

Programming in Python

Part Two: Basics

Python Basics

Indentation

.... four spaces in
Python

Python Operators

Table 1-1: Math Operators from Highest to Lowest Precedence

Operator	Operation	Example	Evaluates to...
**	Exponent	2 ** 3	8
%	Modulus/remainder	22 % 8	6
//	Integer division/floored quotient	22 // 8	2
/	Division	22 / 8	2.75
*	Multiplication	3 * 5	15
-	Subtraction	5 - 2	3
+	Addition	2 + 2	4

Demo

precedence.py

Comparison Operators

Table 2-1: Comparison Operators

Operator	Meaning
==	Equal to
!=	Not equal to
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to

Let's brainstorm!

What's the output of:

- `42 == '42'`
- `42 == 42.0`
- `42.0 == 000000000000000042.000`

Unveil

- `42 == '42'` False
- `42 == 42.0` True
- `42.0 == 0000000000000000042.000` True

We need
definitions to
hold our data

Primitive Data Types

Type	Representation
int	Number
float	Number with a decimal separator (.)
boolean	True/False
str	" or "" or "" or ""

Primitive Data Types Example

count = 90

price = 90.0

greet = 'Hello'

name = "James Bond"

poem = ""

A thing of beauty is a joy forever. Its loveliness increases. It will never pass into nothingness.

""

state = True

flag = False

Demo

string.py, boolean.py

What's a variable?

It's where we keep our stuff.

A variable is something you want the computer to remember while your program is running. When Python “remembers” something, it's storing that information in the **computer's memory**.

Python can remember values of several types, including **number** values (like 7, 42, or even 98.6) and **strings** (letters, symbols, words, sentences, or anything you can type on the keyboard and then some).

Demo

variable.py

Variable Names

Table 1-3: Valid and Invalid Variable Names

Valid variable names	Invalid variable names
balance	current-balance (hyphens are not allowed)
currentBalance	current balance (spaces are not allowed)
current_balance	4account (can't begin with a number)
_spam	42 (can't begin with a number)
SPAM	total_\$um (special characters like \$ are not allowed)
account4	'hello' (special characters like ' are not allowed)

We need to
organize our data

Control Flow

Conditions

Conditions always evaluate down to a Boolean value, True or False.

Blocks of Code

Lines of Python code can be grouped together in blocks.

Control Flow

Flow **based on** Condition

if, elif, else

Flow **until the** Condition

for, while

Flow **break** and Flow **continue**

break, continue

Demo

if-elif-else.py, while.py,
break-continue.py, for.py

Python Data Structures

Data Structures

Type	Representation
list	[] elements separated with ,
tuple	() elements separated with ,
dict	{:} elements separated with , and every element should have key:value
set	{ } elements separated with ,

About Data Structures

Lists are **ordered sequences** of values.

Tuples are **ordered, immutable** sequences of values.

Sets are **unordered bag** of values.

Dictionaries are unordered bag of **key-value** pairs.

Data Structures Example

colors = ["yellow", "blue", "green"]

shapes = ("square", "rhombus", "circle")

friends = {"James", "Tom", "George"}

baseball_team = {'Colorado' : 'Rockies', 'Boston' : 'Red Sox', 'Minnesota': 'Twins'}

Demo

`list.py, tuple.py, dictionary.py`

Are sets
immutable?

Type Conversion

int \Rightarrow float

list \Rightarrow tuple

tuple \Rightarrow list

str \Rightarrow int (??)

...

Functions

What's a function?

You're already familiar with the `print()`, `input()`, and `len()` functions. Python provides several builtin functions like these, but you can also write your own functions.

A function is like a mini-program within a program which is used to encapsulate repetitive code.

Structure

```
def function_name(param1, param2):
```

```
    ...
```

```
function_name(arg1, arg2)
```

The function `function_name` only runs when it's called.

Local & Global Scope

Parameters and variables that are assigned in a called function are said to exist in that function's **local scope**. Variables that are assigned outside all functions are said to exist in the **global scope**.

“return”

- The return keyword
- The value or expression that the function should return

Demo

function.py

Recursive Function

A recursive definition is one in which the defined term appears in the definition itself.

Factorial: 2's factorial is $1*2 = 2$, 5's factorial is $1*2*3*4*5 = 120$

Demo

factorial.py

Code??

User Input: Number of rows

Example: If the user input is 5, then the output is:

*

* *

* * *

* * * *

* * * * *

Code??

User input: List of numbers

Write a program that prints out all the elements of the list that are less than 5.

For example, if the input is 1, 2, 10, the output is [1, 2]
