Introduction to programming using Python

Session 1

Matthieu Choplin

http://pythonguy.com/

Recommended reading

- Books:
  - Automate the boring stuff by Al Sweigart: https://automatetheboringstuff.com/
  - Introduction to programming using Python by Daniel Liang
- Website:
  - The official python tutorial: https://docs.python.org/3/tutorial/

Tools

- IDE: Idle first and then Pycharm
- Online tool: pythontutor.com

Objectives

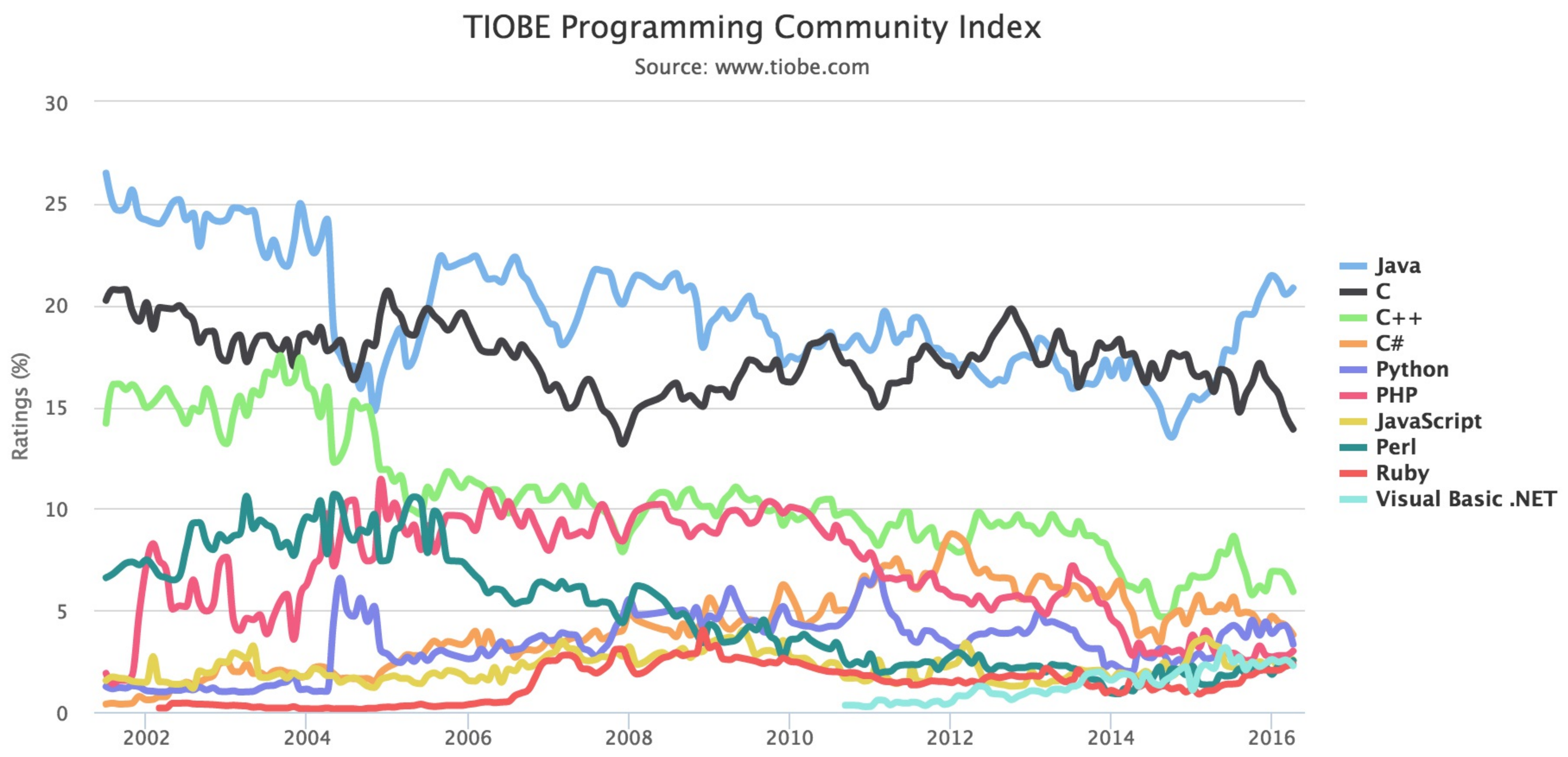
- Think like a programmer.
- Introduction to Python. Variables. Loops. Main method. Conditional structures. Data structure.
- Functions and methods
- Debugging in Python. How to read a program.
- File manipulation: Reading and writing files.
- Object Oriented programming in Python: classes, objects, inheritance, polymorphism, encapsulation.
- Introduction to the Python standard library.
- Testing in Python. Presentation of doctest and unittest.
- Error handling: exceptions.

