

AI and ML for Chemists

Topic 2: Introduction to Python

Pre-lecture Explore

Dr. Ganna (Anya) Gryn'ova

Programming languages

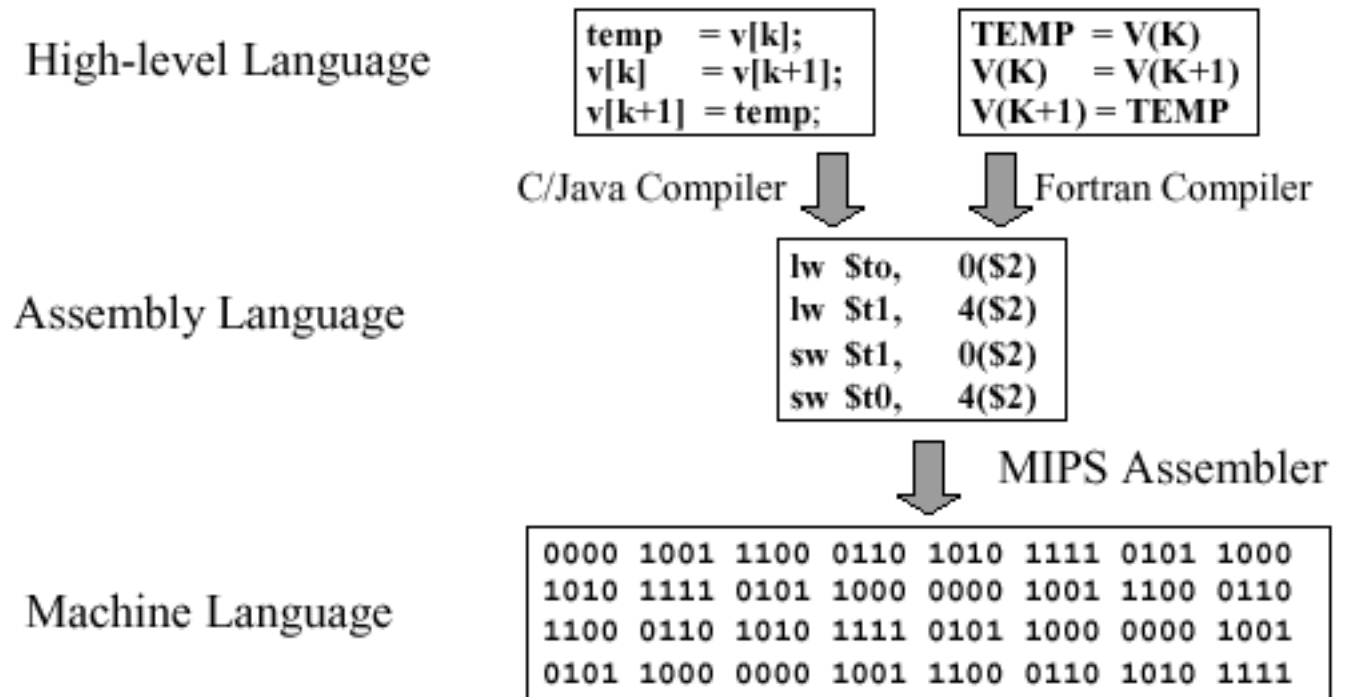
are systems of notation for writing computer programs.

Each programming language has its own command system executed by a computer.

High-level programming languages, such as Fortran, Pascal, C++, and Python use English-like commands for easier human understanding.

Assembly language converts symbolic high-level language into machine-readable code.

Machine language (code) is a binary code instructing the computer to execute the commands.



High-level programming languages

allow algorithms to be independent of specific computer hardware.

