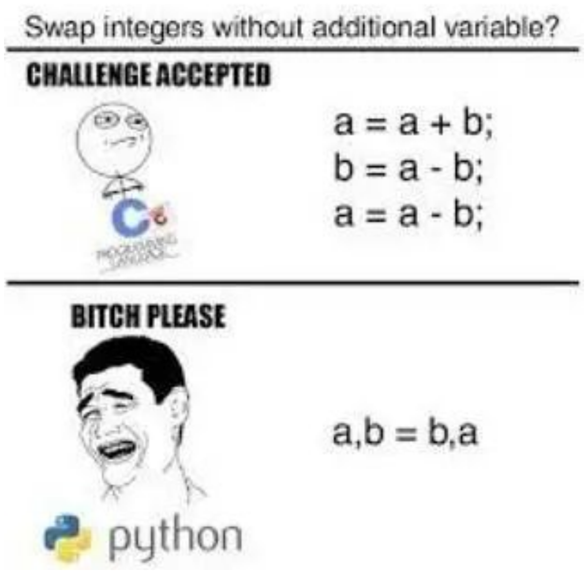
Python

Kameswari Chebrolu

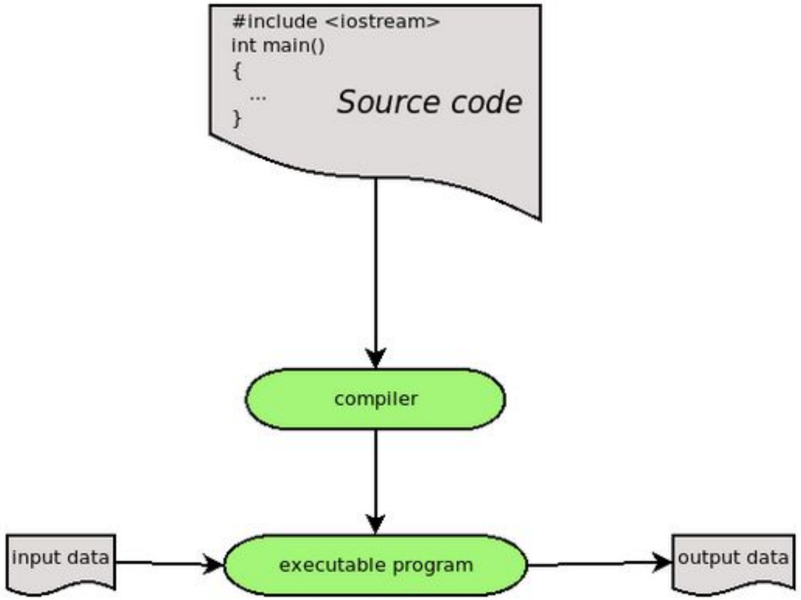


Python is the easier language to learn. No brackets, no main.
You get errors for writing an extra space

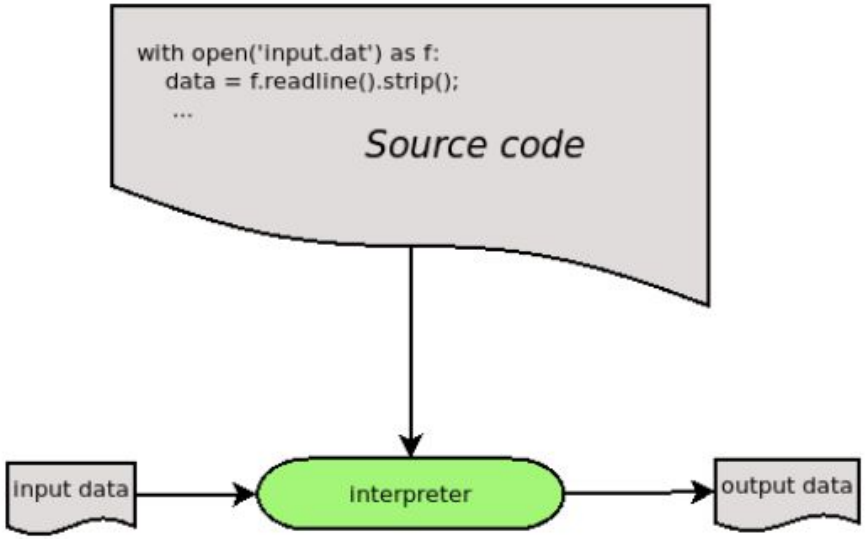
Background: Compilers and Interpreters

- Source code written as plain text in a programming language
- But computers need machine code to execute
- Two models: compilers and interpreters
- Compiler:
  - Takes program source code as input and translates into executable (binary machine code)
  - Executable runs as a process

