

A decorative graphic on the left side of the slide, consisting of white lines and circles on a teal background, resembling a circuit board or a stylized tree structure.

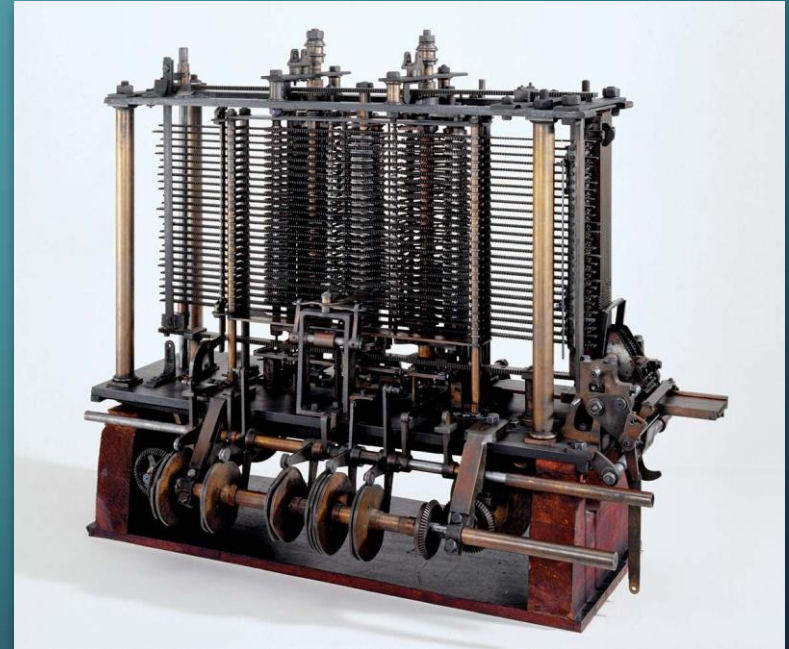
INTRODUCTION TO PROGRAMMING

INSTRUCTOR: JHUN BRIAN M. ANDAM

THE ANALYTICAL ENGINE

The Analytical Engine, designed by Charles Babbage in the 1830s, was a remarkable invention that was conceived with several purposes in mind, even before Ada Lovelace made her pioneering contributions to the concept of programming it. Babbage's vision for the Analytical Engine included the following primary purposes:

- Mathematical Calculations
- Error Reduction
- Data Processing
- Mechanical Automation