Objectives for Session 1

- To get a brief overview of what Python is
- To understand computer basics and programs
- Installing python and introducing IDLE
- To understand what statement, variable and expressions are
- To explain the differences between syntax errors, runtime errors, and logic errors.
- Write your first python program

Python's popularity

Python has been amongst the top 10 programming languages for more than a decade according to the Tiobe index

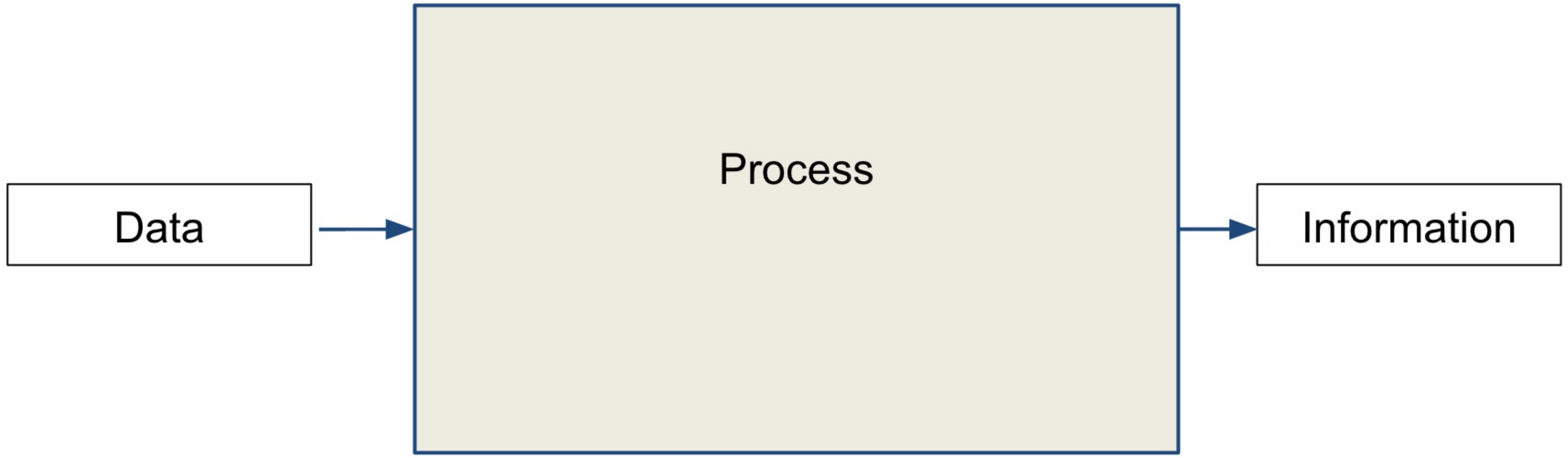


Who uses Python?

