Python Programming Language

Lecture 2: Variables and Data Types

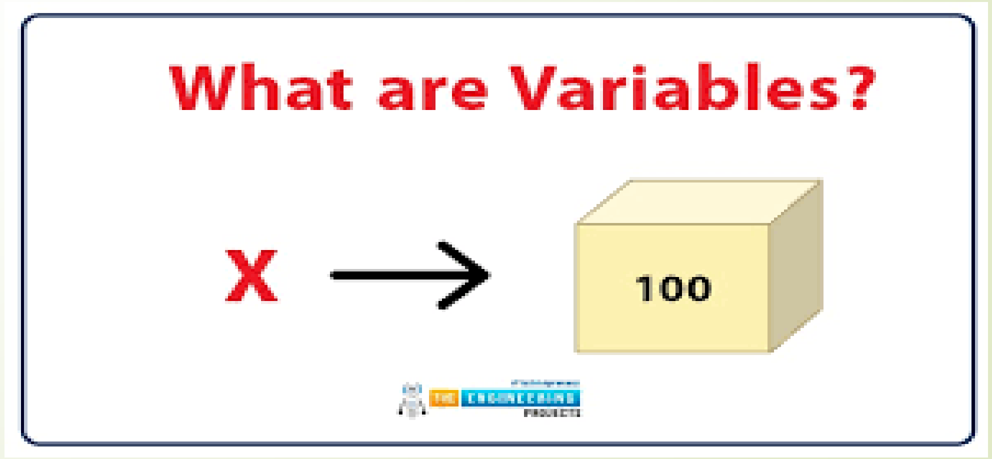
Hafsa Sultana
Computer Science and Engineering
Khulna University

Today’s Learning Objectives

- Variables,
- Naming Variables,
- Creating Variables,
- Data Types,
- More on Data Types,
- Expressions,
- Statements,
- Assignment Statements,
- Multiple Assignment,
- User Input,
- Advanced User Input,
- Variable Scope,
- Local Variables,
- Global Variables,
- Casting,
- Practical Examples,
- Practical Exercises,
- Debugging Tips,

What is a Variable

- Technically, variables act as an address where data is stored in memory.
- A container for storing data values.
- For example, x = "apples" can then be x = 5.
- [Think of it like a box with a label or address of a house to find the specific value.]