- Interpreter:
  - takes program source code as input, reads line by line and translates internally to computation to perform
  - Simulates execution of that computation



```
#include <iostream>
int main()
{
   ...
}
```



```
with open('input.dat') as f:
    data = f.readline().strip();
     ...
```

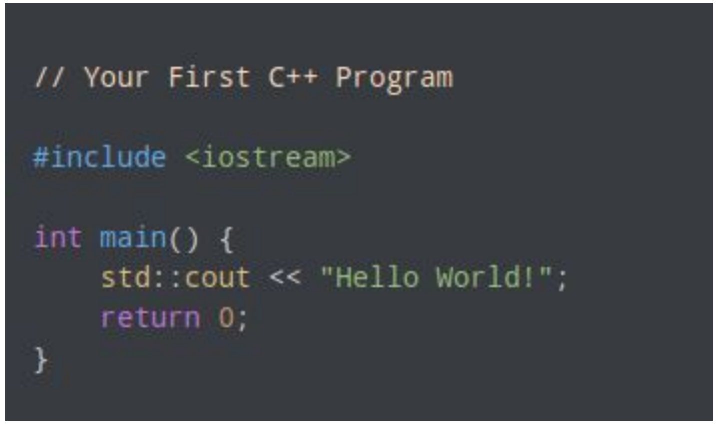
Compiler	Interpreter
Scans the whole program in one go	Translates program line by line
Errors shown in one go at end	Errors shown line by line
Fast execution	Slow execution
Does not require source code for later execution	Requires source code for later execution
C, C++, Rust, Go	Python, Perl, Javascript

Python

- Created by Guido van Rossum, released in 1991
- Usage:
  - Web-development: django, flask, beautifulsoup, selenium
  - Data Science: numpy, pandas, matplotlib, nltk, opencv
  - ML & AI: Tensorflow, Pytorch
- Latest version Python3
  - Use IDE for heavy coding!

- Python code easy to read (closer to English)
- Python uses new lines to complete a command
  - As opposed to semicolons in other languages
- Indentation (via whitespaces) very important
  - Helps define scope (of loops, functions and classes etc)
  - As opposed to curly-brackets in other languages

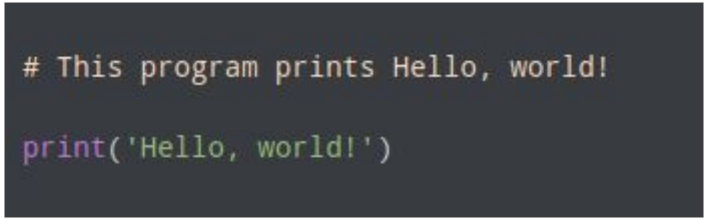
C++ vs Python



```cpp
// Your First C++ Program

#include <iostream>

int main() {
    std::cout << "Hello World!";
    return 0;
}
```



```python
# This program prints Hello, world!

print('Hello, world!')
```

