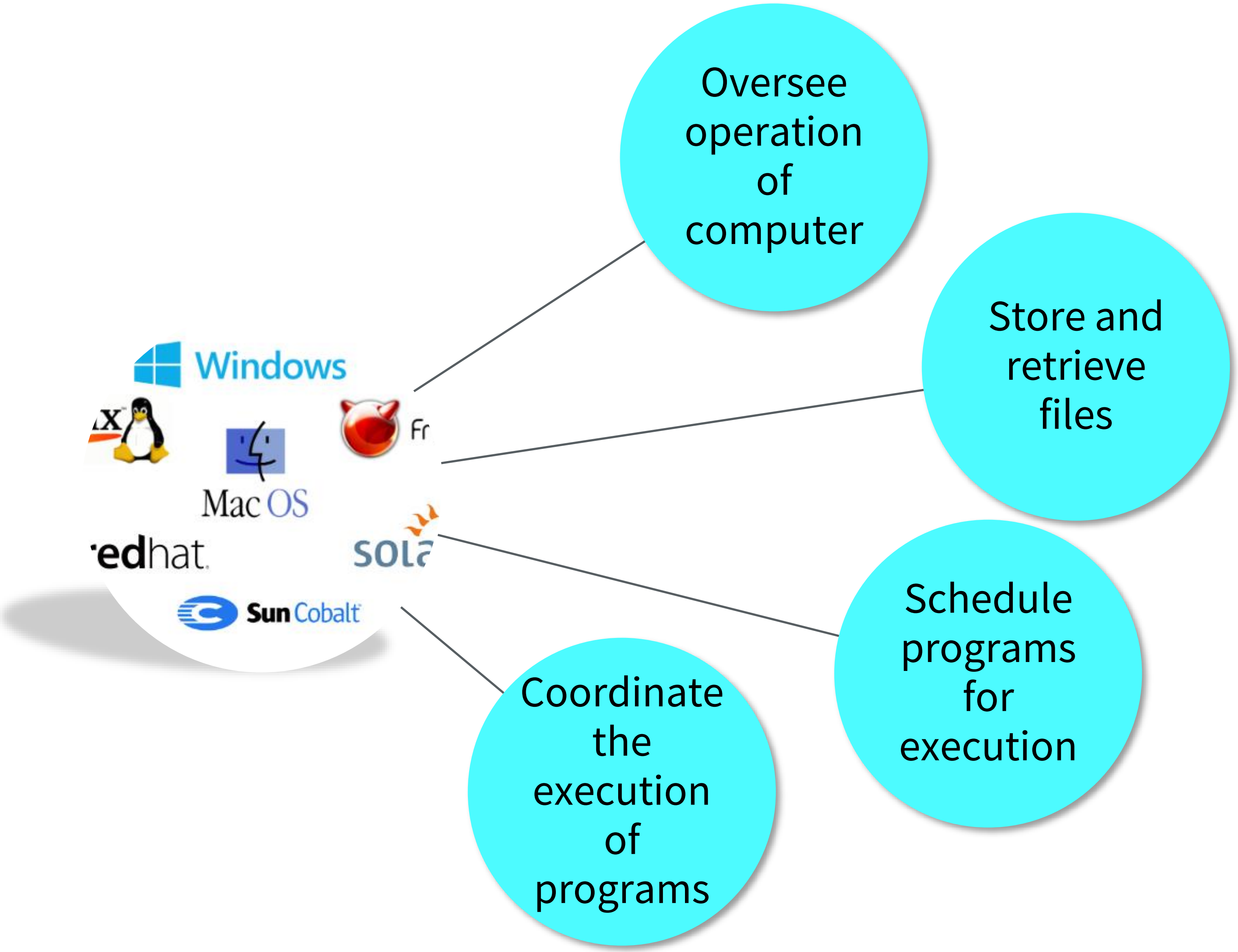
BIOS provides those instructions, and it is stored in a BIOS ROM.

The instruction typically located at memory location FFFF0h, or right at the end of the system memory.

This process is known as booting, or booting up, which is short for bootstrapping.

The whole process is hidden because the only thing displayed by the monitor is the logo of the machine manufacturer or the logo of the BIOS company.

