Object-Oriented Programming I

Computer Programs

Slides by Magdin Stoica
Updates by Georg Feil

Learning Outcomes

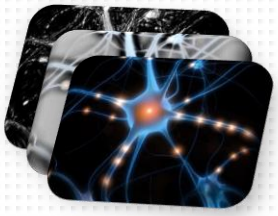
1. Identify the parts of a computer and the role each part plays in a computer program.
2. Define what computer programs are.
3. Describe how programs are executed by computers.
4. Analyze the process of creating a computer program. How do we create a computer program?
5. Define what a programming language is.
6. Describe the two fundamental states of developing a program: design time and runtime.
7. Explain what an Integrated Development Environment (IDE) is.

Reading Assignments

- Let's start the learning pyramid with good old-fashioned reading
- Introduction to Java Programming (required)
  - Chapter 1: Introduction to Computers, Programs and Java
- Head First Java (recommended)
  - Chapter 1: Dive in A Quick Dip: Breaking the Surface
    - From "The way Java works" to "Writing a class with a main"

Us – How do we function?

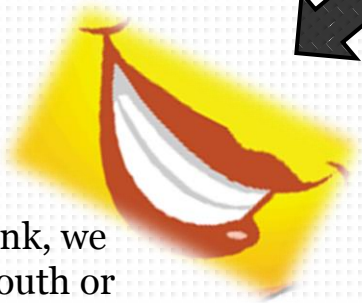
Our senses send our brain information



Memory Brain Cells

Our Brain

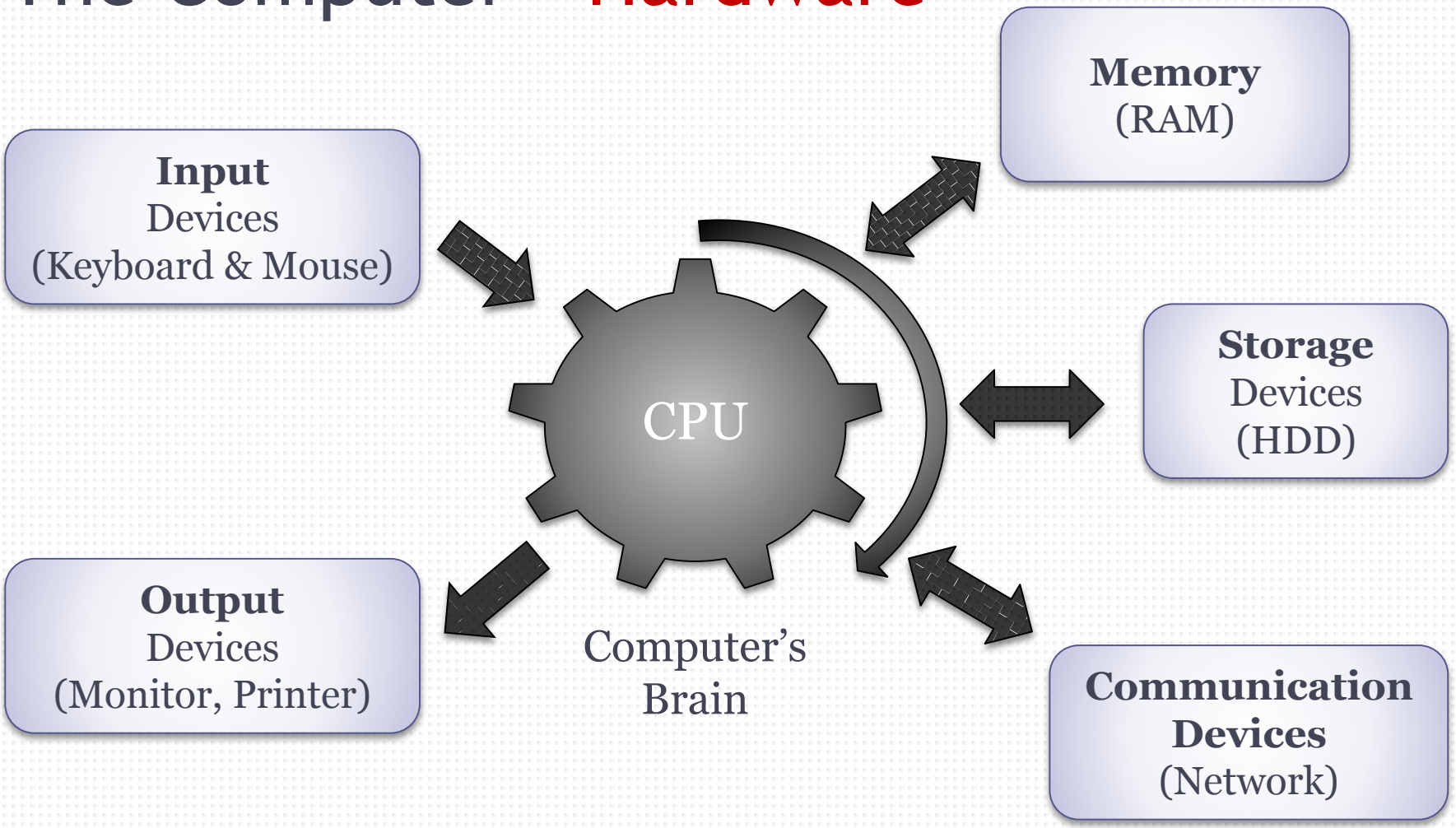
Better memory



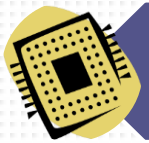
Once we think, we open our mouth or so it's supposed to work.

Only Vulcans can mind-meld, we tweet

The Computer - Hardware



Computer Parts



The CPU is the brain of the computer, in charge of executing things

RAM is the memory is where the computer remembers things

Storage devices are where the computer remembers things PERMANENTLY (even after a restart)

Communication devices are what computers use to talk to each other

Input devices like the keyboard, mouse, touch surfaces provide information for processing

Output devices are used to present information to the user, the results of what the computer did