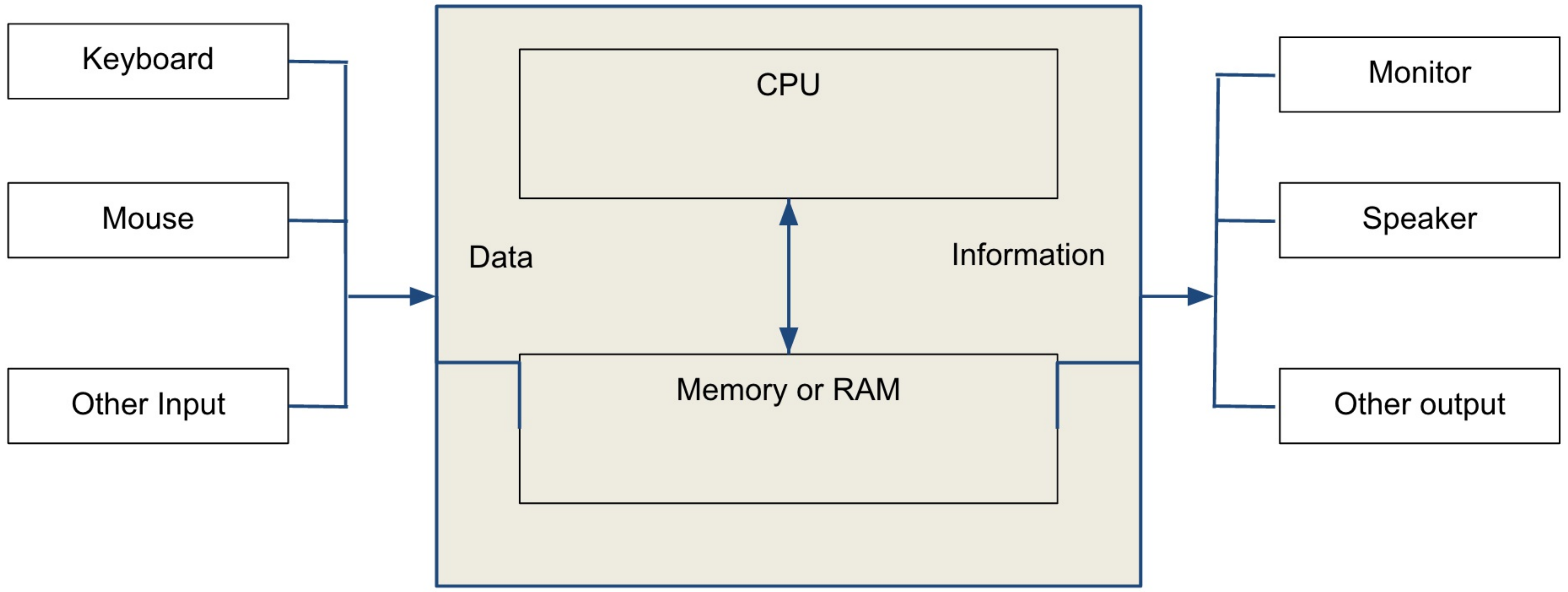
cars.com
Confidence Comes Standard.
Massachusetts Institute of Technology
FreeBSD
NetBSD
NASA
Walt Disney Animation Studios
NOKIA Connecting People
YAHOO!
redhat
instagram
evite.com
Atlassian bitbucket
YouTube
CANONICAL
LexisNexis
Consumer Notebook
National Weather Service
Los Angeles Times
EVE ONLINE
ebay
ableton
mozilla Firefox
Google
bit.ly
AstraZeneca
reddit
Quora
TRUECar Know the Real Price
Dropbox
Industrial Light & Magic
A Lucasfilm Company

Before going more into details... What is a computer? (1/2)

An electronic device that is receiving data input, storing (in RAM) and processing (in the CPU) them and producing information in output.



What is a computer? (2/2)



What is a program?

- Computer programs, known as software, are *instructions to* the computer.
- You tell a computer what to do through programs. Without programs, a computer is an empty machine. Computers do not understand human languages, so you need to use computer languages to communicate with them.
- Programs are written using programming languages.

Different types of Programming Languages

- **Machine language** is a set of primitive instructions built into every computer. The instructions are in the form of binary code. The programs in machine language are highly difficult to read and modify. For example, to add two numbers, you might write an instruction in binary like this:

  1101101010011010

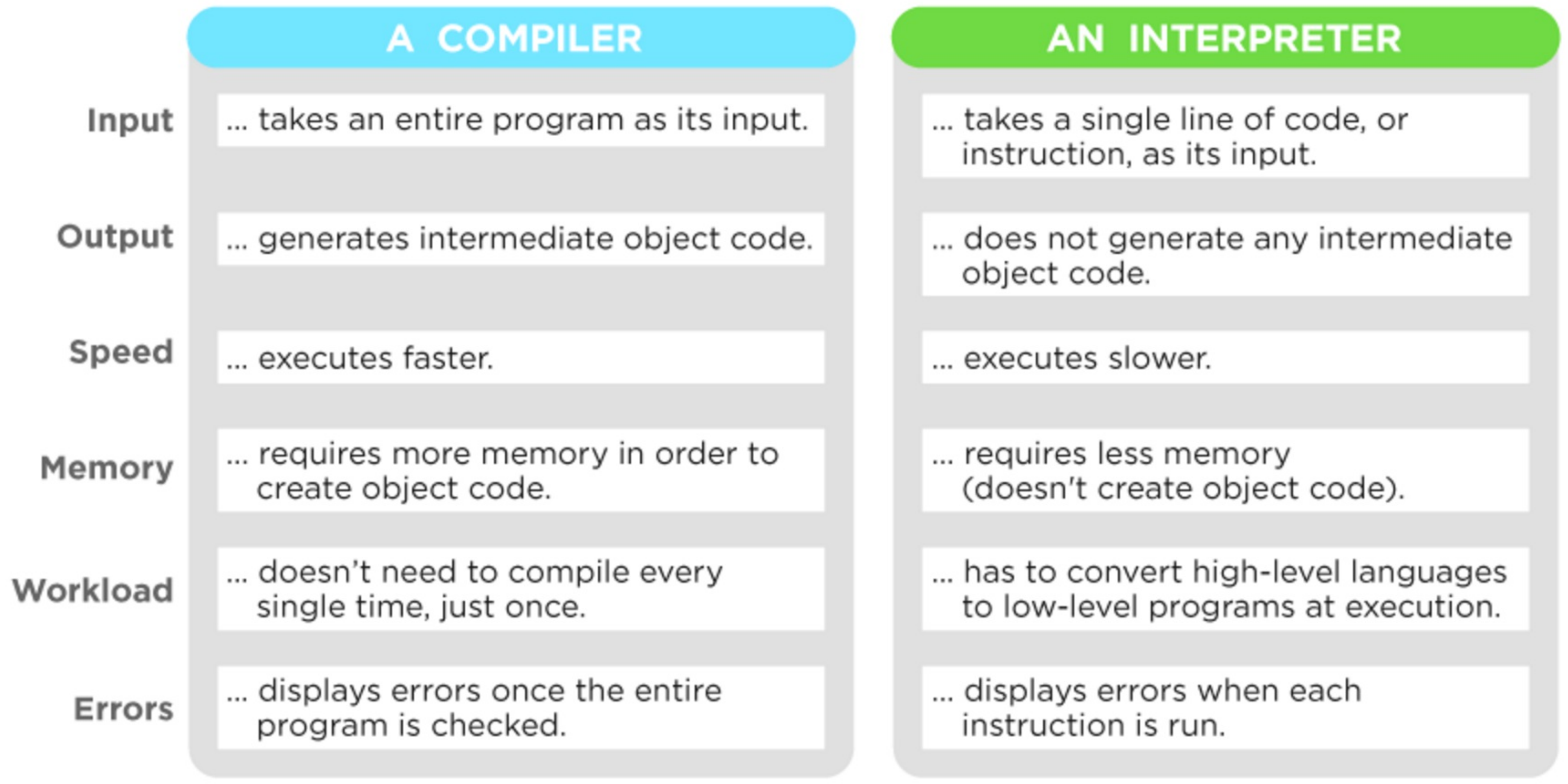
- **The high-level languages** are English-like and easy to learn and program. For example, the following is a high-level language statement that multiply two number:

```
area = 5 * 5
```

Computer works using binary logic.

- It is extremely difficult for humans to program in binary.
- Computer languages have to be translated to binary logic for the computer to understand.
- Two types of translation:
  - Compilation
  - Interpretation

Comparison of interpreted vs compiled



	A COMPILER	AN INTERPRETER
Input	... takes an entire program as its input.	... takes a single line of code, or instruction, as its input.
Output	... generates intermediate object code.	... does not generate any intermediate object code.
Speed	... executes faster.	... executes slower.
Memory	... requires more memory in order to create object code.	... requires less memory (doesn't create object code).
Workload	... doesn't need to compile every single time, just once.	... has to convert high-level languages to low-level programs at execution.
Errors	... displays errors once the entire program is checked.	... displays errors when each instruction is run.

Installing python (Windows)

- Be sure to download a version of Python 3.
- You'll find Python installers for 64-bit and 32-bit computers for each operating system on the download page, so first figure out which installer you need.
- Here's how to find out for sure:
  - Select Start > Control Panel > System and check whether System Type says 64-bit or 32-bit.
  - Download the Python installer (the filename will end with .msi) and double-click it. Follow the instructions the installer displays on the screen to install Python
  - Select Install for All Users and then click Next.
  - Install to the C:\Python34 folder by clicking Next.
  - Click Next again to skip the Customize Python section.