- **Procedural programming languages** (C, C++, Java, Pascal, Basic) follow a sequence of statements or commands in order to achieve a desired output. Each series of steps is called a *procedure*, and a program written in one of these languages will have one or more procedures within it.
- **Functional programming languages** (Haskell, Lisp, Scala, Elixir) focus on the output of mathematical *functions* and evaluations. Each *function*—a reusable module of code—performs a specific task and returns a result. The result will vary depending on what data you input into the function.
- **Object-oriented programming languages** (Java, Python, PHP) use *objects* and *classes* for code organisation. They treat a program as a group of objects composed of data and program elements, known as *attributes* and *methods*. Objects can be reused within a program or in other programs. This makes it a popular language type for complex programs, as code is easier to reuse and scale.
- **Scripting languages** (bash, Perl, Python, Ruby) allow automating repetitive tasks, managing dynamic web content, or supporting processes in larger applications.
- **Logic programming languages** (Prolog, Absys, Alma-0) express a series of facts and rules to instruct the computer on how to make decisions.

Features of programming languages

Syntax: The specific guidelines and arrangement that computer languages employ to produce code.

Data Types: The several types of values that may be kept in a program, including strings, integers, etc.

Variables: Named memory locations that can store values.

Control Structures: loops and conditional statements are examples of statements that regulate how a program executes.

Functions/Methods: Blocks of code that can be called from other parts of a program to perform specific tasks.

Memory Management: The process of allocating and deallocating memory for variables and data structures.

Semantics: The meaning or interpretation of a combination of symbols and statements within a programming language.

Parsing: The process of analysing code to determine its structure and meaning.

Markup and Control Language: The ability to add comments and other annotations to code to make it more readable and maintainable.

Abstraction: The ability to hide complex details and provide a simplified interface for users.

History of programming languages

Early 1940s: Konrad Zuse created Plankalkul, considered **the first programming language for computers**. It could store codes, enabling engineers to carry out routine, repetitive tasks far more efficiently and quickly.



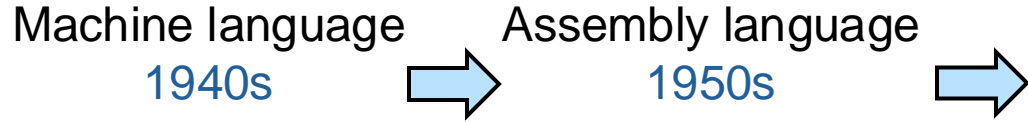
1936: Alan Turing's paper on a **universal machine** that could follow instructions. Turing eventually turned this groundbreaking idea into a plan for a computer powered by electricity that could run programs.



1843: world's **first published computer program** – Ada Lovelace's method for calculating Bernoulli numbers with Charles Babbage's analytical engine.