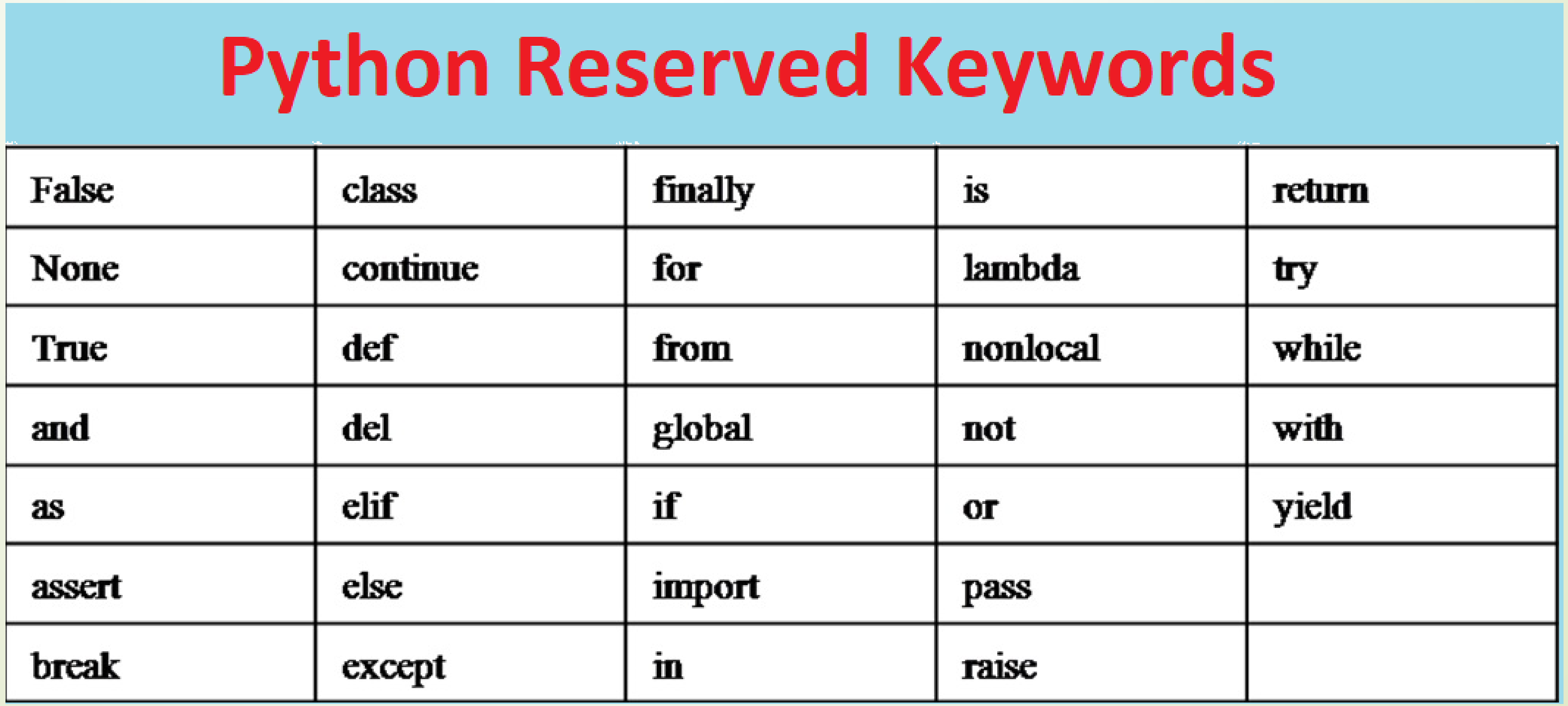
Naming Variables

Rules for Python variable naming:

- Must start with a letter or the underscore character
- Cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Variable names are case-sensitive (age, Age, and AGE are three different variables)
- Cannot use keywords.
- Examples: my_variable, age, _name

Keywords

Python has a set of keywords that are reserved words that cannot be used as variable names, function names, or any other identifiers:



Python Reserved Keywords

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

Creating Variables

Simply assign a value to a name.

Example:

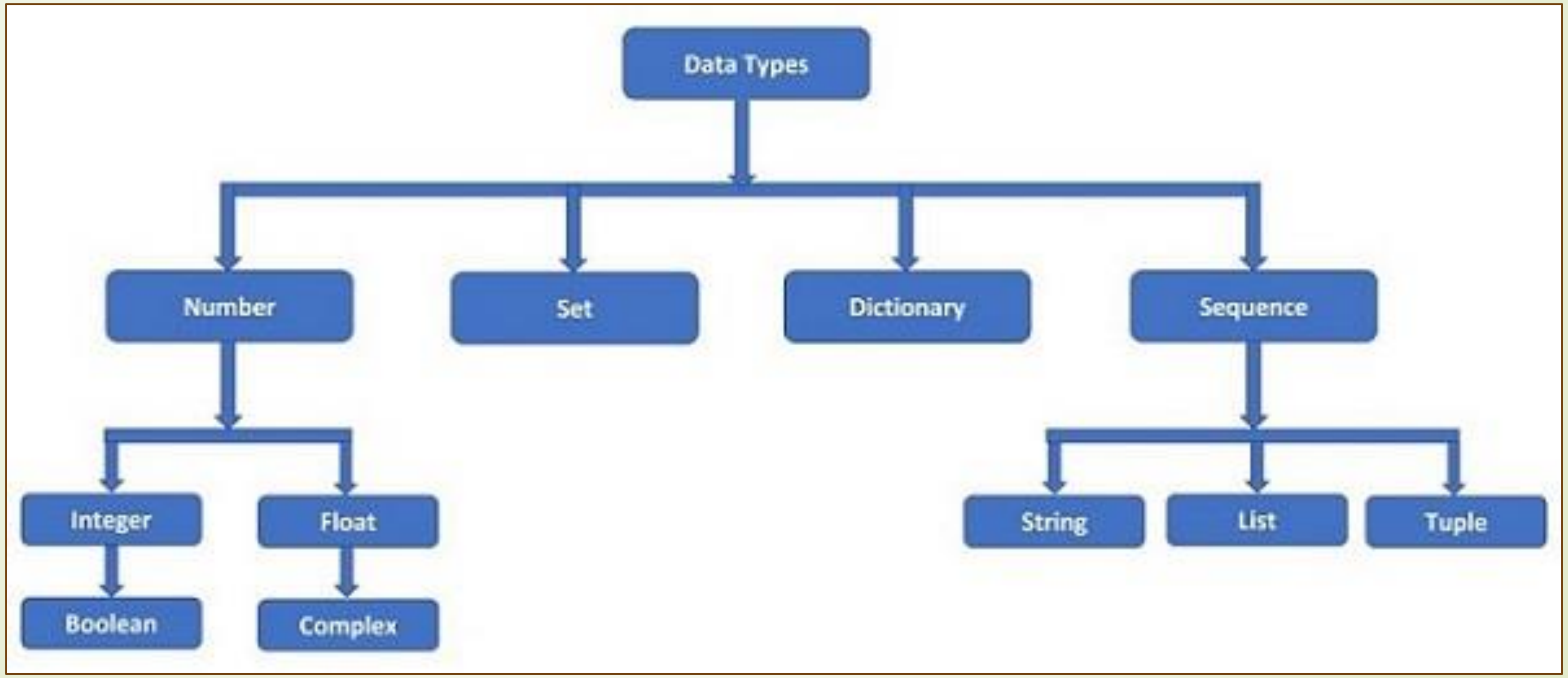
- x = 5,
- Num = 4.7
- name = 'Alice’
- 2myvar = “Kabir"
- my-var = “Aysha"
- my var = “Rajjak"

- myvar = " 'Alice "
- my_var = " 'Alice "
- _my_var = " 'Alice "
- myVar = " 'Alice "
- MYVAR = " 'Alice "
- myvar2 = " 'Alice "