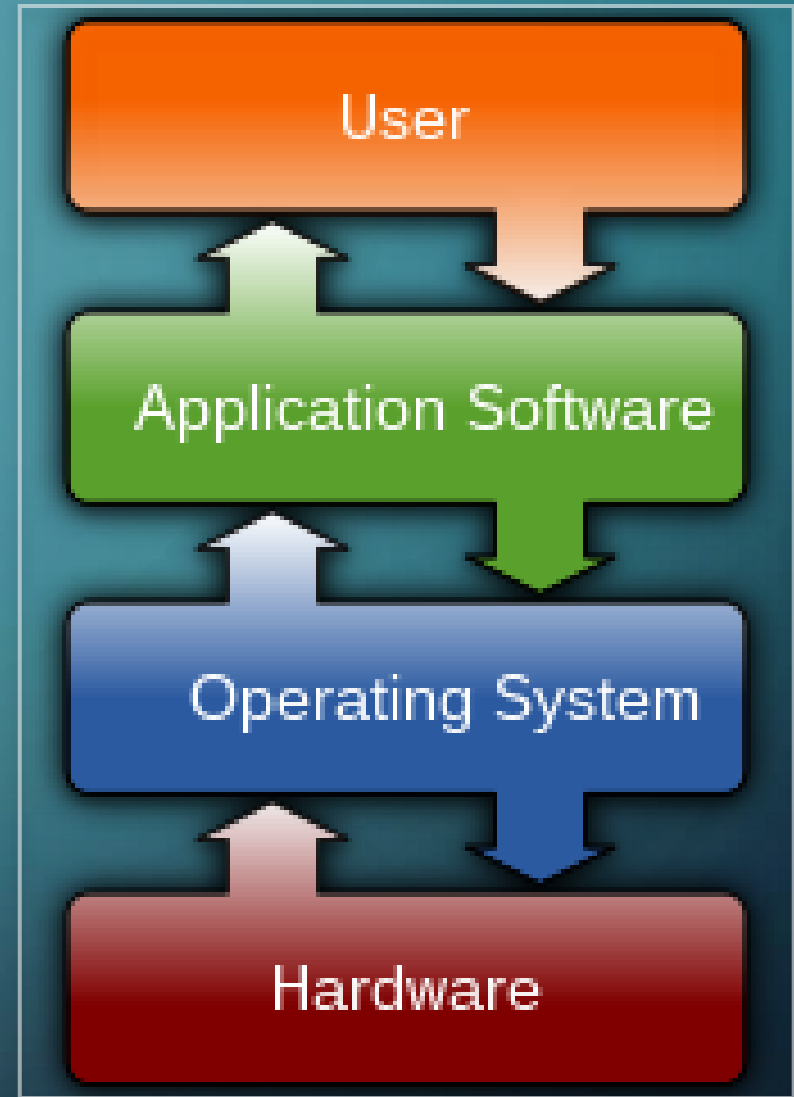
ADA LOVELACE

Augusta Ada King, Countess of Lovelace (née Byron; 10 December 1815 – 27 November 1852) was an English mathematician and writer, chiefly known for her work on Charles Babbage's proposed mechanical general-purpose computer, the Analytical Engine. She was the first to recognize that the machine had applications beyond pure calculation.



WHAT IS A SOFTWARE?



A software is a set of computer programs and associated documentation and data. This is in contrast to **hardware**, from which the system is built and which actually performs the work.