## FUNCTIONS OF OPERATING SYSTEMS



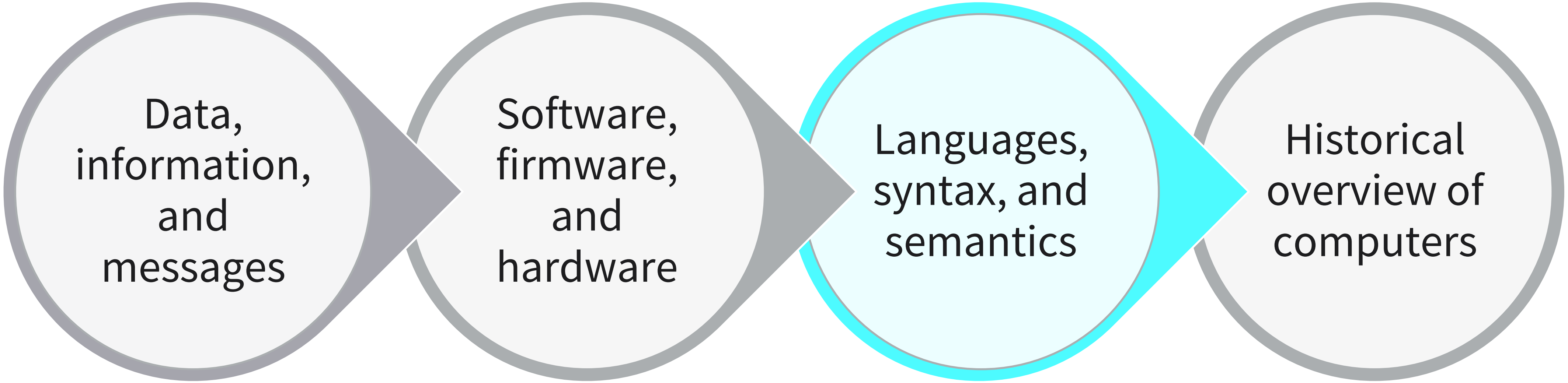
Oversee operation of computer

Store and retrieve files

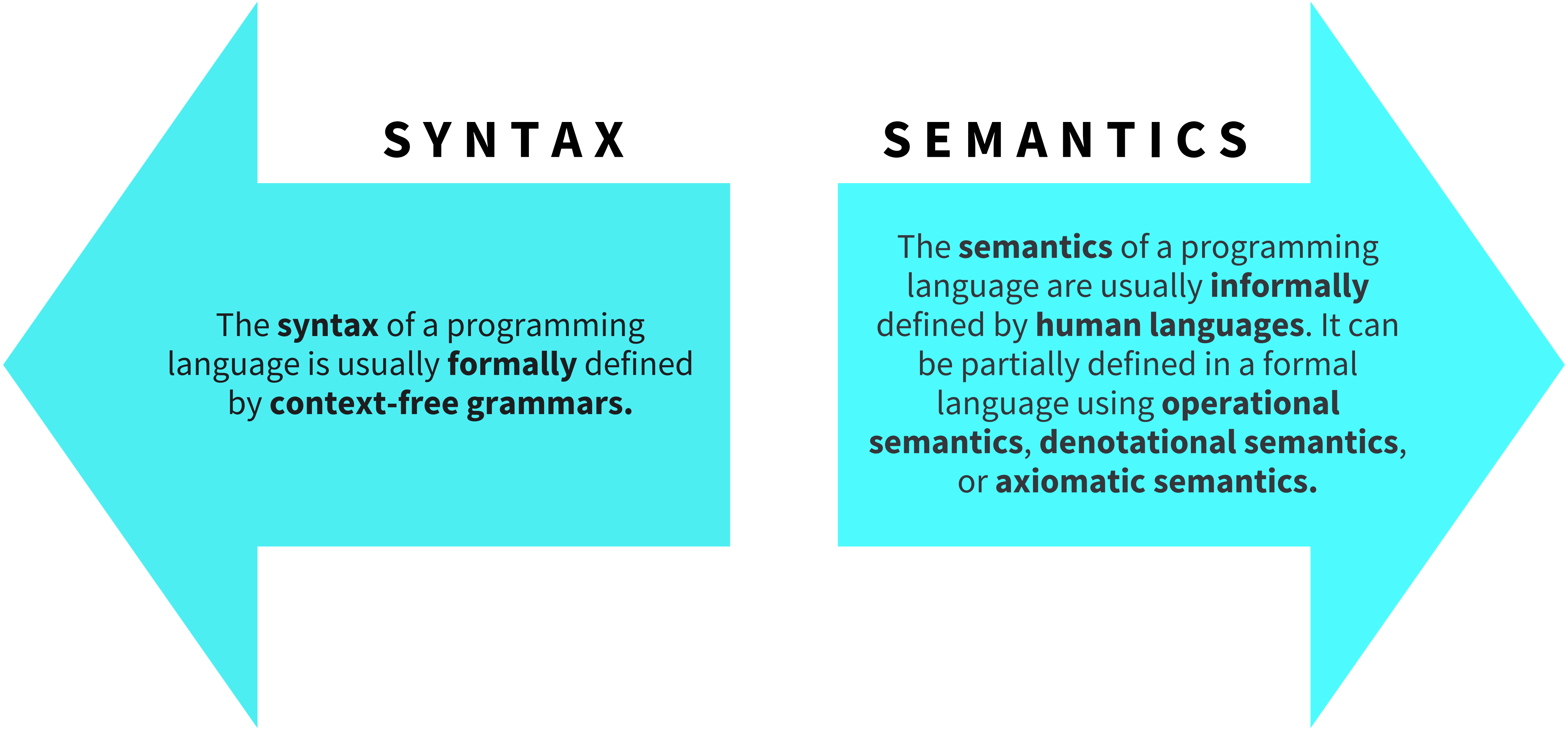
Schedule programs for execution

Coordinate the execution of programs

BASIC CONCEPTS OF DATA PROCESSING


Data, information, and messages
Software, firmware, and hardware
Languages, syntax, and semantics
Historical overview of computers

# FORMAL DEFINITION OF PROGRAMMING LANGUAGES



## SYNTAX

The **syntax** of a programming language is usually **formally** defined by **context-free grammars.**