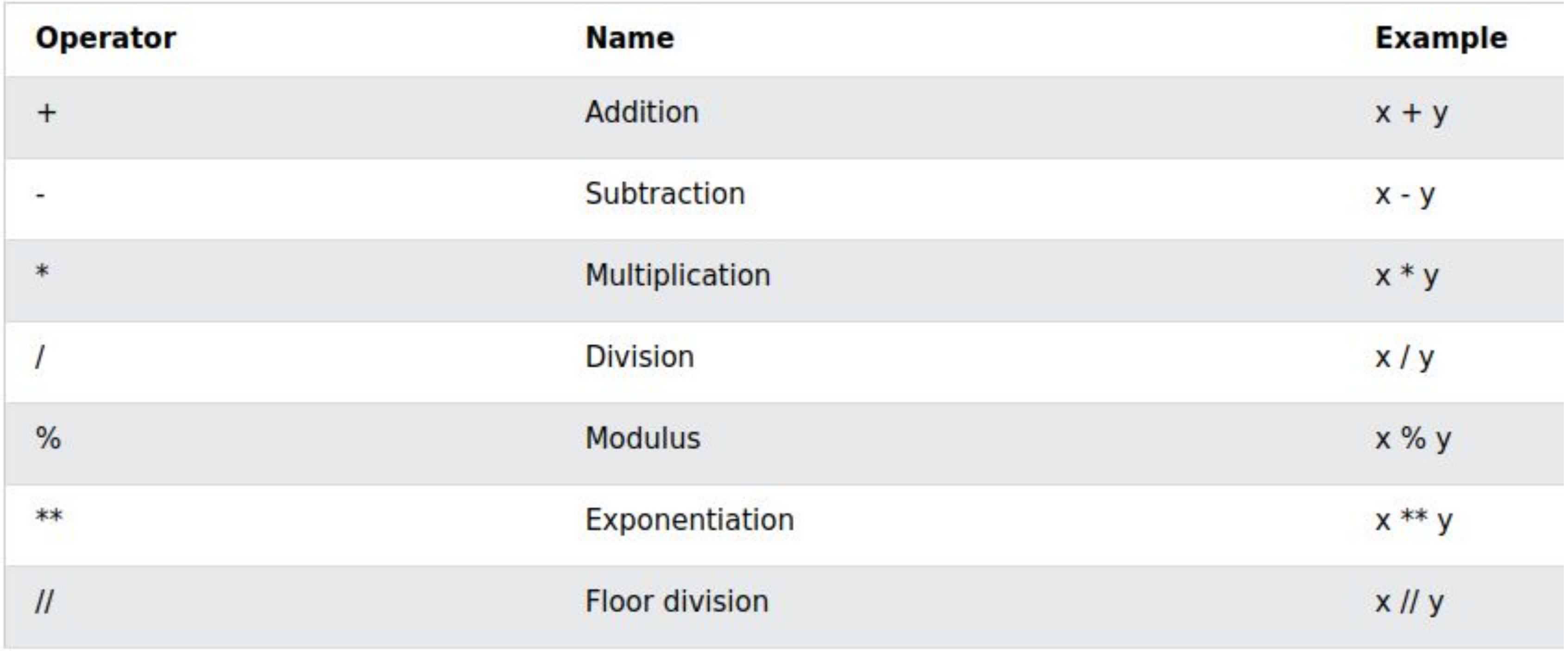
Commenting

- Comments start with a #
- Multiline?
  - Use multiple # (or)
  - Start and end with “””

Variables

- No declaration needed
  - Can also change type later
- Variable names are case sensitive
  - A variable name must start with a letter or the underscore character
  - A variable name cannot start with a number
  - A variable name can only contain alpha-numeric characters and underscores
- Casting: helps specify data type
- Global variables: can be used everywhere, both inside and outside functions
  - Often created outside of a function
  - Can use global keyword inside functions to indicate global scope

Operators



Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
**	Exponentiation	x ** y
//	Floor division	x // y

Operator	Example	Same As
=	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
//=	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
>>=	x >>= 3	x = x >> 3
<<=	x <<= 3	x = x << 3

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

Operator	Description	Example
and	Returns True if both statements are true	x < 5 and x < 10
or	Returns True if one of the statements is true	x < 5 or x < 4
not	Reverse the result, returns False if the result is true	not(x < 5 and x < 10)

Operator	Description	Example
is	Returns True if both variables are the same object	x is y
is not	Returns True if both variables are not the same object	x is not y

Operator	Description	Example
in	Returns True if a sequence with the specified value is present in the object	x in y
not in	Returns True if a sequence with the specified value is not present in the object	x not in y

