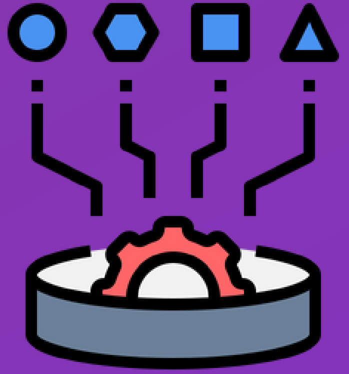
PYTHON VARIABLES

PYTHON VARIABLES

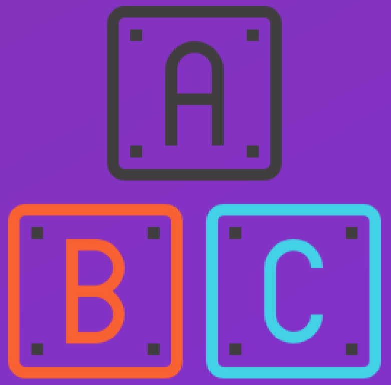
- Variables store data that the application needs to work with.

PYTHON VARIABLES



- Variables store data that the application needs to work with.

- Think of a variable as a labeled box in your computer's memory where you can store a value that your program can use later.



NAMING CONVENTIONS FOR VARIABLES (PEP 8)

STATIC VS. DYNAMIC TYPING

Python Programming

STATIC VS. DYNAMIC TYPING

In **statically typed languages** like Java, C++, or Go, the type of a variable is known at compile-time. You need to declare the type before using the variable.

STATIC TYPING KEY ADVANTAGES

- It helps catch type-related errors early, during the compilation process.

STATIC TYPING KEY ADVANTAGES

- It helps catch type-related errors early, during the compilation process.
- Programs often perform better at run-time because there’s no need for the interpreter to figure out the types.

PYTHON IS DYNAMICALLY TYPED!

STATIC VS. DYNAMIC TYPING

- **Static typing:** Common in languages like Java and C, catches type errors early and can boost run-time performance but slows down development.

STATIC VS. DYNAMIC TYPING

- **Static typing:** Common in languages like Java and C, catches type errors early and can boost run-time performance but slows down development.
- **Dynamic typing:** Makes Python more flexible and intuitive, but it can lead to run-time bugs and harder-to-diagnose issues.

STATIC VS. DYNAMIC TYPING

- **Static typing:** Common in languages like Java and C, catches type errors early and can boost run-time performance but slows down development.
- **Dynamic typing:** Makes Python more flexible and intuitive, but it can lead to run-time bugs and harder-to-diagnose issues.
- **Type annotations:** Provide a middle ground, helping you catch potential issues earlier without sacrificing Python’s dynamic nature.

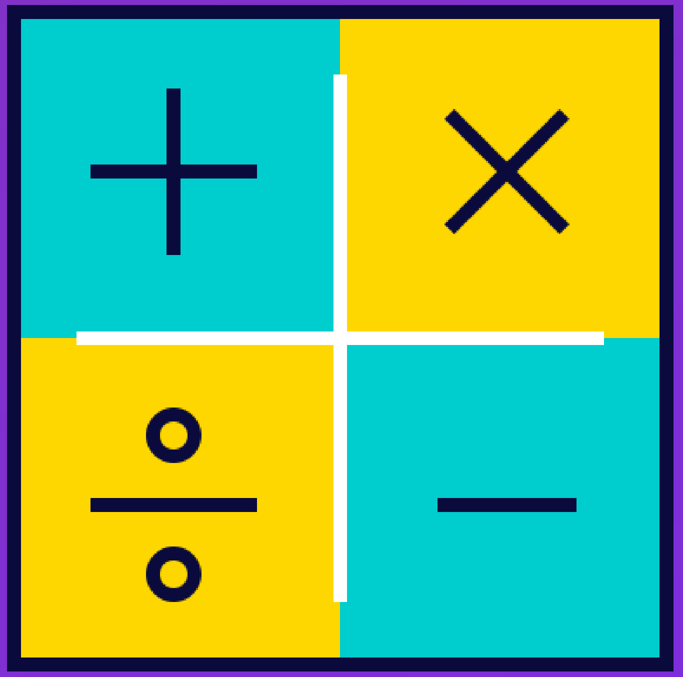
PYTHON OPERATORS

An operator is a symbol that tells Python to perform a specific operation on values.

PYTHON OPERATORS

An **operator** is a symbol that tells Python to perform a specific operation on values.

- Arithmetic Operators: + - * / // ** %
- Assignment Operators: = += -= *= /=
- Comparison Operators: == != > >= < <=
- Identity Operators: "is" "is not"
- Logical Operators: "and" "or" "not"



ORDER OF OPERATIONS (OPERATOR PRECEDENCE)

1. Exponentiation (**)
2. Multiplication (*) and division (/)
3. Addition (+) and subtraction (-)

ASSIGNMENT OPERATORS

- Equals (=)
- Plus equals (+=)
- Minus equals (-=)
- Star equals (*=)
- Slash equals (/=)
- Double stars equals (**=)
- Percent equals (%=)

COMPARISON OPERATORS

- Equal to (==)
- Not equal to (!=)
- Greater than (>)
- Greater than or equal to (>=)
- Less than (<)
- Less than or equal to (<=)

MUTABILITY vs. IMMUTABILITY

Python Programming

MUTABILITY vs. IMMUTABILITY

Mutable objects can be changed after they're created, while immutable objects can't.