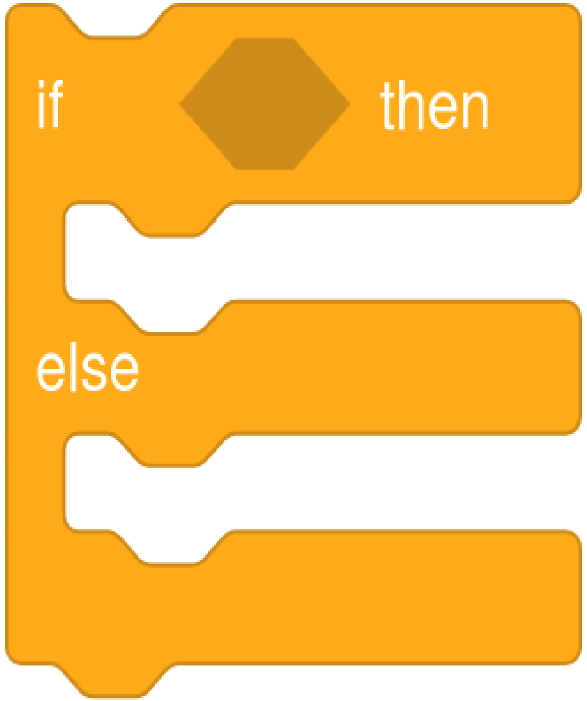
## SEMANTICS

The **semantics** of a programming language are usually **informally** defined by **human languages**. It can be partially defined in a formal language using **operational semantics**, **denotational semantics**, or **axiomatic semantics.**

## SYNTAX

An if-statement consists of the word “**if**” followed by an expression inside parentheses, followed by a **statement**, followed by an optional else part consisting of the word “**else**” and another **statement**.

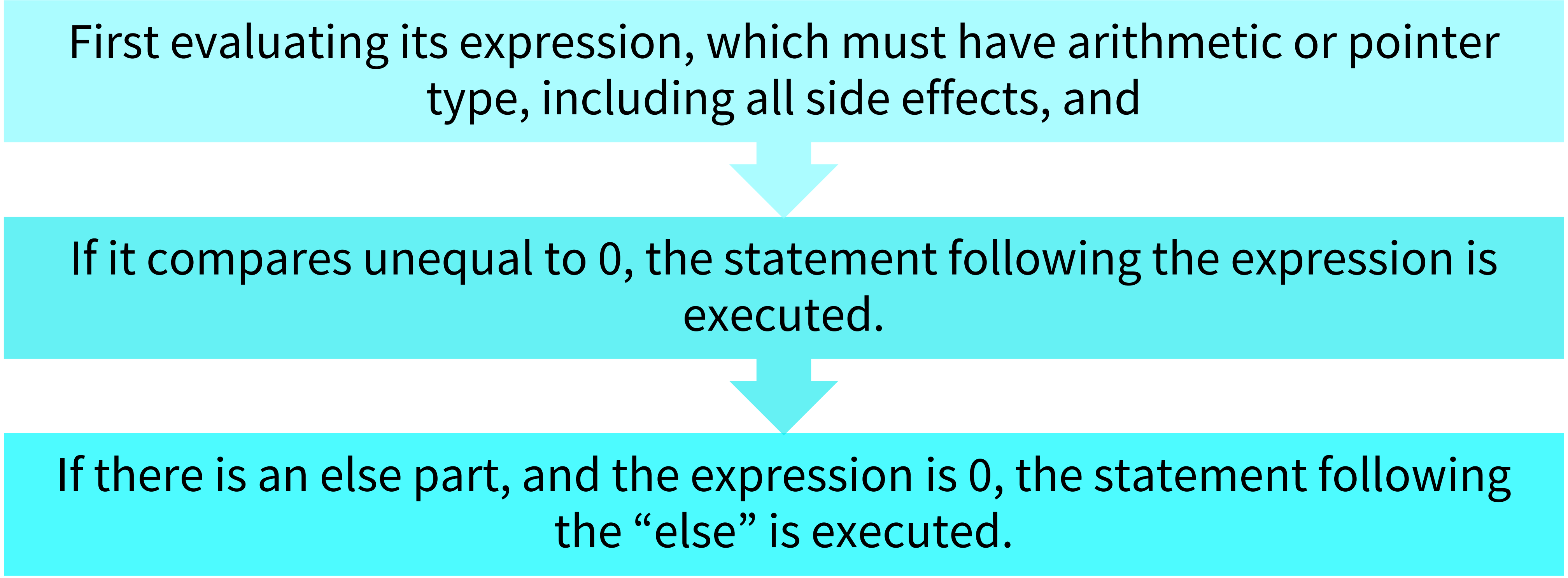
**if (expression) statement else statement**



Picture: Scratch is a project of the Scratch Foundation, in collaboration with the Lifelong Kindergarten Group at the MIT Media Lab. It is available for free at https://scratch.mit.edu