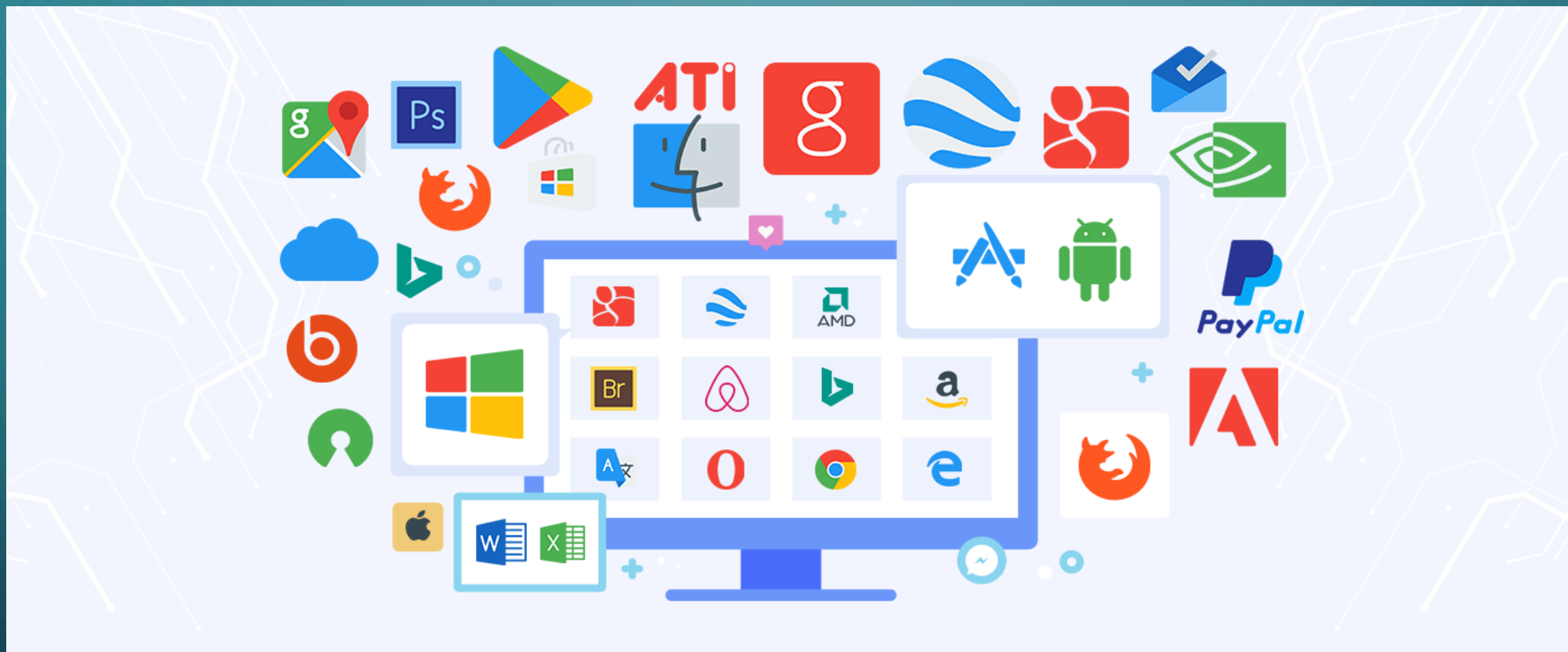


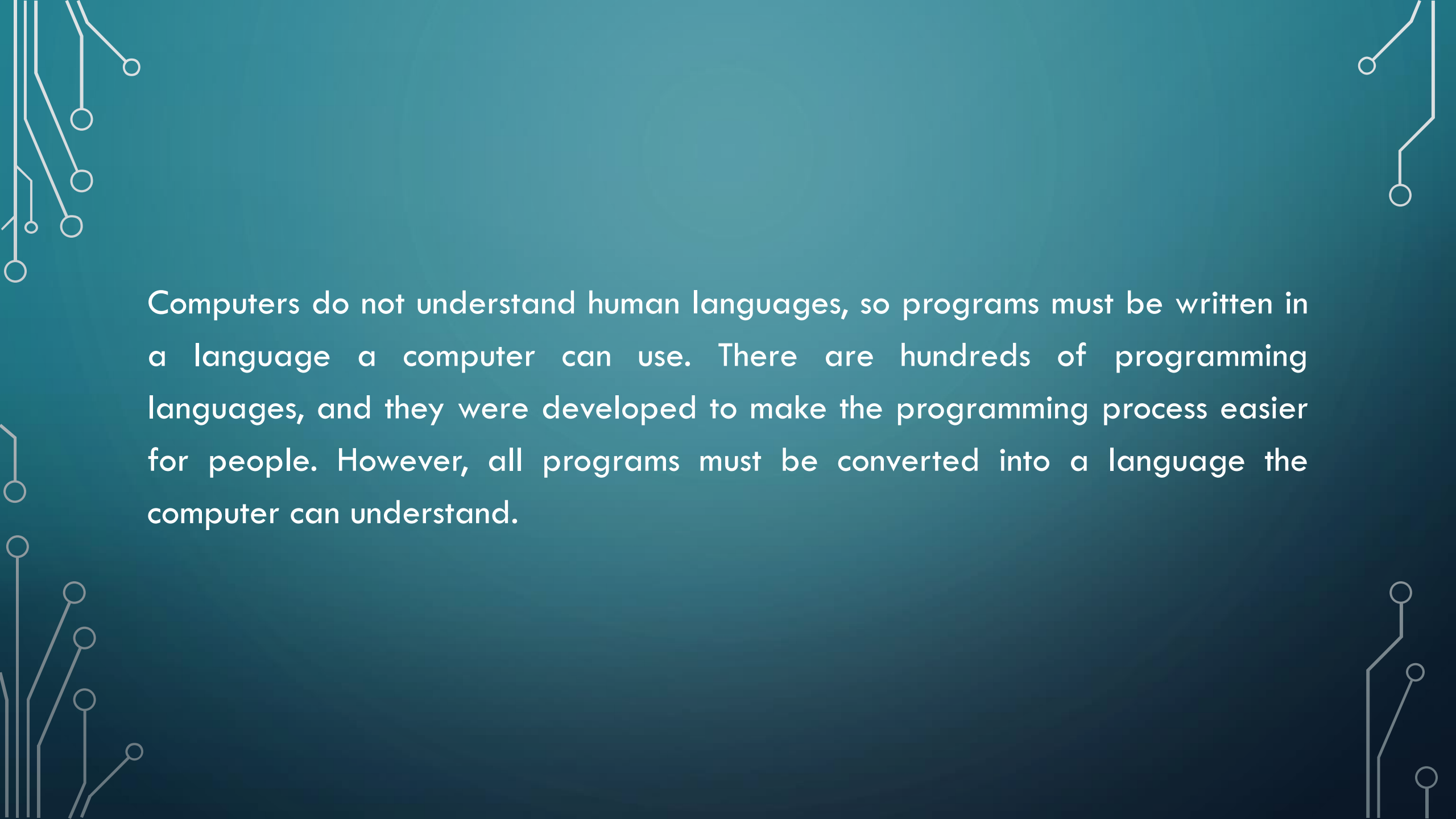


APPLICATION OR SOFTWARE?

An application, also referred to as an application program or application software, is a computer software package that performs a specific function directly for an end user or, in some cases, for another application. An application can be self-contained or a group of programs.