Data Types

1. Integer: int (e.g., 5)
2. Floating-Point: float (e.g., 3.14)
3. Strings: str (e.g., 'hello')
4. Booleans: bool (e.g., True or False)

Data Types


Data Types
Number
Set
Dictionary
Sequence
Integer
Float
Boolean
Complex
String
List
Tuple

More Data Types

1. Lists: list (e.g., [1, 2, 3])
2. Dictionaries: dict (e.g., {'name': 'Alice', 'age': 10})
3. Tuples: tuple (e.g., (1, 2, 3))
4. Set: set (e.g., ({1,2,3})

Expressions

- A combination of values, variables, operators, and function calls that can be evaluated to produce a result.
- An expression always returns a value.
- Examples:
  - 2 + 3
  - x * y
  - len("hello")

Types of Expressions

- Arithmetic Expressions:

  Examples: 5 + 3, 10 - 2, 7 * 4, 8 / 2
- String Expressions:

  Examples: "Hello, " + "World!", "Python" * 3
- Boolean Expressions:

  Examples: 5 > 3, 10 == 10, x != y

Statements

- Instruction that Python can execute.
- Example: print('Hello, World!')

Type of Statements:

- Assignment Statements:
  - Assign a value to a variable.
  - Example: x = 10
- Conditional Statements:
  - Perform actions based on conditions.
  - Example: if x > 0:

```
if x > 0:
    print("Positive")
```

Statements

Type of Statements:

- Loop Statements:
  - Repeat actions
  - Example:
    ```
    for i in range(5):
        print(i)
    ```
- Functional Statements

  -
    ```
    def greet(name):
        print("Hello, " + name + "!")
    greet("Alice")
    ```
- Break, continue, import, return, del

Multiple Assignment

Assign values to multiple variables in one line.

Example:

```
a, b, c = 1, 2, 3
 print(a)
 print(b)
 print(c)
```

```
x, y, z = "Orange", "Banana", "Cherry"
print(x)
print(y)
print(z)
```

User Input

Use input() function to get input from the user.

Example:

```
name = input('What is your name? ')
print('Hello, ' + name)
```

Advanced User Input

Getting numbers from the user.

Example:

```
age = int(input('Enter your age: '))
height = float(input('Enter your height: '))
```

Variable Scope

Determines where a variable can be used.

Local vs Global variables.

Local Variables

Declared inside a function.

Only accessible within that function.

Example:

```
def my_function():
    x = 10
    print(x)
```

Global Variables

Declared outside any function.

Accessible anywhere in the code.

Example:

```
x = 10
def my_function():
    print(x)
```

Casting

Converting a variable from one type to another.

Example: str(3) converts integer 3 to string '3'

```
x = int(1)
y = int(2.8)
z = int("3")
print(x)
print(y)
print(z)
```

```
x = float(1)     # x will be 1.0
y = float(2.8)   # y will be 2.8
z = float("3")   # z will be 3.0
w = float("4.2") # w will be 4.2
print(x)
print(y)
print(z)
print(w)
```

```
x = str("s1") # x will be 's1'
y = str(2)    # y will be '2'
z = str(3.0)  # z will be '3.0'
print(x)
print(y)
print(z)
```

Practical Examples

Create variables of different types.

Example:

```
age = 10
pi = 3.14
name = 'Alice'
is_student = True
```

Practical Exercises

1. Concatenate Two Strings

   Input: Enter the first word: Hello

   Enter the second word: World

   Output: HelloWorld

2. Print the First and Last Character

   Input: Enter a word: Python

   Output: First character: P

   Last character: n

Practical Exercises

3. Length of a String

    Input: Enter a sentence: Python is great!

    Output: 16

4. Average of Three Numbers

    Input: Enter the first number: 3

    Enter the second number: 4

    Enter the third number: 5

    Output: 4.0

Practical Exercises

5. Area of a Rectangle

Input: Enter the width of the rectangle: 5

Enter the height of the rectangle: 10

Output: The area is: 50

6. Celsius to Fahrenheit (F = C * 9/5 + 32)

Input: Enter temperature in Celsius: 0

Output: The temperature in Fahrenheit is: 32.0

Practical Exercises

7. Draw:

```
*
**
***
****
*****
```

8. Draw:

```
    *
   ***
  *****
 *******
*********
```

7 Sln:

```
print("*")
print("**")
print("***")
print("****")
print("*****")
```

8 Sln:

```
print("    *    ")
print("   ***   ")
print("  *****  ")
print(" ******* ")
print("*********")
```

9. Draw:

```
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
```

9 Sln:

```
print("    *    ")
print("   ***   ")
print("  *****  ")
print(" ******* ")
print("*********")
print(" ******* ")
print("  *****  ")
print("   ***   ")
print("    *    ")
```

Debugging Tips

Use print statements to debug.

Check variable values and types.

Any Question?

References

- https://www.w3schools.com/python/
- https://www.futurelearn.com/info/courses/introduction-to-programming-with-python-fourth-rev-/0/steps/264867