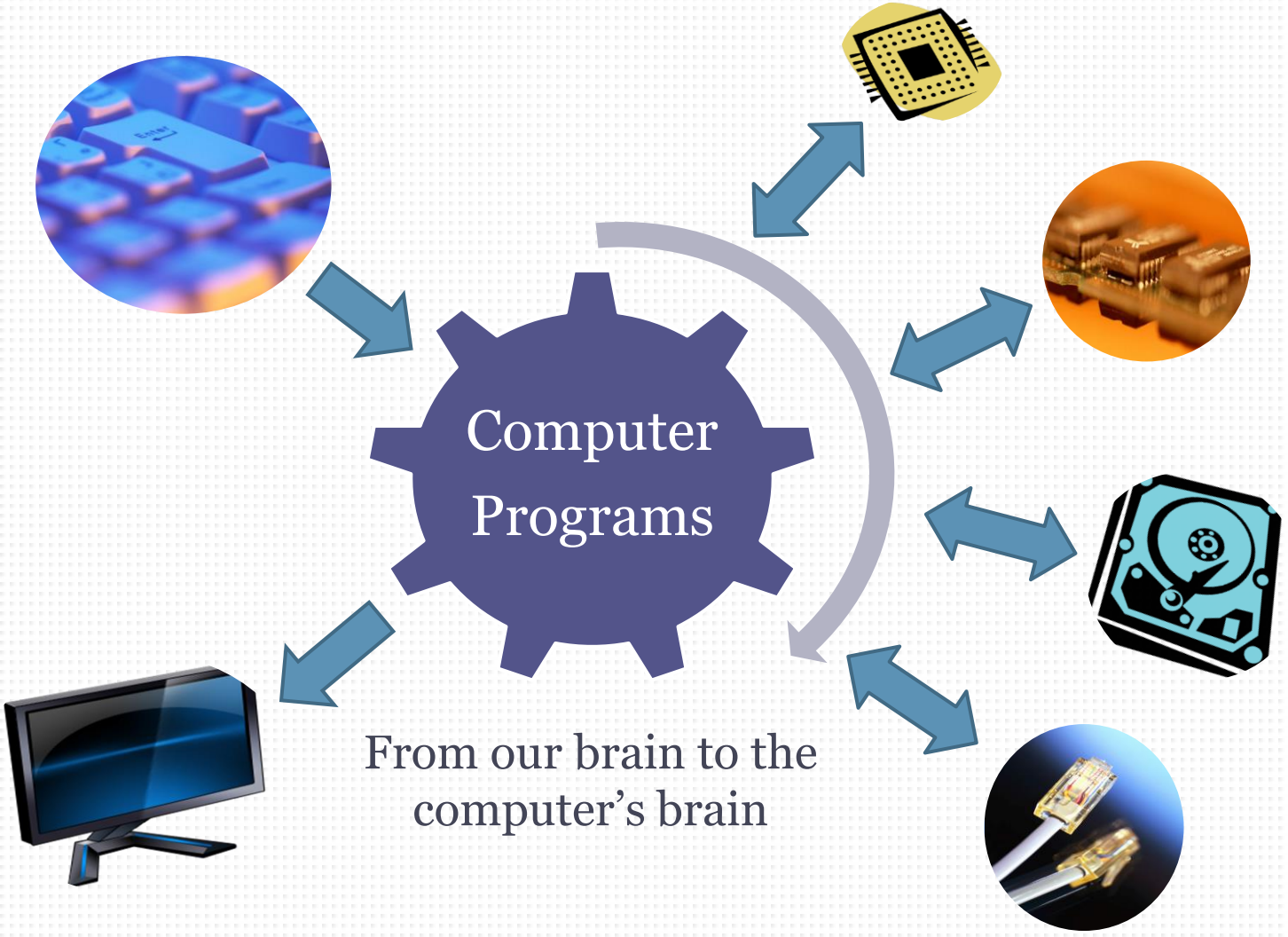
Hardware

=

Electronic computer devices

(hard to change)

Computer Programs – Software



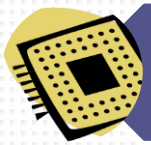
Software

=

Computer Programs

(easier to change)

How software uses hardware



Programs tell the CPU what to do using commands or instructions

Programs store temporary information in memory