SEMANTICS

## An if-statement is executed by

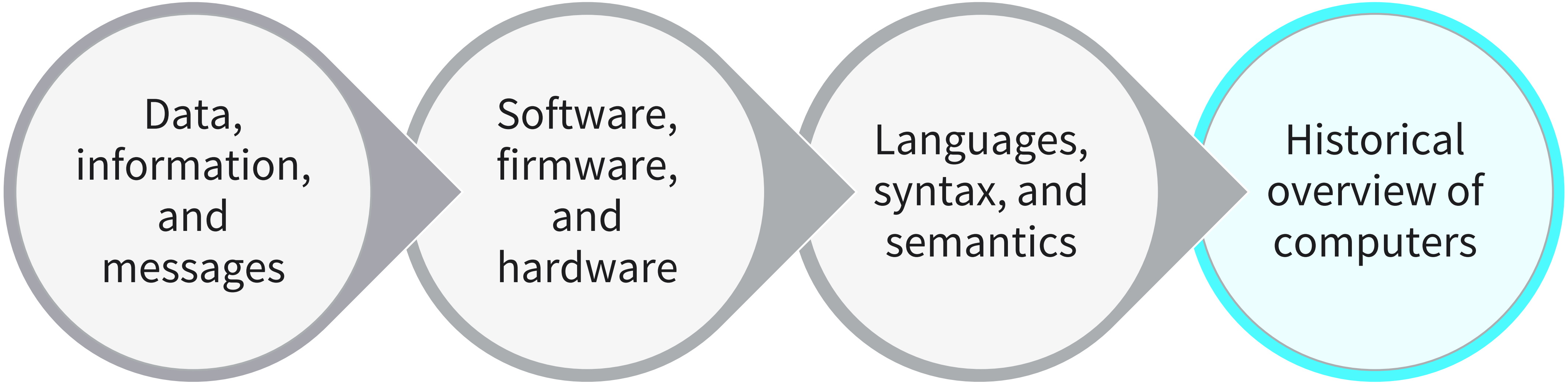


First evaluating its expression, which must have arithmetic or pointer type, including all side effects, and

If it compares unequal to 0, the statement following the expression is executed.

If there is an else part, and the expression is 0, the statement following the “else” is executed.

## BASIC CONCEPTS OF DATA PROCESSING



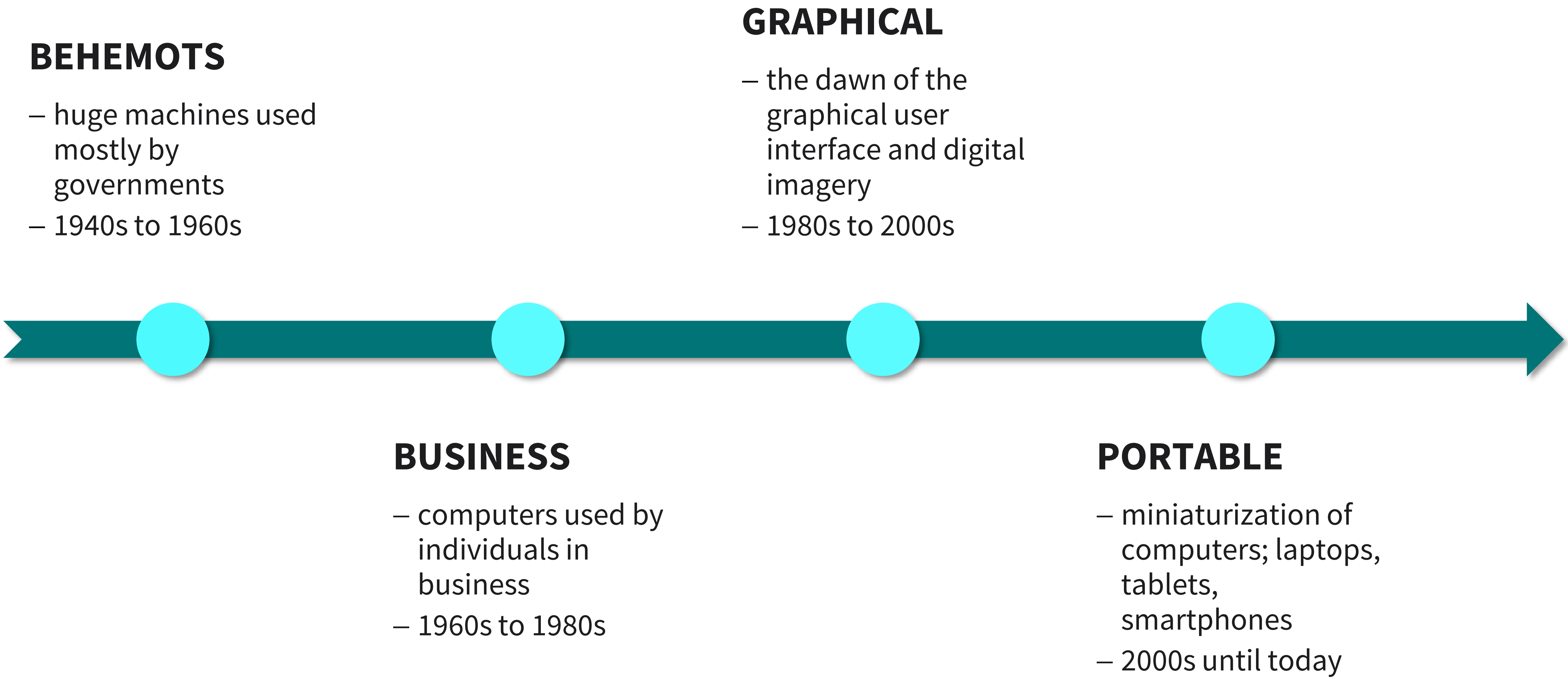
Data, information, and messages

Software, firmware, and hardware

Languages, syntax, and semantics

Historical overview of computers

# ERAS OF COMPUTING



## BEHEMOTS

- huge machines used mostly by governments
- 1940s to 1960s

## BUSINESS

- computers used by individuals in business
- 1960s to 1980s

## GRAPHICAL

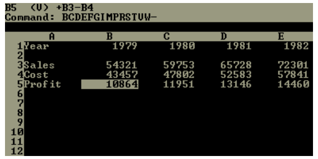
- the dawn of the graphical user interface and digital imagery
- 1980s to 2000s