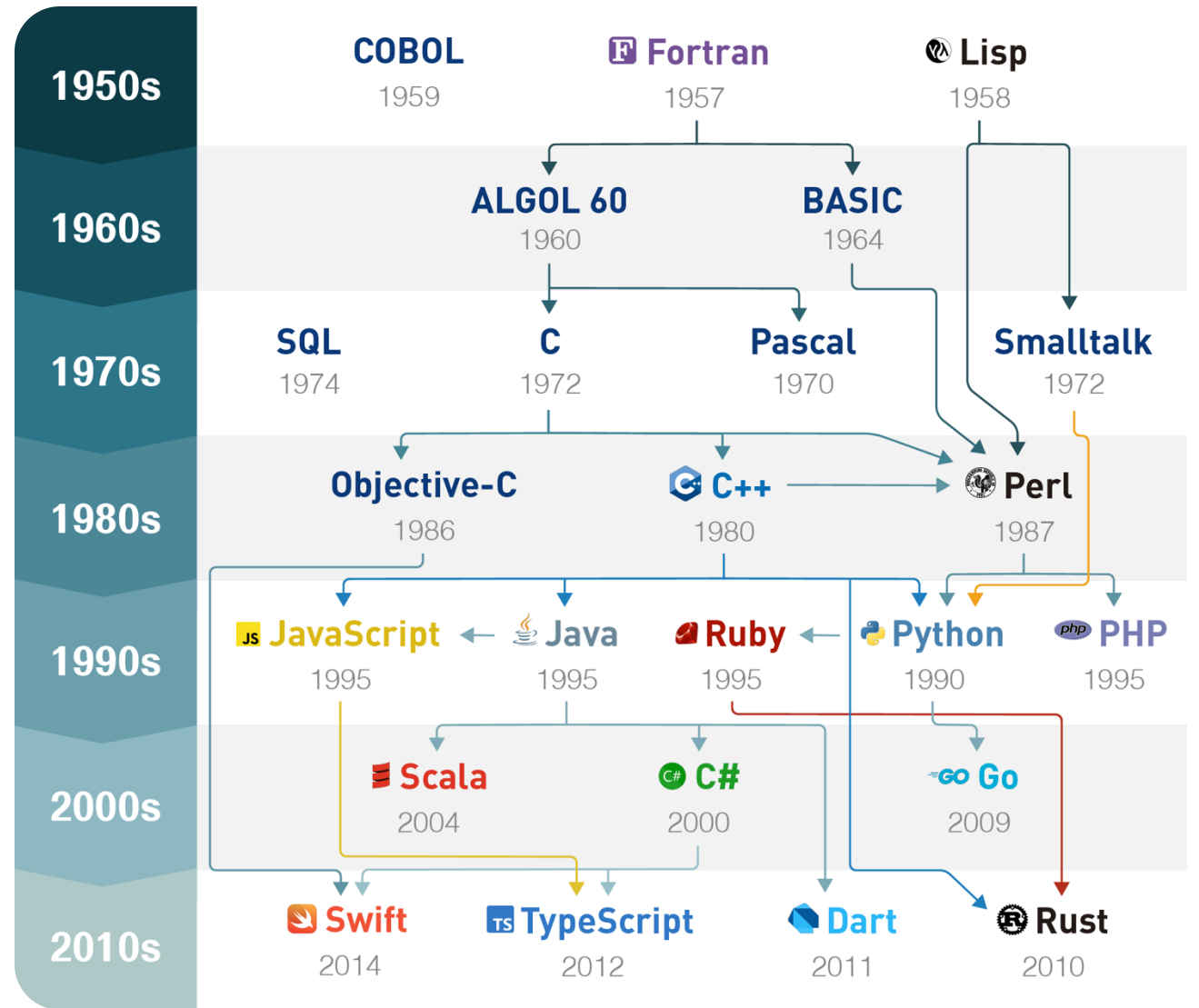
History of programming languages



Early 1940s: Konrad Zuse created Plankalkul, considered **the first programming language for computers**. It could store codes, enabling engineers to carry out routine, repetitive tasks far more efficiently and quickly.

1936: Alan Turing's paper on a **universal machine** that could follow instructions. Turing eventually turned this groundbreaking idea into a plan for a computer powered by electricity that could run programs.