Programs store permanent information on storage devices

Programs use communication devices to "talk" to other programs

Programs get information by reading from input devices (from users, other computers etc.)

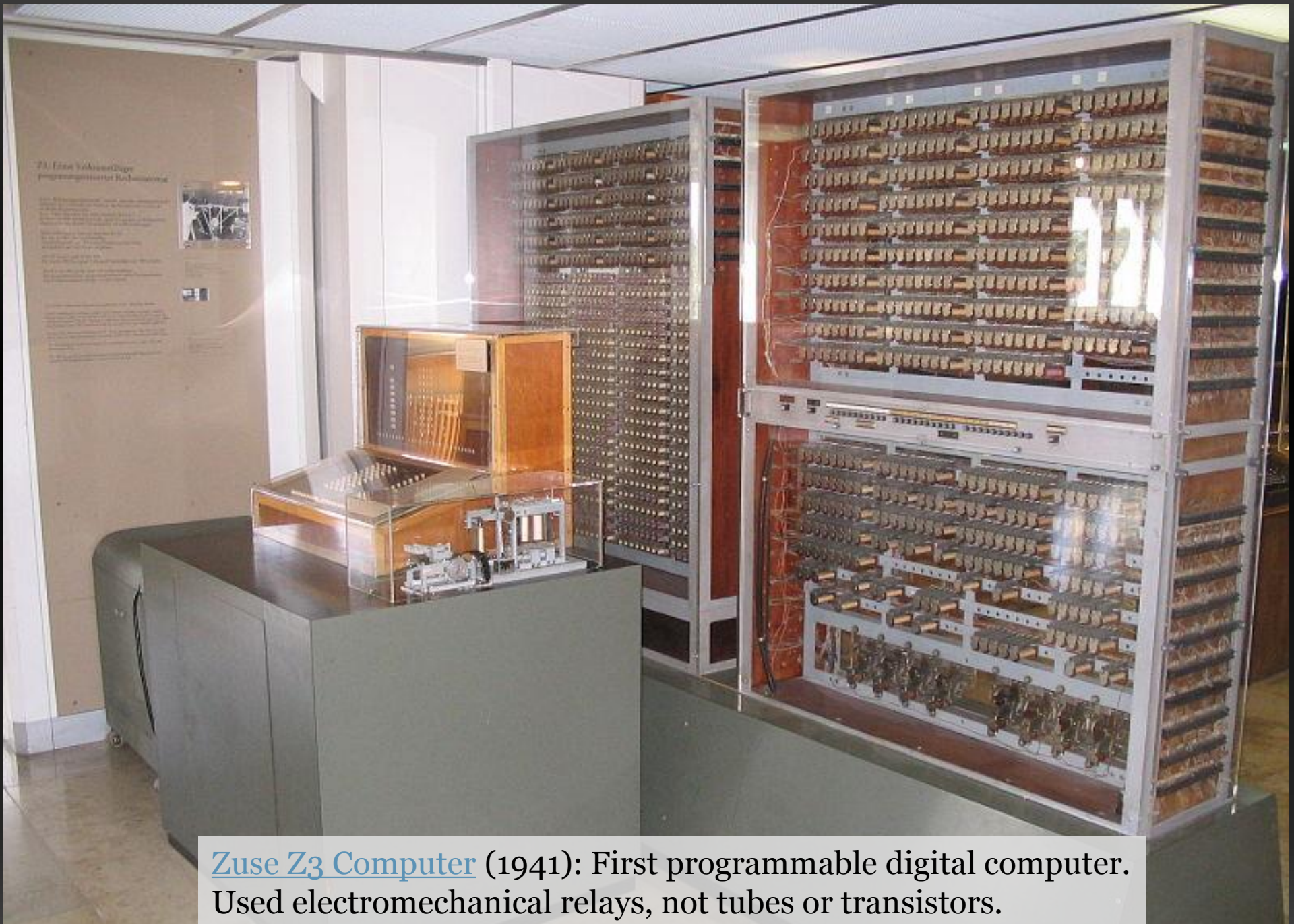
Programs present their results by writing or printing to output devices (to users / other comp.)