## PORTABLE

- miniaturization of computers; laptops, tablets, smartphones
- 2000s until today

# USER INTERFACE EVOLUTION MAP

1962



```
B5  (V) +B3-B4
Command: BCDEFGIMPRSTVW-
        A         B         C         D         E
 1Year          1979      1980      1981      1982
 2
 3Sales        54321     59753     65728     72301
 4Cost         43457     47802     52583     57841
 5Profit       10864     11951     13146     14460
 6
 7
 8
 9
10
11
12
```

1978

1991

BLOG

1997

2004

HTML

5

2012

1964

1983

1995

Visual Studio Visual Studio

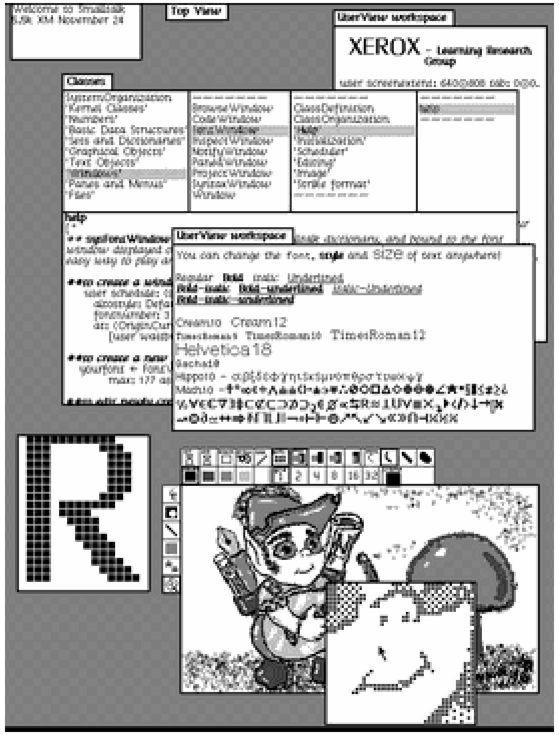
2003

WEB 2.0

2011

## GRAPHICAL USER INTERFACES (GUI)

In 1981 **Xerox introduced a pioneering product, Star**, a workstation incorporating many of PARC's innovations. Although not commercially successful, Star greatly influenced future developments, for example at Apple, Microsoft and Sun Microsystems.

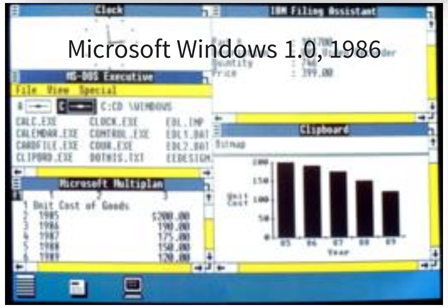


You can change the font, style and SIZE of text anywhere!

Xerox Smalltalk on Star

Apple Lisa, 1983

Apple Macintosh, 1984



Microsoft Windows 1.0, 1986

Sun, 1988

Pictures: Wikimedia Commons, CC-BY and Public Domain

## STUDY GOALS REVIEW

- Understand the **concepts** of data, information, and computer messaging.
- Learn about the **difference** between hardware, firmware, and software.
- Know the **basics** of binary and data interpretation.
- Understand what syntax and semantics refer to in **programming languages.**
- Know the history of **computers and data processing**.

