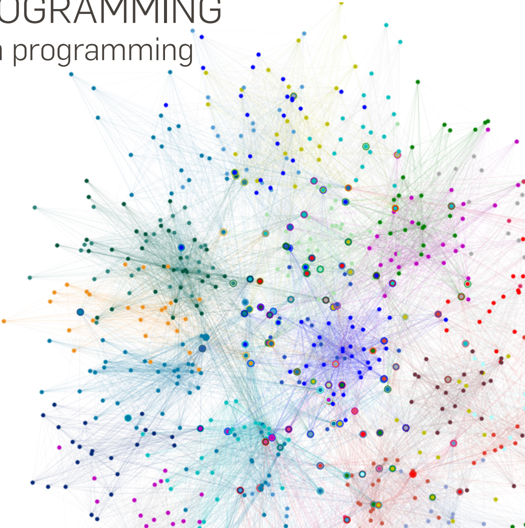
ThangLong University

CS100: INTRO TO PROGRAMMING

Getting started with python programming

October 10, 2017



Lecturer: Thang Duc Nguyen

Email: thangdn.tlu@gmail.com
Phone: (+84)968-486-632

Mathematics and Informatics
ThangLong University

Overview

.

In this lecture, our plan is to lead you into the world of Python programming by taking you through the basic steps required to get a simple program running.

Overview

WHY LEARN TO CODE?

Why learn to code?

Programming is a powerful tool you can use to solve all kinds of problems

What do you want to do?

→ Build a phone app to help you find directions

→ Calculate how much money you need to buy a car

→ See what people are saying about your business on social media

→ Program a wearable device so it tweets you when you should re-apply sunscreen

"Don't just play on your phone, program it."
— President Barack Obama
Anybody can learn!
Start with one #HourOfCode
HOUR OF CODE

Why Python?

Beginner Friendliness and Easy to Understand

python
powered

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

Very Flexible

Source: bestprogramminglanguagefor.me

Career Opportunities

Salary Range

43K - 135K

Average Salary

$94,053

Popular sites built with Python

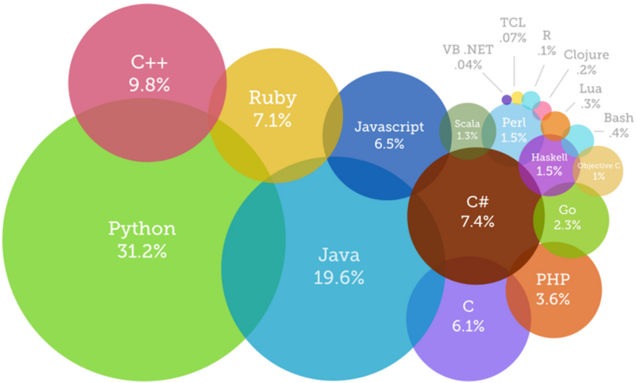
Google Dropbox Pinterest Instagram

Source: bestprogramminglanguagefor.me

And as a bonus

Once you learn how to code in one programming language it will be easier to learn another programming language, and another, and another...

Most Popular Coding Languages of 2015



PROGRAMMING IN PYTHON

Booksite: Introduction to programming in python

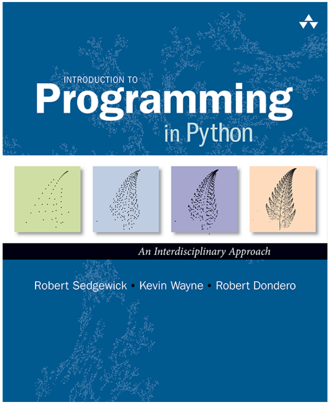
Introduction to Programming in Python

Site: introcs.cs.princeton.edu/python

CS100: Chapter 1 - Elements of Programming

The booksite consists of the following elements:

→ Excerpts
→ Exercises
→ Python code


INTRODUCTION TO
Programming
in Python
An Interdisciplinary Approach
Robert Sedgewick • Kevin Wayne • Robert Dondero

To program in Python, you need to:

1. Compose a program by typing it into a file named, say, myprogram.py

Using: gedit myprogram.py &

```
import stdio
#Write 'Hello, World' to standard output.
stdio.writeln('Hello, World')
```

2. Run (or execute) it by typing python myprogram.py in the terminal window.

Using: python3 myprogram.py

COMMAND-LINE ARGUMENT

What is command-line argument?

```
cp input.txt ~/Desktop/cti034
```

Progranm name: cp

Argument 1: input.txt

Argument 2: ~/Desktop/cti034

Typically, we want to provide input to our programs, that is, data that they can process to produce a result. The simplest way to provide input data is illustrated in the above example.