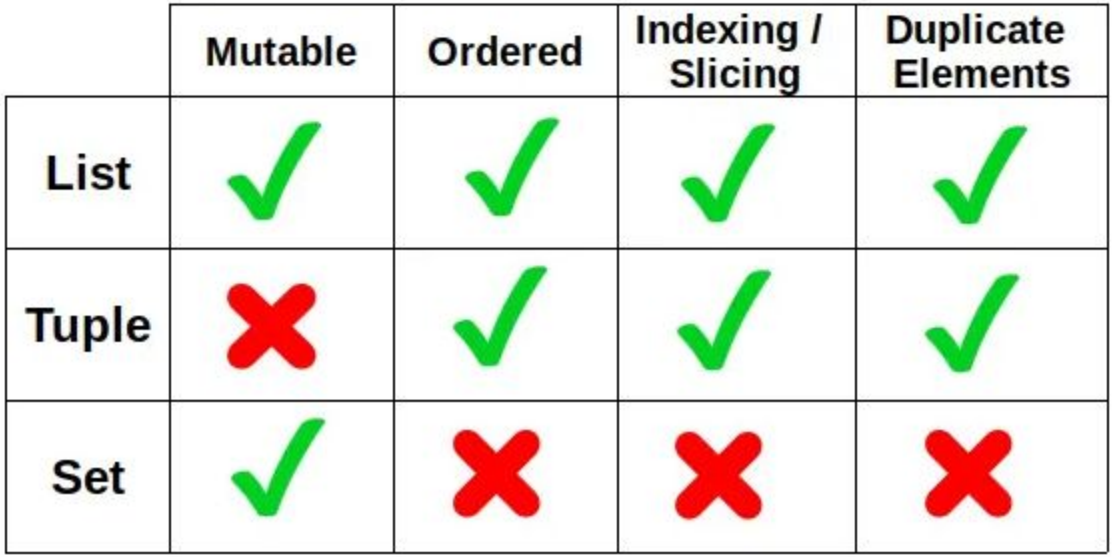
Strings

- Strings can be surrounded by either single quotation marks or double quotation mark
  - We will use double quotation marks mostly
- Can be considered as arrays of characters
- String is an object with its own methods!

Collections

- **List** is a collection which is ordered and changeable. Allows duplicate members.
- **Tuple** is a collection which is ordered and unchangeable. Allows duplicate members.
- **Set** is a collection which is unordered, changeable. No duplicate members.
- **Dictionary** is a collection which is ordered and changeable. No duplicate members.

Collections



	Mutable	Ordered	Indexing / Slicing	Duplicate Elements
List	✓	✓	✓	✓
Tuple	✗	✓	✓	✓
Set	✓	✗	✗	✗

- Lists: If you need to store a collection of items that may change over time (e.g. shopping cart items)
- Tuples: When you have a collection of items that will not change (e.g. pin codes)
- Set: Need to store a collection of unique elements and the order of elements doesn't matter
- Dictionary: If you need to store data as key-value pairs and require fast lookups based on keys

List Comprehension

Syntax: [expression for item in iterable if condition]

- expression: expression to be evaluated for each element in the iterable. Result of this expression will be included in the new list.
- item: The variable representing each element in the iterable
- iterable: The iterable (e.g., list, tuple, range) from which elements are taken.
- if condition (optional): An optional condition that filters the elements
  - Only elements for which the condition evaluates to True will be included in the new list.

Indentation

- Indentation: space at the beginning of a line of code
- Indentation in Python is very important.
  a. Other programming languages use indentation for readability
  b. Python uses indents to denote blocks of code
     - Lines of code that begin a block, end in a colon:
     - Lines within the code block are indented at the same level
     - To end a code block, remove the indentation
- Example: Below will give errors

```
if 5 > 2:

print("Five is greater than two!")

if 5 > 2:

    print("Five is greater than two!")

        print("Five is greater than two!")
```

Conditionals and Loops

- Usual logical conditions
  - Equals: a == b
  - Not Equals: a != b
  - Less than: a < b
  - Less than or equal to: a <= b
  - Greater than: a > b
  - Greater than or equal to: a >= b
- Keywords: if, elif, else

- Python supports:
  - while loops
  - for loops
    - Range function is useful here

Functions

- Function is defined using the “def” keyword
- To call a function, use the function name followed by parenthesis
- By default, a function must be called with the correct number of arguments, else you will get error
- We can pass a variable number of arguments to a function using special symbols
  - *args (Non Keyword Arguments, tuple or list or sets)
  - **kwargs (Keyword Arguments, dictionary)
- Lambda function: a special type of function without the function name (anonymous functions)

Class/Objects

- Almost everything in Python is an object, with its properties and method
- A Class an object constructor, or a "blueprint" for creating objects
- Built-in functions
  - __init__ function
  - __str__ function

Modules

- Module is like a code library: additional pieces of code that further extend Python's functionality
  - Can contain functions, variables etc
- Modules are accessed using import
- Many in-built modules
  - Checkout platform, re, sys, math

Point to Note

- When you import modules, you will see a __pycache__ directory
- Automatically created by Python to store compiled bytecode files (.pyc) generated by the Python interpreter
- Interpreter checks whether there's a corresponding .pyc file in the __pycache__ directory
  - If the .pyc file exists and is up to date with the source code, it will be used to speed up subsequent imports, as Python doesn't need to recompile the source code!

File Handling

- Files are manipulated by creating a file object
  - The key function for working with files in Python is the open() function
  - open() function takes two parameters; filename, and mode
    - Mode can be r, a, w … (read, append, write …)
- File object then can be used with methods to read, write, close etc

References

- https://www.w3schools.com/python
- https://www.w3schools.com/python/python_examples.asp