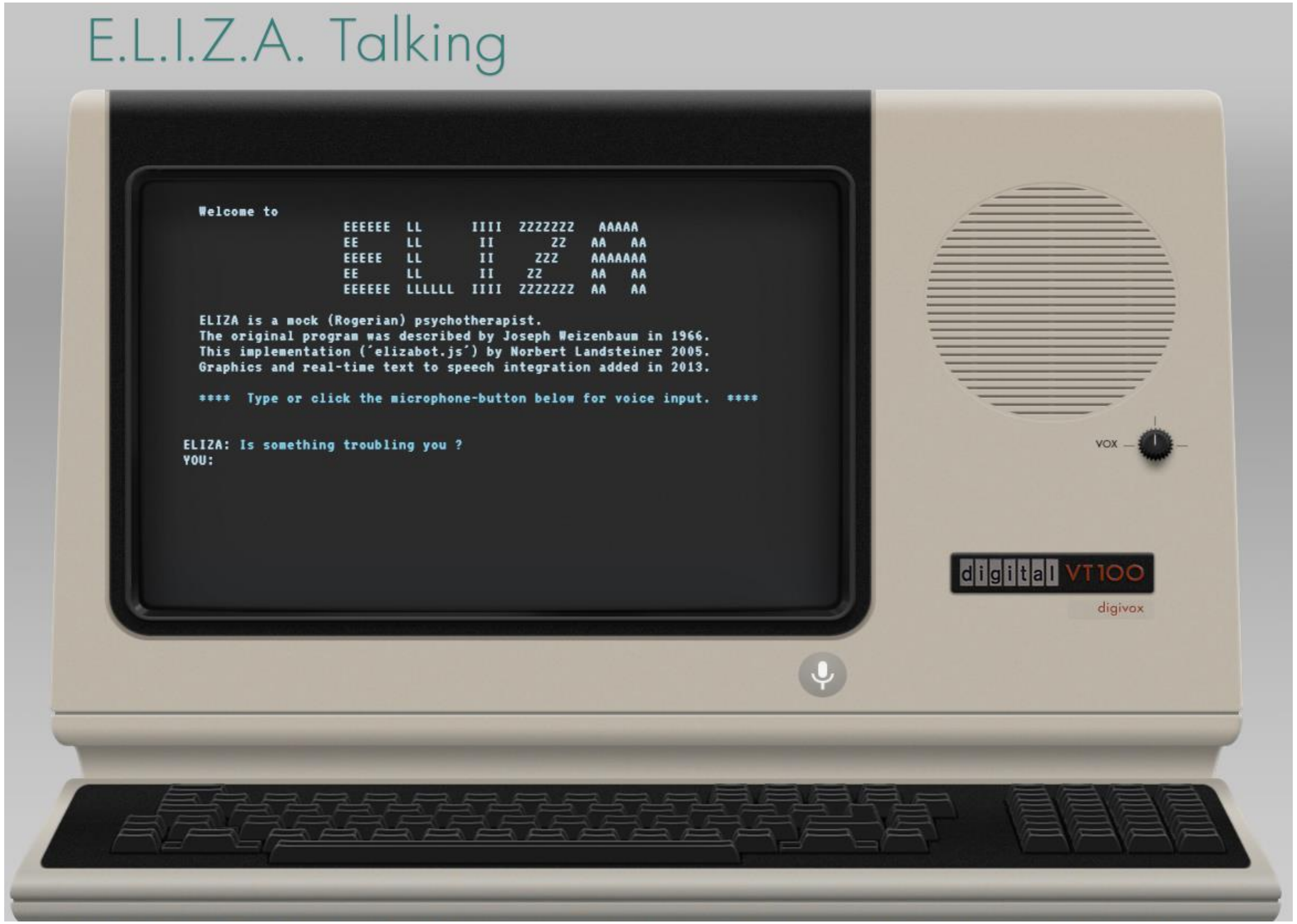
Session 1

# TRANSFER TASK

TRANSFER TASK

# Joseph Weizenbaum 1966

- (1923–2008)
- was a German-American computer scientist as well as a science and society critic.
- Weizenbaum called himself a dissident and heretic of computer science.



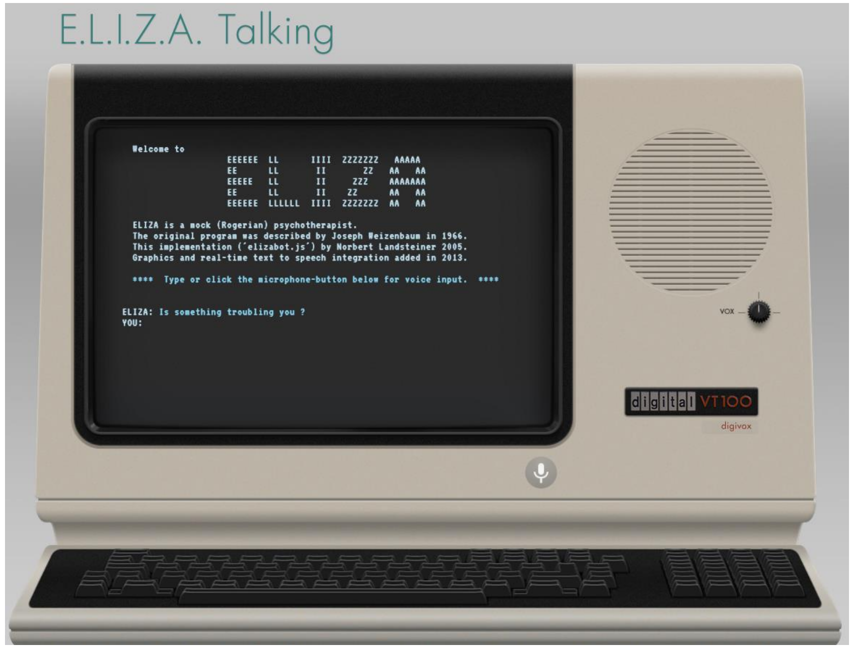
```
  Welcome to
                    EEEEEE  LL      IIII  ZZZZZZZ   AAAAA
                    EE      LL       II        ZZ   AA   AA
                    EEEEE   LL       II      ZZZ    AAAAAAA
                    EE      LL       II     ZZ      AA   AA
                    EEEEEE  LLLLLL  IIII  ZZZZZZZ  AA   AA

  ELIZA is a mock (Rogerian) psychotherapist.
  The original program was described by Joseph Weizenbaum in 1966.
  This implementation (´elizabot.js´) by Norbert Landsteiner 2005.
  Graphics and real-time text to speech integration added in 2013.

  ****  Type or click the microphone-button below for voice input.  ****


ELIZA: Is something troubling you ?
YOU:
```