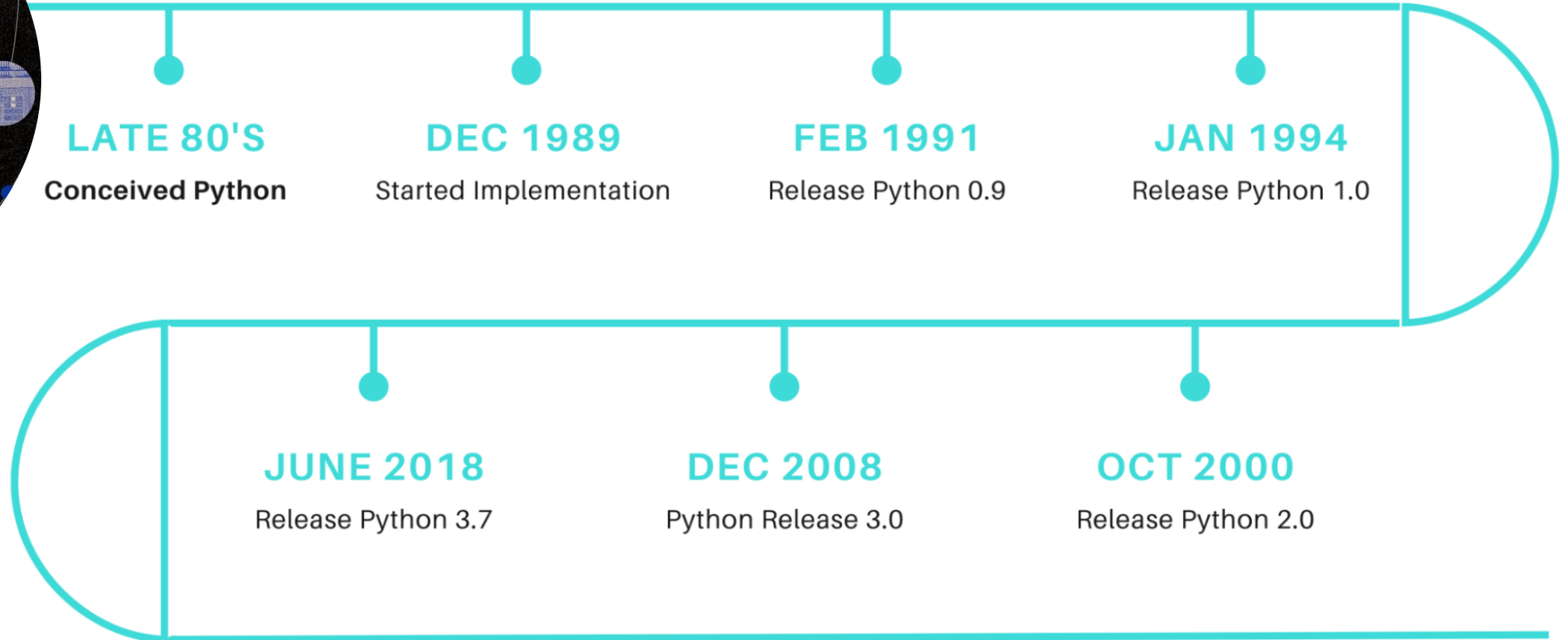
1843: world's **first published computer program** – Ada Lovelace's method for calculating Bernoulli numbers with Charles Babbage's analytical engine.



