In the name of Allah

Hello Everybody

Let’s get started!

Python as a Language

Parseltongue

Parseltongue is the language of serpents and those who can converse with them. An individual who can speak Parseltongue is known as a Parselmouth. It is a very uncommon skill, and may be hereditary. Nearly all known Parselmouths are descended from Salazar Slytherin.

Python

Python is the language of Python Interpreter and those who can converse with them. An individual who can speak Python is known as a Pythonista. It is a very uncommon skill, and may be hereditary. Nearly all known Pythonistas are descended from Guido van Rossum.

Why Python?

Guido van Rossum

He is a Dutch programmer best known as the creator of the Python programming language.

Syntax Errors

When you make a mistake, the computer does not think you are “cute”. It says “syntax error”, given that it knows the language and you are just learning it.

Elements of Python

1. **Vocabulary / Words -** variables and Reserved words
2. **Sentence structure -** valid syntax patterns
3. **Story structure -** constructing a program for a purpose

Python Scripts

Types of control structures

1. Sequence
2. Branching (Selection)
3. Loop (Repetition)

Sequence structure

Constants

Fixed values such as numbers, letters, and strings, are called “constants” because their value does not change.

Reserved Words

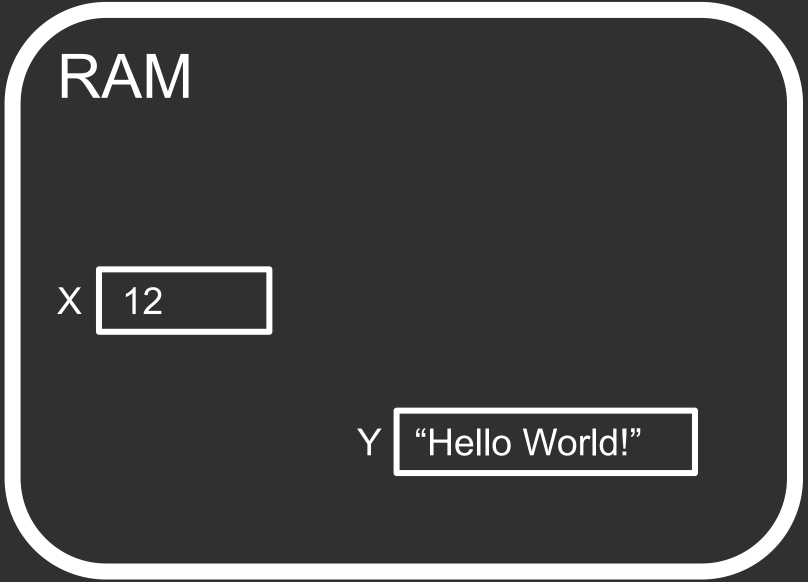
Keywords are predefined, reserved words used in Python programming that have special meanings to the interpreter.

Variables

A variable is a named place in the memory where a programmer can store data and later retrieve the data using the variable “name”

Variable

```
X = 12
Y = “Hello World!”
```



In python, you can name your variables whatever you want, with some restrictions:

1. Variables must start with a letter or underscore

_cats	2cats

2. The rest of the name must consist of letters, numbers, or underscores

cats2	hey@you

3. Names are case-sensitive

Cats != CATS	Cats != cats

Most python programmers prefer to use standard style conventions when naming things :

Most variables should be snake_case (underscore between words)

Most variables should be lowercase, with some exceptions :

CAPITAL_SNAKE_CASE usually refers to constants (e.g. PI = 3.14)

UpperCamelCase usually refers to a class

variables that start and end with two underscores (called “dunder” for double underscore) are supposed to be private or left alone

__no_touchy__

Comments

A comment is a piece of text that provides explanatory notes or remarks within the code. Comments are used to make the code more understandable for other developers or for your future self. They are entirely ignored by the Python interpreter and have no impact on the program's execution

Data Type

It defines what type of data we are going to store in a variable. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters.

Static vs dynamic

Data Types in Python

1. Integer

Int, or integer, is a whole number, positive or negative, without decimals, of unlimited length

2. Float

It is a data type used to represent floating-point numbers. A floating-point number is a number that has a decimal point in it or is written using scientific notation (e.g., 3.14, -2.718, 1.0e-5).

3. String

It is a data type used to represent a sequence of characters. A string can contain letters, numbers, symbols, and even white spaces. Strings are created by enclosing the characters within either single quotes (' '), double quotes (" "), or triple quotes (""" """ or ''' ''')

String Concatenation

Concatenation is combining multiple strings together. In python you can do this simply with the “+” operator.

```
str_one = “hello”
str_two = “world”
str_three = str_one + “ “ + str_two + “!”   #hello world!
```

F-Strings (Python 3.6+)

In Python, an f-string (formatted string literal) is a way to embed expressions inside string literals. It provides a concise and convenient way to format strings in a more readable and expressive manner.

```
x = 10
formatted = f”I’ve told you {x} times already!”
```

None

The None type represents the absence of a value. It is a special constant that is often used to indicate the absence of a meaningful result or the lack of a value for a variable or expression.

Converting Data Types

You can convert variables by using the of the built in type as a function

```
decimal = 12.35363
integer = int(decimal)          # 12
string = str(integer)           # ’12’
decimal = float(integer)        # 12.0
```

Operators

Operators are symbols or special characters that are used to perform operations on variables or values. Python supports various types of operators that serve different purposes, such as arithmetic operations, comparison, logical operations, assignment, membership testing, and identity testing.

Arithmetic Operators

Operator	Operator	Description	Example
+	Addition	Adds together two values	x + y
-	Subtraction	Subtracts one value from another	x - y
*	Multiplication	Multiplies two values	x * y
/	Division	Divides one value by another	x / y
%	Modulus	Returns the division remainder	x % y
**	Exponent	Left operand raised to power of right	x ** y
//	Floor division	Division that results into whole number	x // y

Assignment Operators

Operator	Example	Same As	Operator
=	x = 5	x = 5	=
+=	x += 3	x = x + 3	+=
-=	x -= 3	x = x - 3	-=
*=	x *= 3	x = x * 3	*=
/=	x /= 3	x = x / 3	/=
%=	x %= 3	x = x % 3	%=

Comparison Operators

Operator	Name	Example
==	Equal to	x == y
!=	Not equal to	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

Logical Operators

Operator	Name	Description	Example
and	**Logical and**	**Returns true if both statements are true**	x < 5 and x < 10
or	**Logical or**	**Returns true if one of the statements is true**	x < 5 or x < 4
not	**Logical not**	**Reverse the result, returns false if the result is true**	not x < 10

I/O in Python

Input and output (I/O) in Python refer to the process of getting data into a program (input) and displaying or writing data from the program to the external world (output). Python provides built-in functions and modules to handle input and output operations easily.

Input

To take input from the user, Python provides the “input()” function. The “input()” function waits for the user to enter a line of text and returns it as a string. You can store the input in a variable for further processing.

```
name = input("Enter your name: ")
print("Hello, " + name)
```

Output

To display output to the console or terminal, Python uses the “print()” function. You can pass one or multiple expressions separated by commas to the “print()” function. It will convert the expressions to strings and display them on the screen.

```python
print("Hello, World!")
x = 10
y = 20
print("The sum of", x, "and", y, "is", x + y)
```

Exercise 1

Write a program to prompt the user for his/her name and says hello.

```
Hello,
What is your name: matin

Nice to meet you, matin.
```

Exercise 2

Write a program to prompt the user for hours and rate per to compute gross pay.

```
Enter Hours: 35
EnterRate: 2.75

Pay: 96.25
```

That’s all for today!