Photo: https://www.masswerk.at/eliza/

**TRANSFER TASK**

# Exploring the past

1. We use ELIZA
   - Go to https://www.masswerk.at/eliza/
   - Try it out
   - Explain the idea of Eliza
   - Briefly describe your experience



```
  Welcome to
                    EEEEEE  LL      IIII  ZZZZZZZ   AAAAA
                    EE      LL       II        ZZ  AA   AA
                    EEEEE   LL       II      ZZZ   AAAAAAA
                    EE      LL       II    ZZ      AA   AA
                    EEEEEE  LLLLLL  IIII  ZZZZZZZ  AA   AA

  ELIZA is a mock (Rogerian) psychotherapist.
  The original program was described by Joseph Weizenbaum in 1966.
  This implementation ('elizabot.js') by Norbert Landsteiner 2005.
  Graphics and real-time text to speech integration added in 2013.

  ****  Type or click the microphone-button below for voice input.  ****


ELIZA: Is something troubling you ?
YOU:
```

Photo: https://www.masswerk.at/eliza/

## TRANSFER TASK
## PRESENTATION OF THE RESULTS

Please present your results.

The results will be discussed in plenary.

**SAMPLE SOLUTION**

ELIZA is a program operating within the MAC time-sharing system at MIT which makes certain kinds of natural language conversation between man and computer possible.

Input sentences are analyzed on the basis of decomposition rules which are triggered by key words appearing in the input text.

Responses are generated by reassembly rules associated with selected decomposition rules.

The fundamental technical problems with which ELIZA is concerned are:

1. the identification of key words,
2. the discovery of minimal context,
3. the choice of appropriate transformations,
4. generation of responses in the absence of keywords, and
5. the provision of an ending capacity for ELIZA "scripts".

Source of the text: Weizenbaum, 1966, p.1.

Session 1

# TRANSFER TASK

TRANSFER TASK

2. Use a historic graphical user interface on an Apple McIntosh Emulator
   - Go to http://jamesfriend.com.au/pce-js/
   - Try it out
   - Draw a picture with the painting software installed on the computer
   - Write a small text
   - Create screenshots to document what you have done
   - Briefly describe your experience

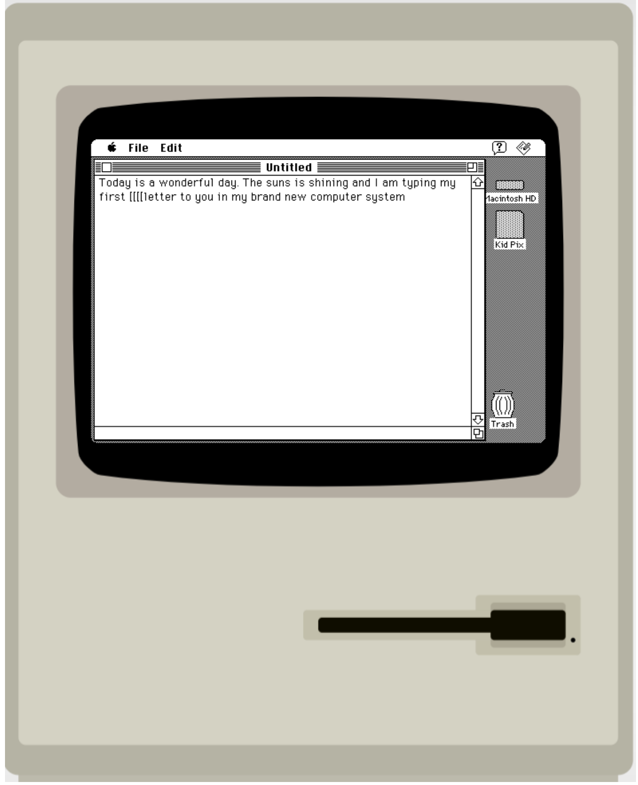
**TRANSFER TASK**
**PRESENTATION OF THE RESULTS**

Please present your results.

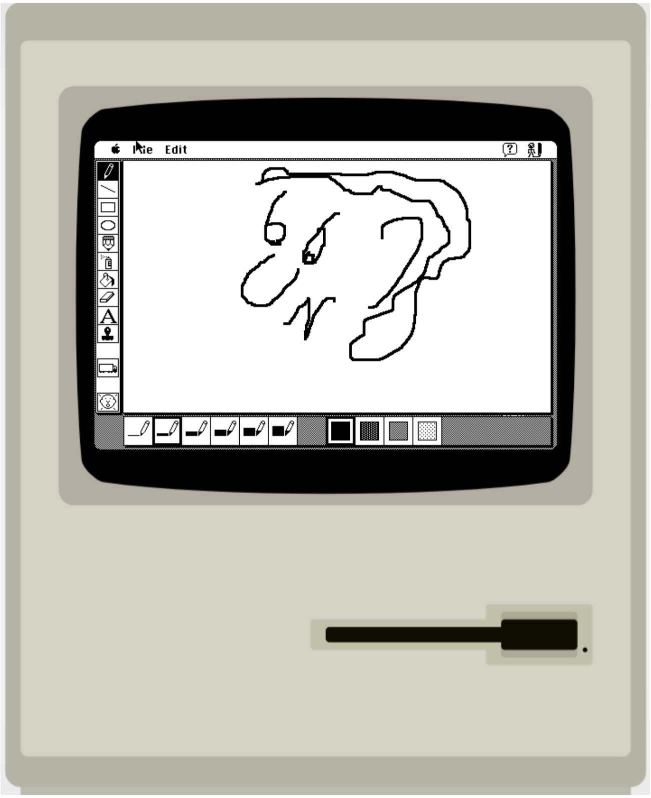
The results will be discussed in plenary.