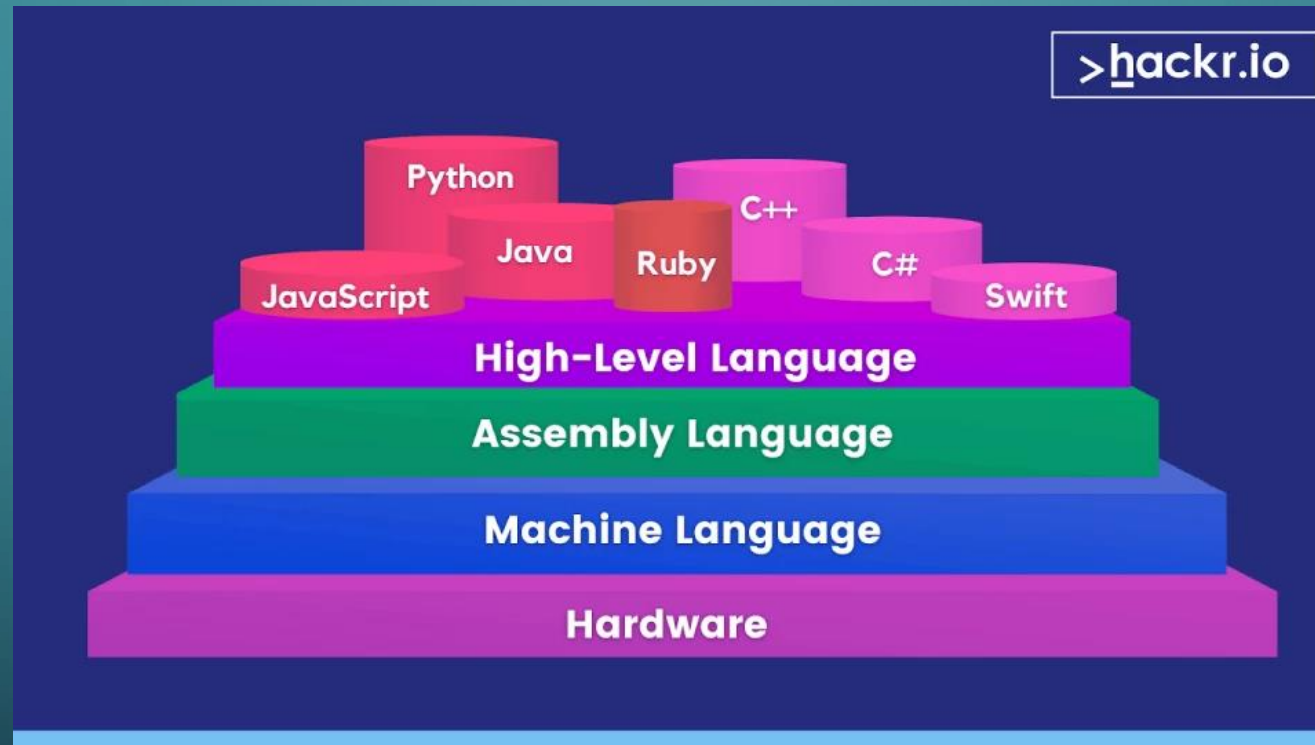




The background is a dark teal gradient. In the corners, there are white line-art illustrations of circuit boards or neural networks. These consist of vertical and horizontal lines of varying lengths, some ending in small open circles. The lines are more dense in the top-left and bottom-left corners, and more sparse in the top-right and bottom-right corners.

Computers do not understand human languages, so programs must be written in a language a computer can use. There are hundreds of programming languages, and they were developed to make the programming process easier for people. However, all programs must be converted into a language the computer can understand.

LEVEL OF ABSTRACTION IN PROGRAMMING



MACHINE LANGUAGE