How are programs executed

- Computer circuits are made of transistors, one of the greatest inventions of the 20th century (originally: vacuum tubes)
- Through their transistors, digital electronic circuits can store only two states: on (1) and off (0)
- Machine Language – language computers understand

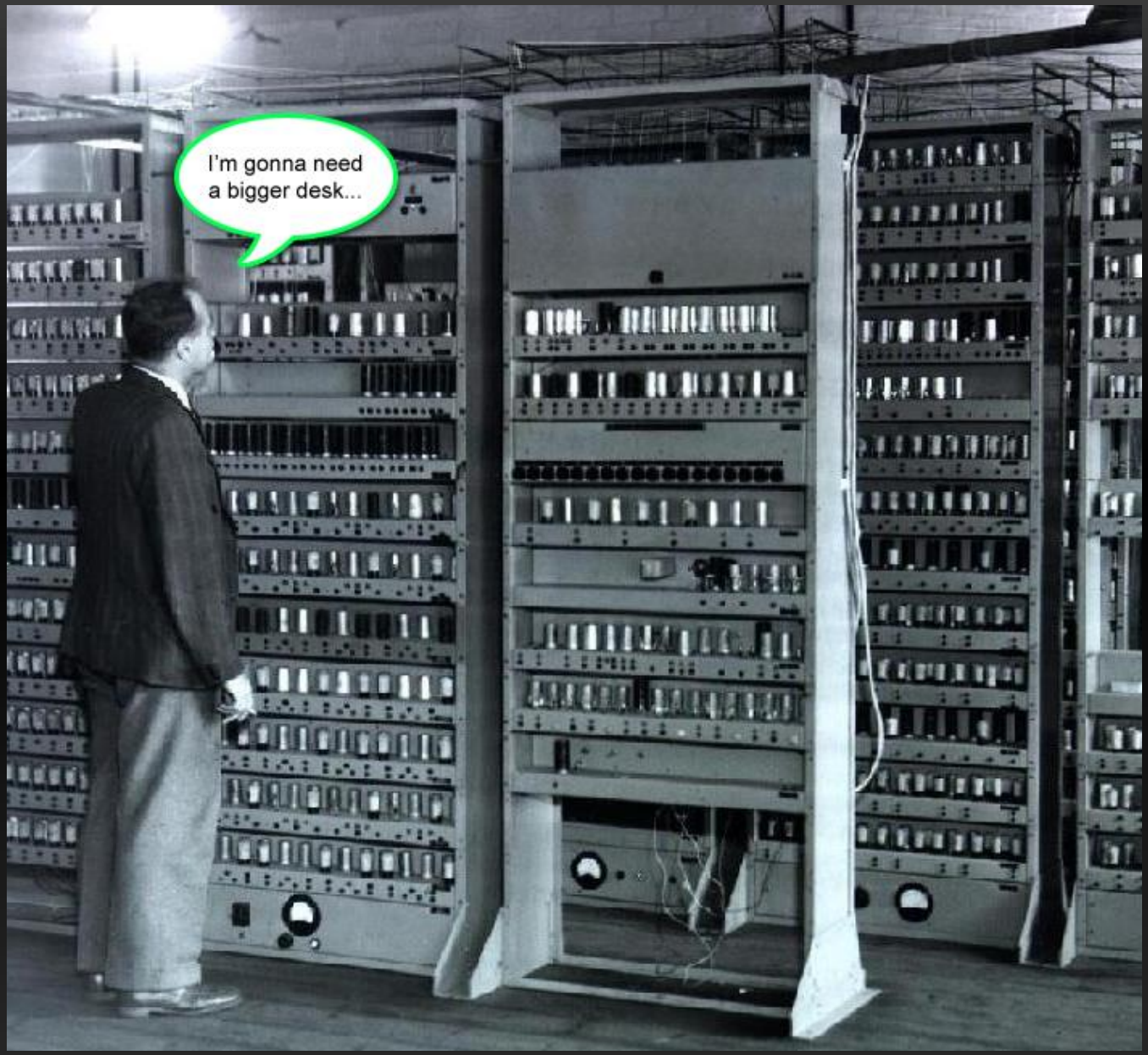
Machine Language

- Language computers understand: if you "speak" it to them they will do as you ask
- Very simple "alphabet": only two symbols: 1 (one) and 0 (zero). These symbols are called "bits" or binary digits