History of python™



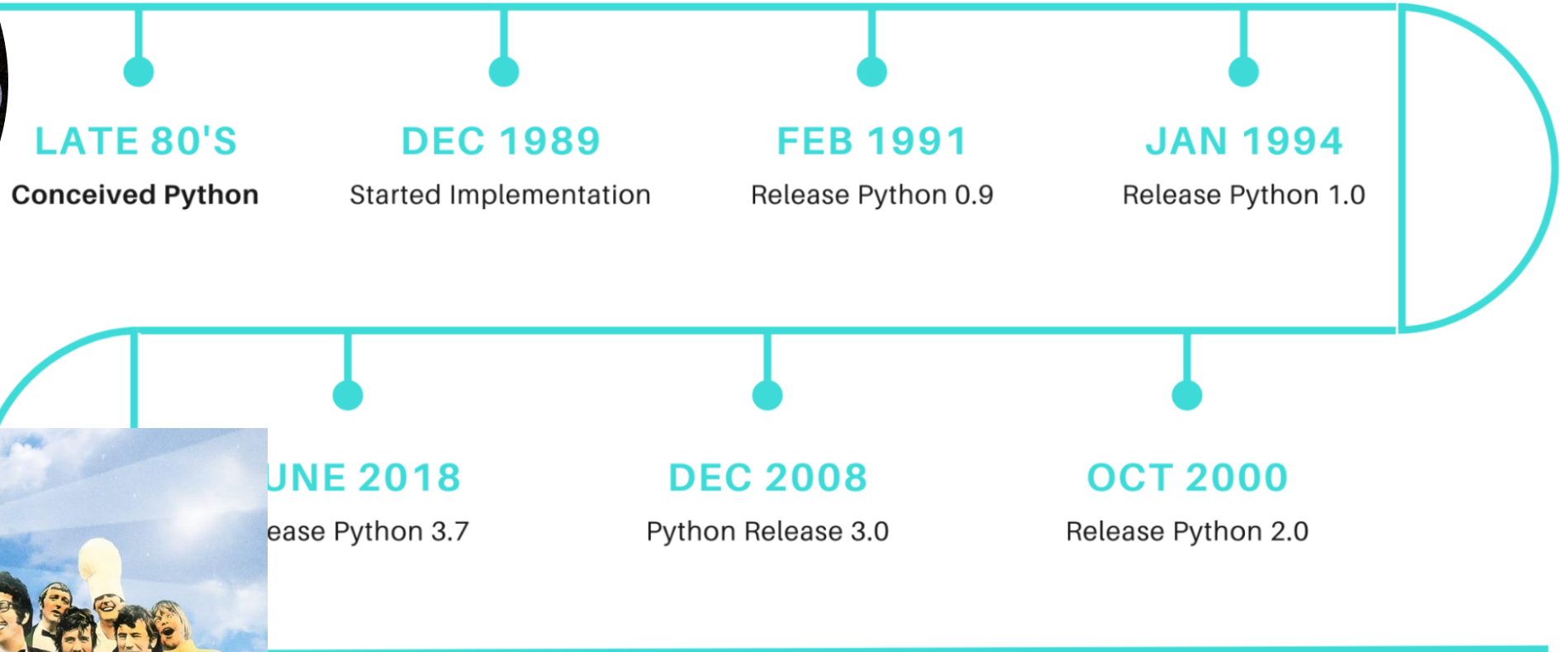
Guido van Rossum



History of python™



Guido van Rossum



Features of python™

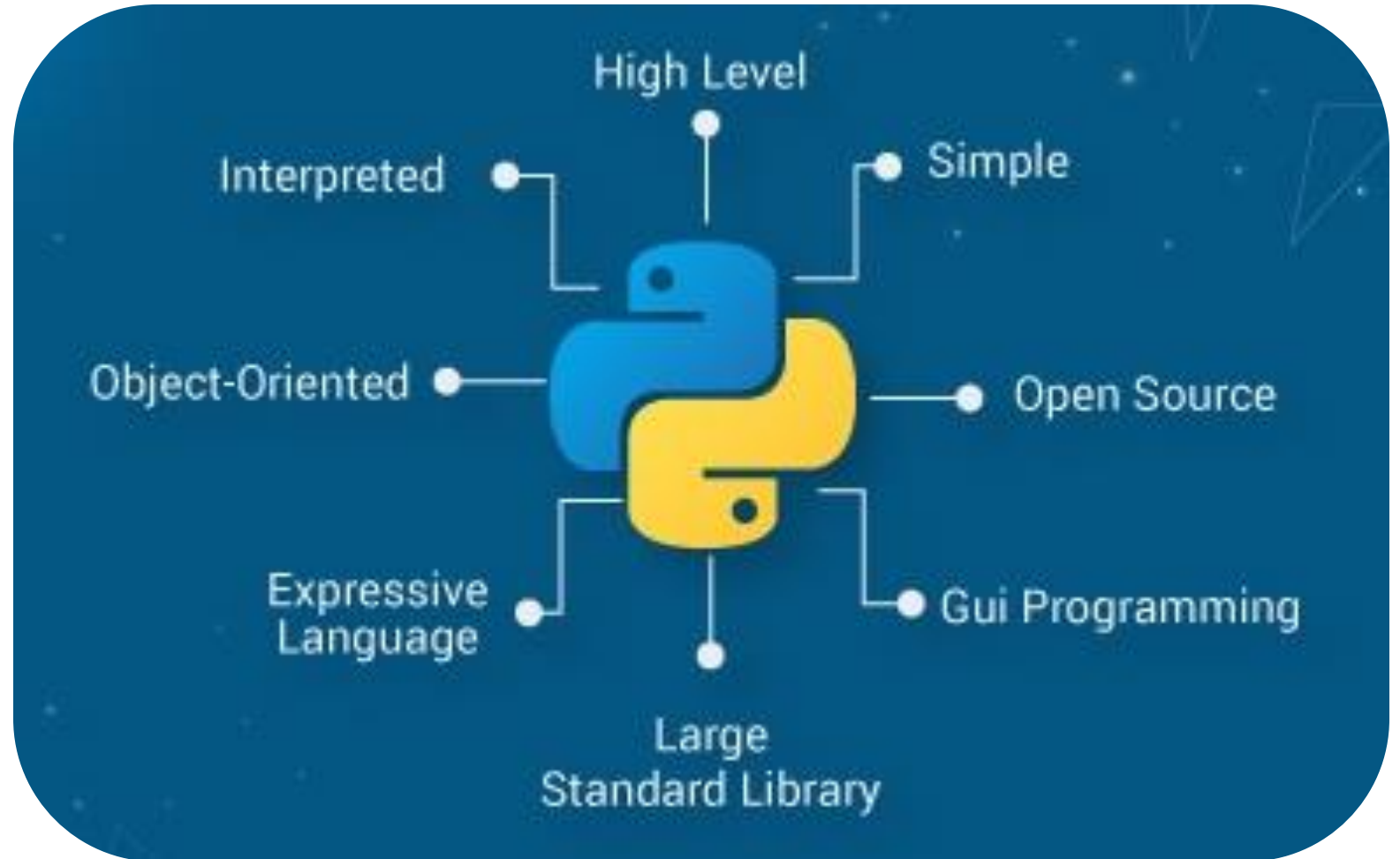
Python has an extremely **simple** syntax and is simple to read, comprehend, and write.