Machine language instructions typically use some bits to represent operations, such as addition, and some to represent operands, or perhaps the location of the next instruction. Machine language is difficult to read and write, since it does not resemble conventional mathematical notation or human language, and its codes vary from computer to computer.

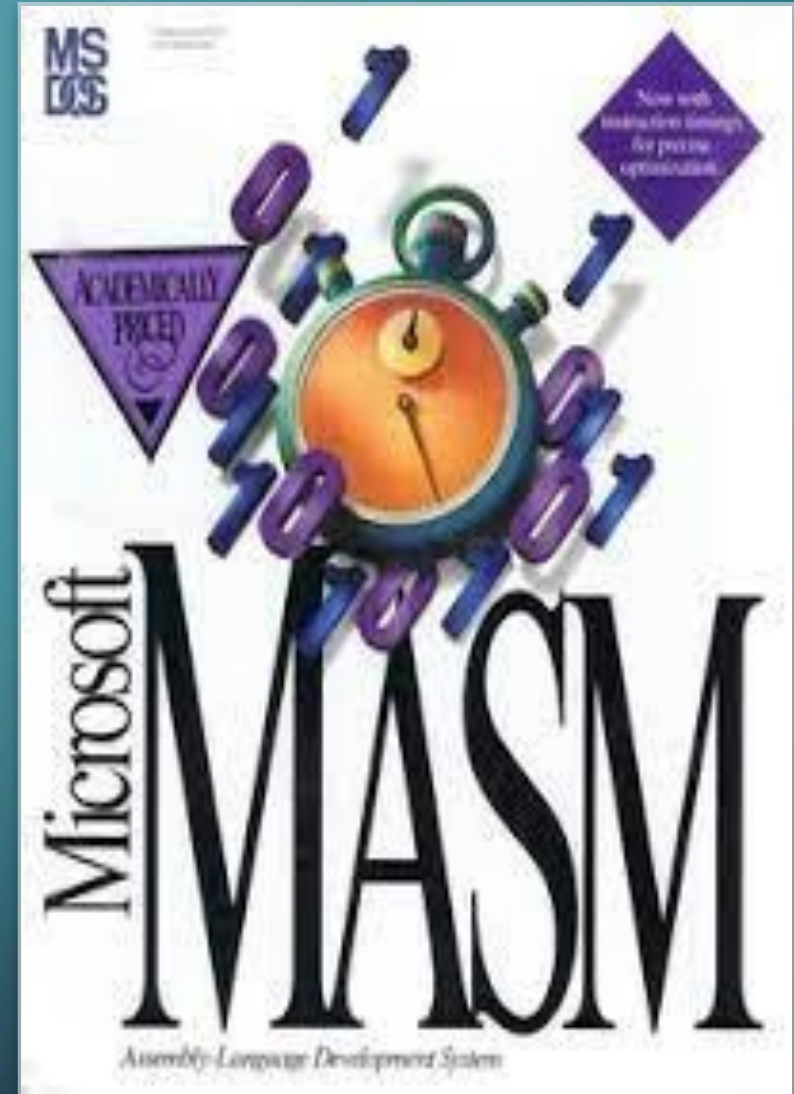
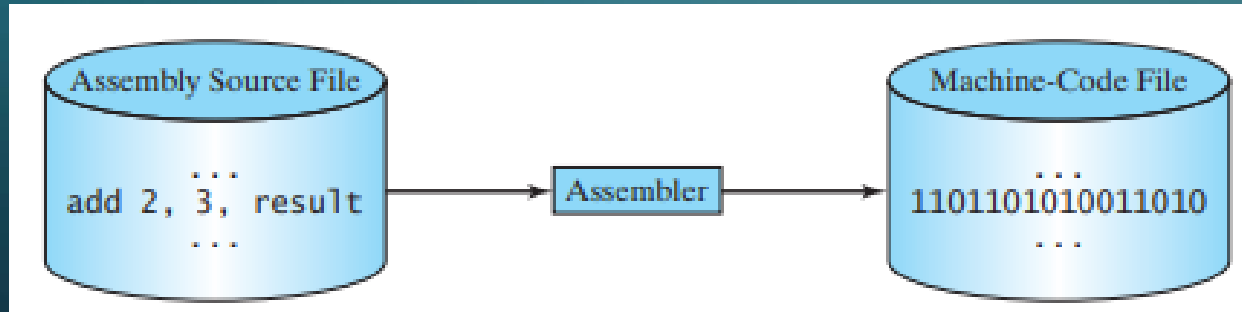
A computer's native language, which differs among different types of computers, is its machine language - a set of built-in primitive instructions. These instructions are in the form of binary code, so if you want to give a computer an instruction in its native language, you have to enter the instruction as binary code. For example, to add two numbers, you might have to write an instruction in binary code, like this:

1101101010011010

Decimal	Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

ASSEMBLY LANGUAGE

Assembly language (alternatively assembler language or symbolic machine code), often referred to simply as assembly and commonly abbreviated as ASM or asm, is any low-level programming language with a very strong correspondence between the instructions in the language and the architecture's machine code instructions.

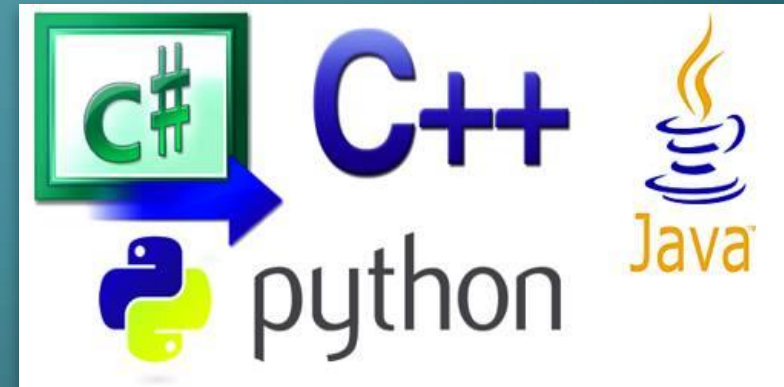


HIGH-LEVEL LANGUAGE

In the 1950s, a new generation of programming languages known as high-level languages emerged. They are platform-independent, which means that you can write a program in a high-level language and run it in different types of machines. High-level languages are English-like and easy to learn and use. The instructions in a high-level programming language are called statements. Here, for example, is a high-level language statement that computes the area of a circle with a radius of 5:

$$A = \pi r^2$$

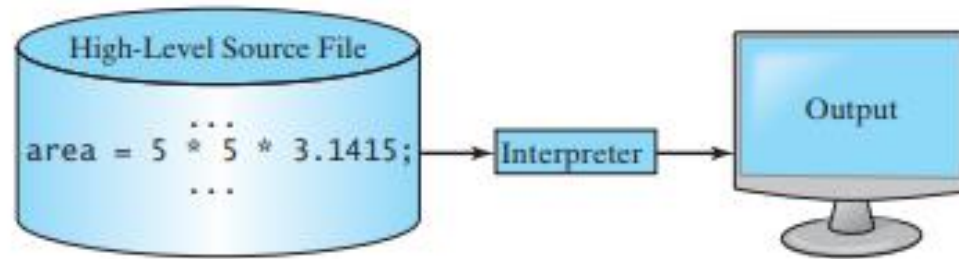
$$A = 3.1415 * 5 * 5$$



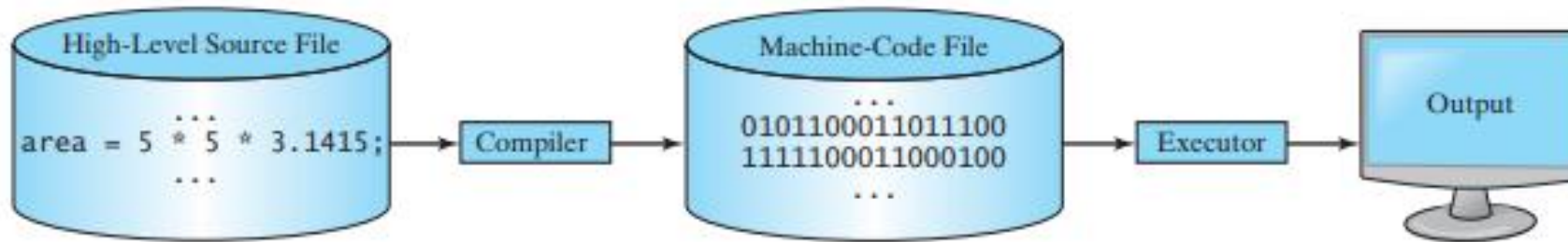
PROGRAMMING LANGUAGES

There are many high-level programming languages, and each was designed for a specific purpose. A program written in a high-level language is called a source program or source code. Because a computer cannot understand a source program, a source program must be translated into machine code for execution. The translation can be done using another programming tool called an interpreter or a compiler.

Language	Short Description
Ada	Mainly used for defense projects
BASIC	Beginner's All-purpose Symbolic Instruction Code.
C	C combines the power of an assembly language with the ease of use and portability of a high-level language.
C++	OO language based on C.
C#	Hybrid of java and C++.
COBOL	Common Business Oriented Language. Used for business applications.
FORTRAN	Formula Translation. Popular for scientific and mathematical applications.
Java	Widely used for developing platform-independent Internet applications.
Pascal	Structured general-purpose language primarily for teaching programming.
Python	A simple general-purpose scripting language good for writing short programs.
Visual Basic	Enables the programmers to rapidly develop Windows-based applications.



(a)



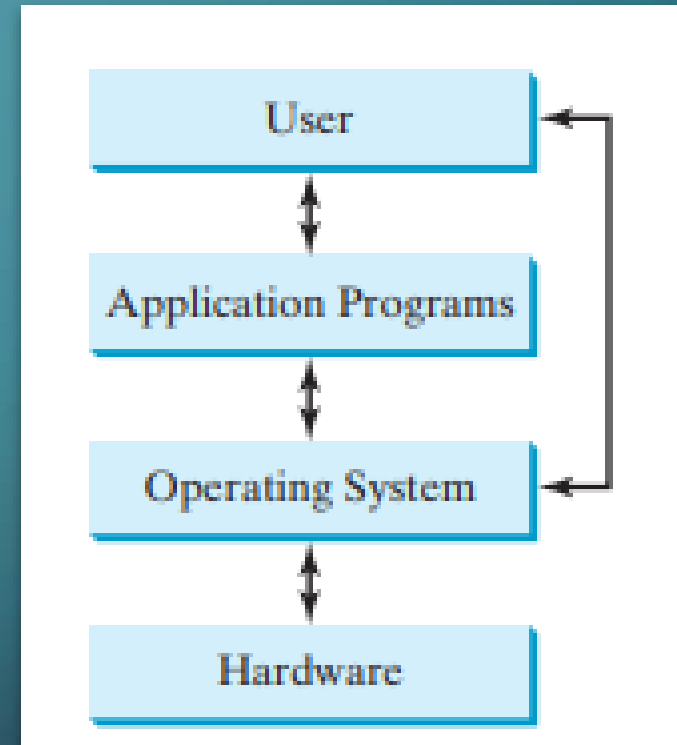
(b)

OPERATING SYSTEMS

The popular operating systems for general-purpose computers are Microsoft Windows, Mac OS, and Linux. Application programs, such as a Web browser or a word processor, cannot run unless an operating system is installed and running on the computer.