- Simple "words": combinations of 1 and 0 (e.g. 100111110001)
- Words the computer understands are called the instruction set. Each "word" in machine language is an "instruction".
  - They're meant to tell the computer (CPU) to do something
- Very simple syntax: only one word at a time, words are only allowed to have a fixed number of symbols (1s and 0s): 8, 16, 32, 64



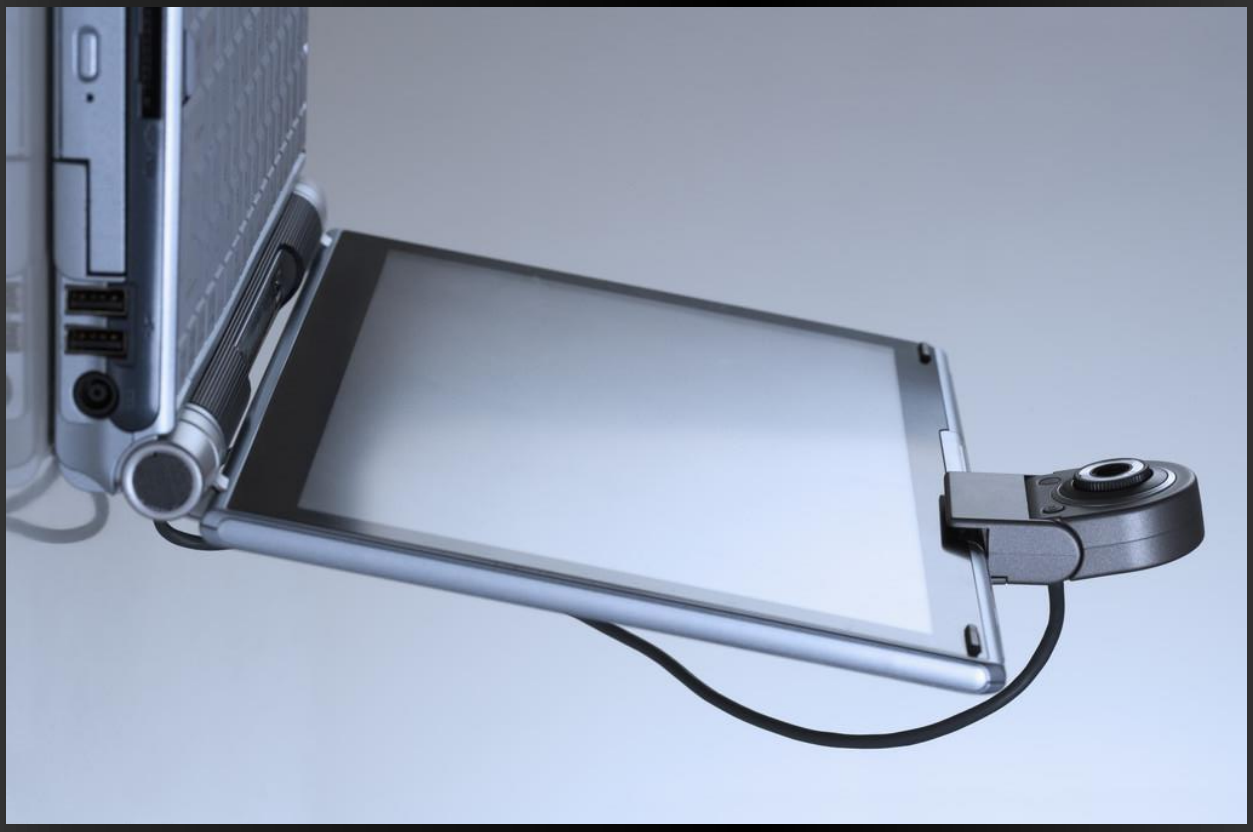
Zuse Z3 Computer (1941): First programmable digital computer. Used electromechanical relays, not tubes or transistors.