SAMPLE SOLUTION

# 2. Exploring the past



Today is a wonderful day. The suns is shining and I am typing my first [[[[letter to you in my brand new computer system



Session 1

# TRANSFER TASK

TRANSFER TASK

3. Find out what the major differences between graphical user interface (GUI) and command-line interface (CLI) are.

   Have a look at the following aspects:

   - Ease of use
   - Multitasking
   - Remote access
   - Scripting
   - Speed
   - Control

## TRANSFER TASK
## PRESENTATION OF THE RESULTS

Please present your results.

The results will be discussed in plenary.

**SAMPLE SOLUTION**

The differences between graphical user interface (GUI) and command-line interface (CLI)

### Ease of use

GUI is easier to learn and use. This is because of its user-friendly interface. Users receive immediate visual feedback when dealing with GUI, while this is not obvious in CLI. It would help if you had a higher degree of memorization and familiarity to effectively navigate and operate devices powered by CLI. New users may have more difficulty operating a CLI than GUI.

### Multitasking

CLI offers a great environment for multitasking. While in in a GUI environment, the ability to perform multiple tasks at once on one screen is not as efficient.
With GUI, users rely on windows to control, manipulate, view, and toggle through multiple folders and programs with ease. You can, for example, perform multiple tasks using MS Word.

### Remote access

A command-line interface allows users to access and manipulate files in another device or computer over a network. But this is not such a straightforward process; you must know the commands to do so, which might pose a challenge to new users.
A graphical user interface also allows users to access another computer remotely. Navigating devices remotely in GUIs is more straightforward and requires little experience, unlike in CLIs. IT professionals use GUI to manage servers and access user computers remotely.

**SAMPLE SOLUTION**

The differences between graphical user interface (GUI) and command-line interface (CLI)

**Scripting**
Users of CLI need to master scripting syntax and commands. New users may find it difficult to create scripts.
GUI provides the option to create scripts using programming software. With programming software, one can write scripts or develop other software without necessarily knowing all the syntax and commands.

**Speed**
CLI is more preferred by professionals looking for performance and speed. With GUI, one has to navigate through different icons. This makes GUI slow. But with CLI, one only needs to utilize the keyboard to navigate through the interface. This results in faster performance.
Modern GUIs have improved their speed, but one still requires to use of the mouse and the keyboard to type. Considering that CLI requires only the keyboard, users find taking their hand off the keyboard to move the mouse pointer slower.

**Control**
GUI allows control over the operating system and files. But one still needs CLI to perform advanced tasks. This is because the command line interface provides total control over the operating system and files. Furthermore, tasks become simpler with CLI.
One can create a script containing a few command lines and leave it to perform most of the tasks. Shortcuts in GUI do not fully support automation or scripting. This way, you find that a user has to manually repeat each action within the graphical user interface

LEARNING CONTROL

1. All digital data stored on the computer is saved as:
   a) binary
   b) analog
   c) hexadecimal
   d) decimal

**LEARNING CONTROL**

2. What is the “brain” of the computer, which controls other hardware?

   a) RAM

   b) CPU

   c) hard drive

   d) motherboard

## LEARNING CONTROL

3. Alan, a senior programmer, has written an application to calculate the maximum amount of inventory for warehouse number 1. Instead, it perfectly calculates the average inventory for warehouse 1. What kind of problem does Alan have?

   a) variable
   b) syntax
   c) semantic
   d) network

**LEARNING CONTROL**

4. Firmware is usually stored:

   a) on a computer chip

   b) on the hard drive

   c) in system RAM

   d) in VRAM

5. Before flat screen monitors existed, computer users viewed data on a:

   a) CRT

   b) “trak” table

   c) magnetic medium

   d) wall projector

# LIST OF SOURCES

Brookshear, G. & Bylow, D. (2011). *Computer science: An overview* (11th ed.). Pearson.

Dale, N. & Lewis, J. (2020). *Computer science illuminated* (7th ed.). Jones & Bartlett Learning.

Downey, A. B. & Mayfield, C. (2020). *Think Java: How to think like a computer scientist.* O'Reilly.

Filho, W. F. (2018). *Computer science distilled: Learn the art of solving computational problems.* Code Energy LLC.

Petzold, C. (2000). *Code: The hidden language of computer hardware and software*. Microsoft Press.

Weizenbaum, J. (1966). ELIZA—a computer program for the study of natural language communication between man and machine. *Communications of the ACM 9* (1), 36-45. https://doi.org/10.1145/365153.365168

Whitington, J. (2016). *A machine made this book: Ten sketches of computer science.* Coherent Press.