Vocabulary Chapter 02

Constant: values that cannot be changed during the execution of an application

* Literal: value taken literally
* Numeric: number taken literally
* Unnamed: no identifier

Variable: a named memory location for which you can store a value that can change

* Camel Casing: a style for naming identifiers

-Upper: first letter of each word is capitalized

-Lower: first letter is lower & all subsequent words are capitalized

Data Type: describes the type of data that can be stored there, how much memory the item occupies, and what types of operations can be performed on the data.

* Primitive Types: simple and uncomplicated; building blocks for more complex types
* Reference Types: complex types; hold memory addresses

\*\***Type-Wrapper Classes:** the built in java classes for each of the primitive types;

-Includes methods that can process primitive type values

**Null:** no value, an unset reference

* Magic Number: a value that does not have an immediate, intuitive meaning or a number that cannot be explained without additional knowledge 🡪 example is an unnamed constant
* Garbage Value: a declared variable with no initialized value

**Scope:** the area which is visible to a program, in which you can refer to a data item using its simple identifier

* Something is in scope from declaration to the end of that block of code (curly braces)

Escape sequences: begins with backslash followed by a character; the pair represents a single character

Token: a set of characters that’s separated from the next whitespace

Prompt: message displayed for the user that requests input; strings in print()

Keyboard Buffer: location in memory where keyboard strokes are stored temporarily

**Arithmetic operators** (Binary operators: require two operands)

* Floating-Point division: either/both operands are floating-point values
* Integer division: both operands & result are integers; any fractional part of the result is lost
* Remainder Operator: most often w/ two integers, result is integer w/ value of remainder

**Type Conversion:** the process of converting one data type to another

* Unifying Type: type in which all operands in an expression are converted so that they are compatible with each other
* Implicit Conversion: automatically convertsType casting: forces a value of one data type to be used as a value of another type
* Cast operator: created by placing the desired result type in parenthesis