Using a command-line argument

```
$ gedit useargument.py &
```

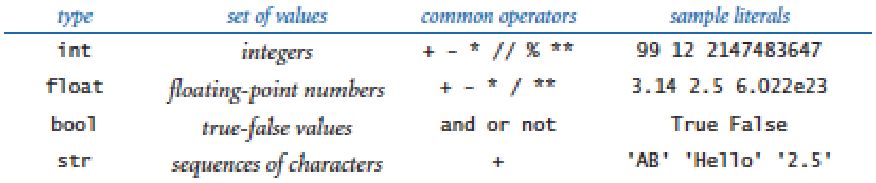
```
import stdio
import sys

stdio.write('Hi, ')
stdio.write(sys.argv[1])
stdio.writeln('. How are you?')
```

```
$ python3 useargument.py Alice
Hi, Alice. How are you?
$ python3 useargument.py Bob
Hi, Bob. How are you?
```

BUILT-IN TYPES OF DATA

Built-in types of data

A data type is a set of values and a set of operations defined on those values.



type	*set of values*	*common operators*	*sample literals*
`int`	*integers*	`+ - * // % **`	`99 12 2147483647`
`float`	*floating-point numbers*	`+ - * / **`	`3.14 2.5 6.022e23`
`bool`	*true-false values*	`and or not`	`True False`
`str`	*sequences of characters*	`+`	`'AB' 'Hello' '2.5'`

Basic built-in data types

a+b program?

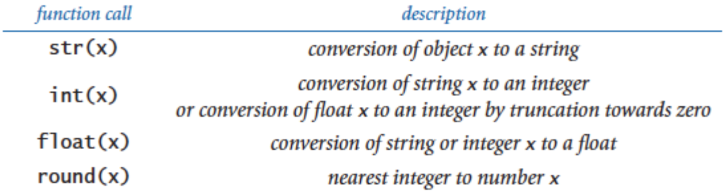
```
$ gedit add.py &
```

```
import stdio
import sys

stdio.write(sys.argv[1]+'+'+sys.argv[2]+'=')
stdio.writeln(sys.argv[1] + sys.argv[2])
```

```
$ python3 add.py 5 6
$ 5 + 6 = 56
```

Type Conversion

Explicit type conversion. Call functions such as int(), float(), str(), and round().



function call	*description*
`str(x)`	*conversion of object* `x` *to a string*
`int(x)`	*conversion of string* `x` *to an integer* *or conversion of float* `x` *to an integer by truncation towards zero*
`float(x)`	*conversion of string or integer* `x` *to a float*
`round(x)`	*nearest integer to number* `x`

APIs for some built-in type conversion functions

[Sovled] a+b program!

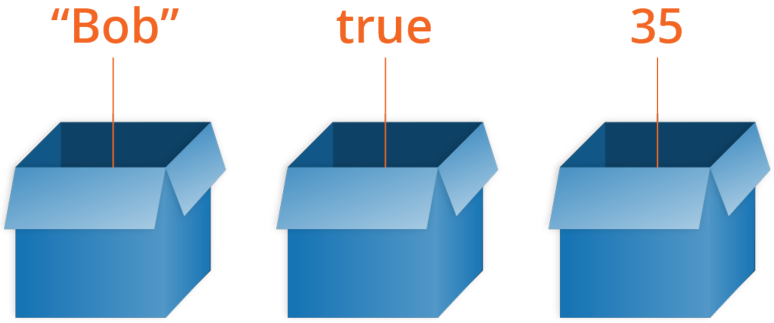
```
$ gedit add.py &
```

```python
import stdio
import sys

stdio.write(sys.argv[1]+'+'+sys.argv[2]+'=')
stdio.writeln(float(sys.argv[1]) + float(sys.
    argv[2]))
```

```
$ python3 add.py 5 6
$ 5 + 6 = 11.0
```

What is variable?

Think of a variable as a box where you can store something and come back to get it later.



If you need to remember more than one value, just create more variables

Variable names

* Rules
  - → Can not contain spaces
  - → Are case sensitive
    firstName and firstname would be two different variables
  - → Cannot start with a number
* Guidelines
  - → Should be descriptive but not too long
    (favoriteSign not yourFavoriteSignInTheHoroscope)
  - → Use a casing "scheme"
    camelCasing or PascalCasing

#Task01

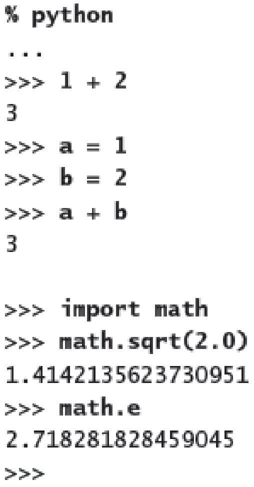
Which of the following do you think would be good names for variables?

- → Variable1
- → First Name
- → Date
- → 3Name
- → DOB
- → DateOfBirth
- → YourFavoriteSignInTheHoroscope

math library

Python’s math library

- → $|x|$: abs(x)
- → min(a,b)
- → max(a,b)
- → $\sqrt{x}$: math.sqrt(x)
- → e^x : math.exp(x)
- → $\sin(x)$: math.sin(x)
- → $\cos(x)$: math.cos(x)
- → $\log(x)$: math.log(x)
- → etc.



```
% python
...
>>> 1 + 2
3
>>> a = 1
>>> b = 2
>>> a + b
3

>>> import math
>>> math.sqrt(2.0)
1.4142135623730951
>>> math.e
2.718281828459045
>>>
```

Your challenge

Build a quadratic formula calculator quickly!
The general quadratic equation is:

$$ax^2 + bx + c = 0$$

The solution(s) to a quadratic equation can be calculated using the Quadratic Formula:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

CONCLUSION