It is **compatible** with other languages of programming such as C, C++, and Java.

Python executes code **line-by-line** making it easy for the programmer to find the mistake that was made in the code.

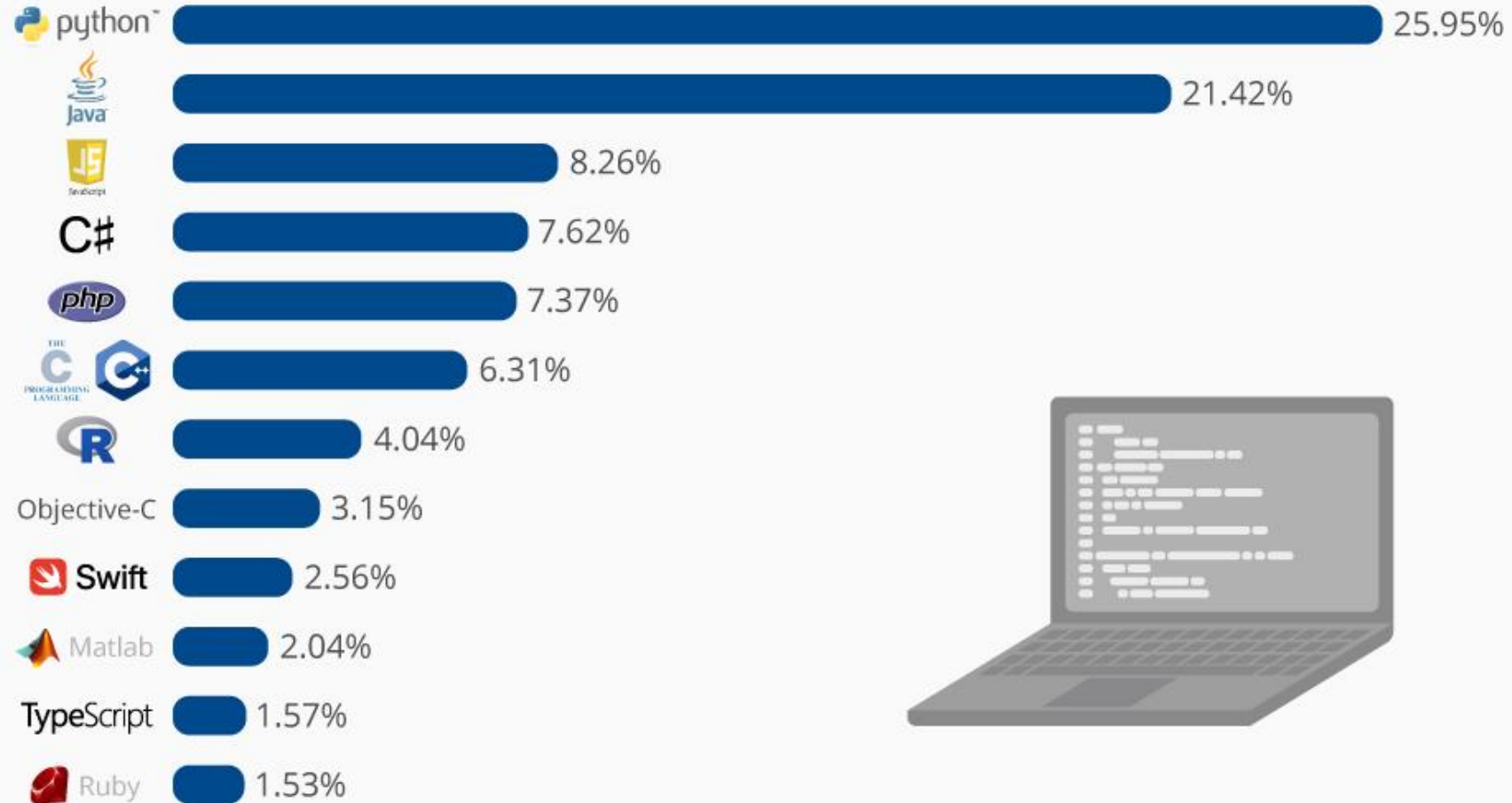
Python is a **platform-independent language**, meaning you can write your code once and execute it everywhere.