1 and 0



Electronic Delay Storage Automatic Calculator (EDSAC, 1949)

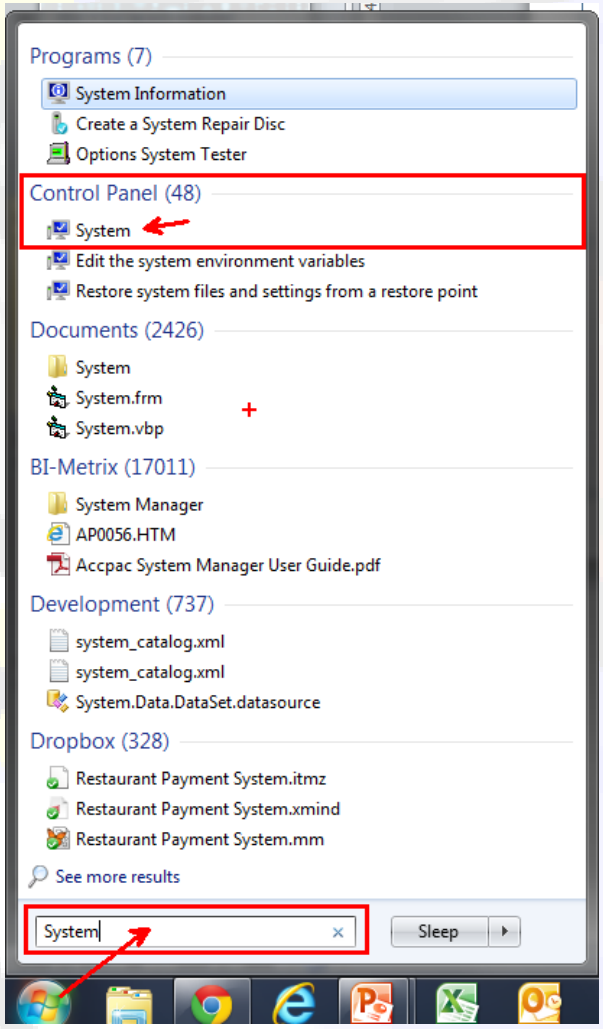
UNIVAC I (1951): One of the first commercially sold mainframe computers



Still 1 and 0

Exercise 1: How many bits does your computer use?

- In the Windows 7 Start Search Box, type “System”
- Open the “System” panel that appears under the “Control Panel” group.
- Find the **System Type** property under the System group and report back how many bits in a “word” does your machine understand
  - Make a note of it
  - You will need to remember it when installing Java



Computer programs – What we run

- Computer programs are a collection of machine language instructions, machine code (1s and 0s)
  - Stored (saved) in executable files or binary files onto storage devices
- Who writes computer programs? Programmers, software developers, software engineers.
  - Sometime programs can write other programs but that's for another day
- Hmm, we don't speak 1s and 0s.
  - We speak a natural language, like English, French, Punjabi, Spanish etc.
  - Complex languages, complex rules, complex vocabulary
  - Too hard to write a compiler that translates natural language commands
  - We need a simpler language, as close to natural as we can make it but simpler
  - We need a programming language

An executable is a file containing a computer program
1s and 0s

Programs are written in a
programming language
(not 1s and 0s)

The compiler creates 1s and 0s

- Computers used to have physical switches to input ones and zeroes, commands for the CPU (instead of a keyboard)
  - These days we let something else do the hard work
- Something else has to **"translate"** the commands we can write with a keyboard into the 1's and 0's that computers understand

THE COMPILER

- The compiler is a computer program itself that knows how to translate commands we give into 1s and 0s that computers can run
- In the end, only executable code made of 1's and 0's can be executed, run by the computer.