Installing python (OS X)

- Be sure to download a version of Python 3.
- You'll find Python installers for 64-bit and 32-bit computers for each operating system on the download page, so first figure out which installer you need.
- Here's how to find out for sure:
  - Go the Apple menu, select About This Mac > More Info > System Report > Hardware, and then look at the Processor Name field. If it says Intel Core Solo or Intel Core Duo, you have a 32-bit machine. If it says anything else (including Intel Core 2 Duo), you have a 64-bit machine.
- Download the .dmg file that’s right for your version of OS X and double-click it. Follow the instructions the installer displays on the screen to install Python.
  - When the DMG package opens in a new window, double-click the Python.mpkg file. You may have to enter the administrator password.
  - Click Continue through the Welcome section and click Agree to accept the license.
  - Select HD Macintosh (or whatever name your hard drive has) and click Install.

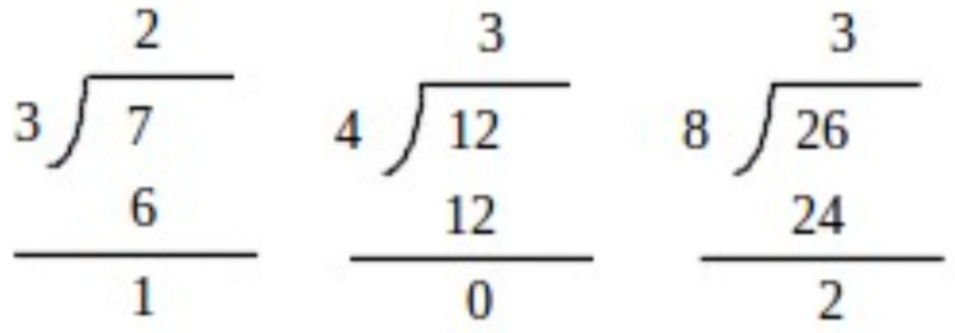
Starting IDLE

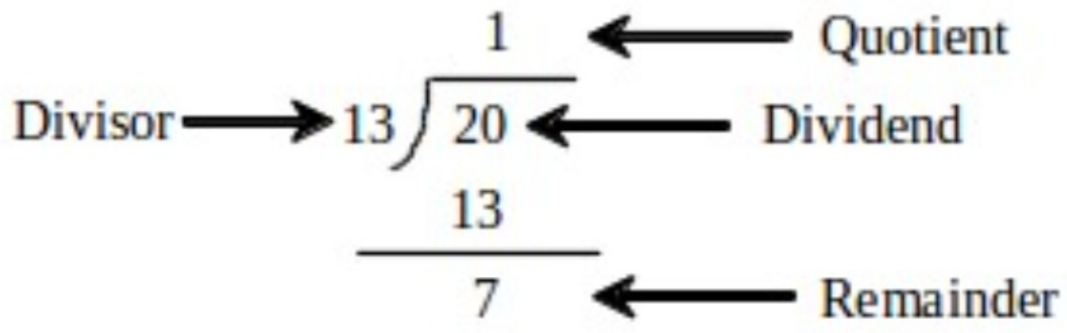
- While the Python interpreter is the software that runs your Python programs, the interactive development environment (IDLE) software is where you'll enter your programs, much like a word processor. Let's start IDLE now.
- On Windows 7 or newer, click the Start icon in the lower-left corner of your screen, enter IDLE in the search box, and select IDLE (Python GUI).
- On Mac OS X, open the Finder window, click Applications, click Python 3.4, and then click the IDLE icon.

Experimenting with the interpreter and the Numeric Operators

Name	**Meaning**	**Example**	**Result**
+	Addition	34 + 1	35
-	Substraction	34.0 - 0.1	33.9
*	Multiplication	300 * 30	9000
/	Float division	1 / 2	0.5
//	**Integer Division**	1 // 2	0
**	**Exponentiation**	4 ** 0.5	2.0
%	**Remainder**	20 % 3	2

The % (modulo or remainder) operator (1/2)





The % (modulo or remainder) operator (2/2)

Remainder or Modulo is very useful in programming. For example, an even number % 2 is always 0 and an odd number % 2 is always 1. So you can use this property to determine whether a number is even or odd.