Not suitable for mobile applications.



The Most Popular Programming Languages

Share of the most popular programming languages in the world*



@StatistaCharts

* Based on the PYPL-Index, an analysis of Google search trends for programming language tutorials.

Source: PYPL

2019
statista

Uses of python™

BEST PYTHON LIBRARIES FOR MACHINE LEARNING

 TensorFlow	 PyTorch
 Keras	 orange
 matplotlib	 NumPy
 SciPy	 learn
 pandas	 theano



Basic structure

```
# This is a comment
```

```
import random # Importing a module
```

```
# Defining variables
```

```
x = 10
```

```
y = "Hello, World!"
```

```
z = True
```

```
# Performing arithmetic operation
```

```
result = x + 5
```

```
# Using if-else statement
```

```
if z:
```

```
    print(y)
```

```
else:
```

```
    print(result)
```

```
# Defining a function
```

```
def greet(name):
```

```
    print("Hello, " + name + "!")
```

```
# Using the function
```

```
greet("Alice")
```

Basic structure

Comments: Comments are used to explain the purpose of the code or to make notes for other programmers. They start with a '#' symbol and are ignored by the interpreter.

Import Statements: Import statements are used to import modules or libraries into the program. These modules contain predefined functions that can be used to accomplish tasks.

Variables: Variables are used to store data in memory for later use. In Python, variables do not need to be declared with a specific type.

Data Types: Python supports several built-in data types including integers, floats, strings, booleans, and lists.

Operators: Operators are used to perform operations on variables and data. Python supports arithmetic, comparison, and logical operators.

Control Structures: Control structures are used to control the flow of a program. Python supports if-else statements, for loops, and while loops.

Functions: Functions are used to group a set of related statements together and give them a name. They can be reused throughout a program.

Classes: Classes are used to define objects that have specific attributes and methods. They are used to create more complex data structures and encapsulate code.

Exceptions: Exceptions are used to handle errors that may occur during the execution of a program.