The compiler translates _____ into _____

Fill in the blanks.

A compiler is a program that
translates
programming language
into
machine language

Source Code

- Source code is a set of commands or instructions written in a programming language
- Source code is stored (saved) in source files
  - These are text files with special extensions and of course special content
- Source files are stored (saved) onto storage devices such as the hard drive
  - If stored only in the memory of the computer it would disappear as soon as the computer turns off
- A compiler reads source files, analyzes all the commands and translates them into an executable file that contains machine code
  - The process of transforming a source file into an binary file is called compilation, or compiling a source file
- The computer executes or runs the machine code

Computer Programs – What we create

- A computer program is a collection of source files
  - Simple programs may consist of only one source file
  - Large programs have thousands of source files
  - Remember: source files contain programming language "sentences" called commands, instructions or statements
- Computer programs are written in programming languages
- There are many programming languages with different characteristics, strengths and weaknesses
  - Classified as "generations" of languages: 1st, 2nd, 3rd and 4th generation depending on when they were created and what they can do
  - Java is a 3rd generation language

Exercise 2: Breaking Down The Steps

- To write a computer program you have to break every action down into steps
  - Small, very detailed steps

- Imagine you want to describe how to boil an egg…
- …your instructions are for a 5-year-old girl or boy
- Write down the detailed steps needed
  - Step 1: …
  - Step 2: …
  - …

Design-Time vs. Runtime

Design-Time

- A program is said to be "at design-time" when we are creating it
- At design time we …
  - THINK and PLAN
  - Create and write source files
  - Write commands in a programming language
  - Compile source files and build executable files
  - A program whose programming language "sentences", the commands are incorrect is said to have syntax errors.
  - Syntax errors are detected by the compiler when it tries to translate them into executable code
  - So what else do we do at design time? We fix errors.
- A program that has syntax errors cannot be compiled and therefore cannot be run

Runtime

- A program is said to be “at runtime” when it is running it
- At runtime we …
  - Test the program by using its functionality in every way possible. If the program terminates unexpectedly is said to have runtime errors.
  - Verify the output of the program to be correct. If the output is incorrect the program is said to have logic errors.
  - To fix any errors, we note any error message that is displayed and than stop the program to go back to design time. We fix errors in design time.
  - We will learn more, a lot more about testing later
  - Use the program and enjoy the functionality it provides
- A program at runtime cannot have syntax errors but can have other types of errors: runtime errors or logic errors.