Arithmetic expressions

$$\frac{3+4x}{5} - \frac{10(y-5)(a+b+c)}{x} + 9(\frac{4}{x} + \frac{9+x}{y})$$

...is translated to:

```
(3 + 4 * x) / 5 – 10 * (y - 5) * (a + b + c) / x +\
    9 * (4 / x + (9 + x) / y)
```

NB: the sign \ is an "escaped" character, to break a line for readability

Python Syntax

- Variable
- Expression
- Indentation
- Comments

Variable (1)

It is a space created in memory (in RAM) where we can temporarily store values or data. We use the sign **'='** for assigning a value to a variable.

You can think of it like a box. For example, we store the value (or expression) 1 in the box (i.e the variable) **a**.

Notice that we do not specify the type (integer, string) of the variable, python sees it automatically.

```
a = 1
```

Variable (2)

```
a = 1
```

The variable has a name so that we can reuse it. When we use a variable, it is for retrieving the value that it is holding.

```
>>> a = 1
>>> a
1
```

Variable (3)

- A variable name is a non-empty sequence of characters of any length with:
  - The start character can be the underscore "_" or a capital or lower case letter.
  - Python keywords are not allowed as identifier names!

Variable (4)

Reserved keywords that you should not used:

and	as	assert	break	class	continue	def
del	elif	else	except	exec	finally	for
from	global	if	import	in	is	lambda
not	or	pass	print	raise	return	try
while	with	yield				

Python Syntax - Expression

It represents something, like a number or a string.
Expressions are nothing but values, except they can have operations like addition or subtraction.

```
>>> 1 # is an expression
>>> 2 + 3 # is also an expression
>>> "hello" # as well
```

We can put an expression (i.e. a value) in a variable

Using the function print

The **print** function displays what is given in parameter.

```
>>> print("Hello World")
```

- A function has a name
- A function has parameters (that can be optional)
- A function is executed!

Exercise 1: From algorithm to Python code

- Translate the following algorithm into Python code:
  - Step 1: Use a variable named *miles* with initial value 100 .
  - Step 2: Multiply *miles* by 1.609 and assign it to a variable named kilometers
  - Step 3: Display the value of kilometers with the function print()

👁 Show solution

Exercise 2.1: Area of a squared room

- The *length* and *width* are hardcoded variables for now.
- Use variables (for length, width and area)
- The multiply operator in Python is the sign*
- Formulae of the area of a square: length * width
- Use the **print()** function to display the result

👁 Show solution

Using the function input()

The input() function waits for the user to type some text on the keyboard and press ENTER.

```
>>> a =  input("Enter a value")
>>> print(a)
```

Using the function type()

The **function type** gives the type of a value

```
>>> type(1)
<class 'int'>
```

Using the type() function, can you tell what the type of 'abc' is?

Using the function float()

The **function float** is converting a value into a numeric value if possible

```
>>> float("234")
234
```

Exercise 2.2: Dynamic Area

- The *length* and *width* are dynamic variables now.
- Use the **input()** function for taking the values from the user.
- Convert the input received into a number with the function **float()**

👁 Show solution

Python Syntax - Indentation

The indentation is the increase or decrease of space between the left margin and the first character of the line.

The code need to be properly indented, else python will raise an error.

For example, what is wrong here?

```
  print("what is wrong?")
print("indented properly ")
```

Python Syntax - Comments

If you want to comment a line, you can use the **#** (pound sign) that you place before the commented line

You can also comment multiple lines using ''' (triple quote) before and after the commented paragraph

Example

```
print("This line is not commented")
# print("This line will not do anythin")
'''
print("Nothing to see here")
print("""This whole block is commented.
And it will not show anything when we run the program""")
'''
```

The different type of errors (1/3)

Syntax Error

For example, when we forget a quote to close a string

```
print("Welcome to Python)
```

The different type of errors (2/3)

Runtime Error

For example a division by zero (which is impossible)

```
print(1/0)
```

The different type of errors (3/3)

Logic error

When a program is not doing what we want it to do.

For instance, a wrong formulae for converting pound in kg

```
pounds = float(input("Enter weight in pound: "))
# convert pound in kilogramme
kilograms = pounds / 0.454
```

It should be:

```
pounds = float(input("Enter weight in pound: "))
# convert pound in kilogramme
kilograms = pounds * 0.454
```

Exercise:

Write a program that converts pounds into euros.

- The values can be hard coded for now (it means that the program will not be dynamic)
- Use comments
- Use variables
- Use print

Exercise:

- Write a program that ask the user what amount is to be converted in euros, convert it and display the result.
- Hint: we are going to need the function **input** and the function **float**