The major tasks of an operating system are:

- Controlling and monitoring system activities.
- Allocating and assigning system resources.
- Scheduling operations.



A decorative graphic consisting of white lines and circles on a dark teal background, resembling a circuit board or network diagram. The lines are of varying thickness and connect to small white circles at various points.



CONTROLLING AND MONITORING SYSTEM ACTIVITIES

Operating systems perform basic tasks, such as recognizing input from the keyboard, sending output to the monitor, keeping track of files and folders on storage devices, and controlling peripheral devices, such as disk drives and printers. An operating system must also ensure that different programs and users working at the same time do not interfere with each other. In addition, the OS is responsible for security, ensuring that unauthorized users and programs do not access the system.



ALLOCATING AND ASSIGNING SYSTEM RESOURCES

The operating system is responsible for determining what computer resources a program needs (such as CPU time, memory space, disks, input and output devices) and for allocating and assigning them to run the program.



SCHEDULING OPERATIONS

The OS is responsible for scheduling programs' activities to make efficient use of system resources. Many of today's operating systems support such techniques as multiprogramming, multithreading, and multiprocessing to increase system performance.

SCHEDULING OPERATIONS

- **Multiprogramming** enables multiple programs to run concurrently on a single CPU, making use of its idle time. This improves efficiency as the CPU is often waiting for tasks like data transfers or system resource responses. For instance, it allows you to edit a file with a word processor while your web browser downloads a file simultaneously.
- **Multithreading** allows a single program to execute multiple tasks at the same time. For instance, a word-processing program allows users to simultaneously edit text and save it to a disk. In this example, editing and saving are two tasks within the same application. These two tasks may run concurrently.
- **Multiprocessing**, or parallel processing, uses two or more processors together to perform subtasks concurrently and then combine solutions of the subtasks to obtain a solution for the entire task. It is like a surgical operation where several doctors work together on one patient.

LET'S WRAP IT UP

- What language does the CPU understand?
- What is an assembly language?
- What is an assembler?
- What is a high-level programming language?
- What is an interpreter and a compiler?
- What are the major responsibilities of an operating system?

ACTIVITY

- Divide the class into 10 groups.
- Research about the assigned algorithm.
- Make an outline of discussion to be submitted by the end of the period.
 - Definition and concept of the algorithm.
 - General idea of the algorithm.
 - Applications of the algorithm
 - Basic procedure (steps)
 - Complexity Analysis (e.g, time and space)

ACTIVITY

Outline Example

Title of Outline

I. Main topic

- A. Subtopic that gives more information about main topic I.
 - 1. Detail that gives more information about subtopic A.
 - 2. Detail for subtopic A.
- B. Subtopic for main idea I.
 - 1. Detail for subtopic B.
 - 2. Detail for subtopic B.

II. Main topic

- A. Subtopic that gives more information about main topic II.
 - 1. Detail that gives more information about subtopic A.
 - 2. Detail for subtopic A.
- B. Subtopic for main idea II.
 - 1. Detail for subtopic B.
 - 2. Detail for subtopic B.

III. Main topic

- A. Subtopic that gives more information about main topic III.
 - 1. Detail that gives more information about subtopic A.
 - a. Detail for detail 1.
 - b. Detail for detail 1.
 - 2. Detail for subtopic A.
- B. Subtopic for main idea III.
 - 1. Detail for subtopic B.
 - 2. Detail for subtopic B.

REFERENCES

- <https://en.wikipedia.org/wiki/Software>
- <https://www.techtarget.com/searchsoftwarequality/definition/application#:~:text=An%20application%2C%20also%20referred%20to,or%20a%20group%20of%20programs.>