Design-Time vs. Runtime

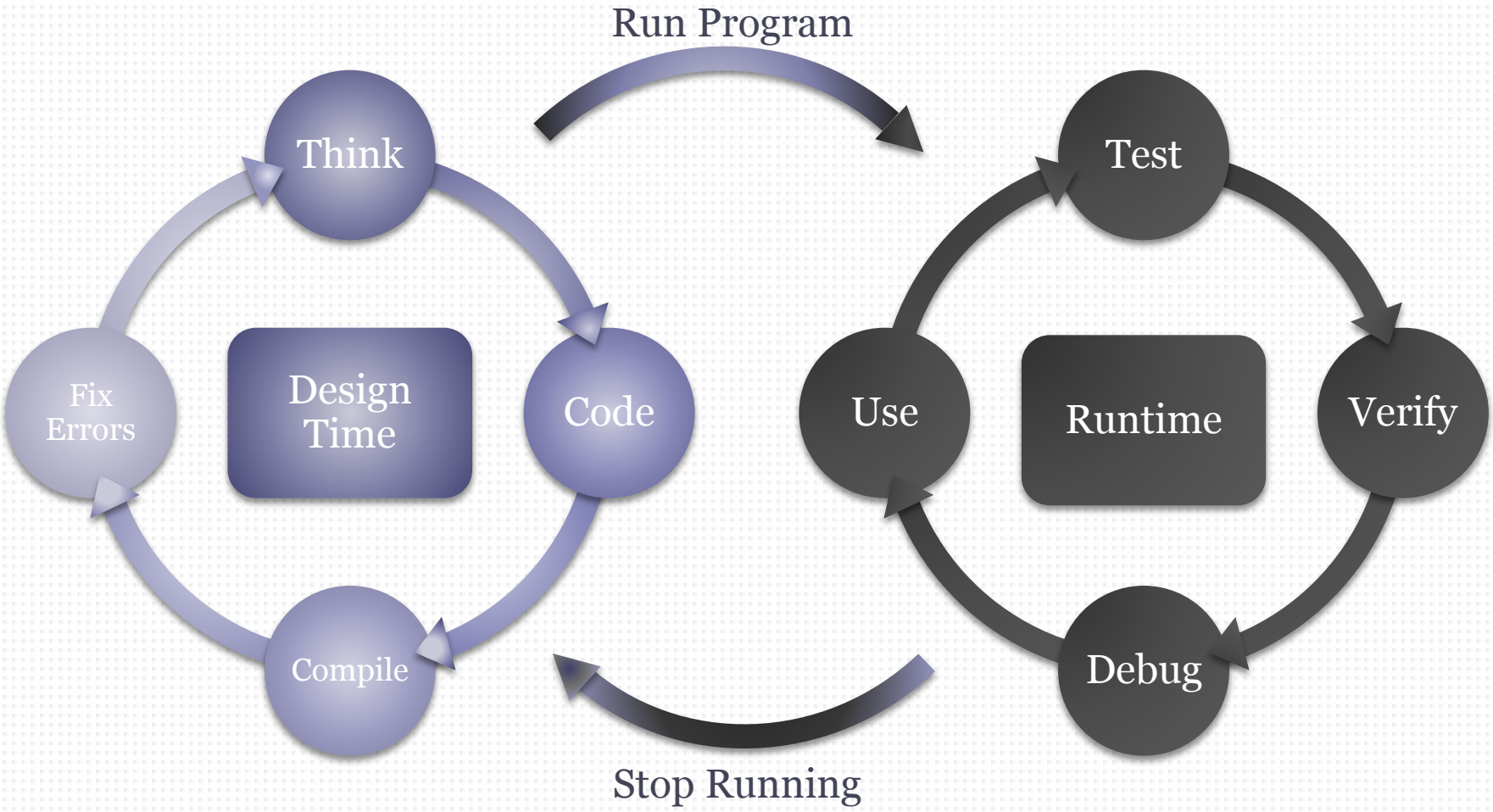
Design-Time

- We are creating the program
- Source code
- Source files
- Programming language is high-level, English-like
- Find syntax errors
- We think, code, fix syntax, runtime or logical errors
- We compile
- We have fun creating something great

Runtime

- We are running the program
- Machine code
- Executable files
- Machine language is made of `1`s and `0`s or similar
- Find runtime or logic errors
- We test different input, verify the output, use the program
- We debug (more on this later)
- We have fun using what we have created with our mind

Programming



Puzzle 1: Who am I? (one thing)

- I only have two symbols in my alphabet, 1 and 0
- All my words have the same number of symbols, for example 32
- Computers speak it fluently no matter how old or how new
- If you want to tell a computer to do something you must use it, somehow.
- My two symbols, 1 and 0 are called bits
- I am a language
- I am a language that machines (computers) speak

Puzzle 2: Who am I? (one thing)

- I can do anything, from saving lives to landing space robots on Mars
- I keep jumping between design-time and run-time
- I am created in text files but when I run, I am made of cool unreadable stuff: 1s and 0s.
- I require a mind and a computer to exist
- I am easy to make as long as whoever makes me practices and practices… and practices. Like any great thing it requires hard work.
- I am almost made with simple English.

Puzzle 3: Who am I? (one thing)

- I am a program
- I am a program that translates
- I make 1s and 0s (or something like it)
- I read source files and create binary files (or something like it)
- I can almost read English and I can create no human can create
- If I don't understand what you tell me I will give you syntax errors
- I compile source code into machine code

Puzzle 4: Who am I? (one thing)

- I am smart!
- I am cool!
- I work hard and I learn a lot, every day!
- I can create amazing things with my mind and a computer!
- I make today's world work!
- I speak at least two languages
- I actually understand words made of 1s and 0s