

# Python 101

## Introduction

---

PYTHON 101 TEAM

### Course overview

---

- **Day 1:** Installing and configuring Python; Introduction to Python programming
  - **Day 2:** Data structures
    - Strings
    - Lists
    - Tuples
    - Dictionaries
    - Sets
  - **Day 3:** Control flow
    - Conditional statements (if, elif)
    - Loops (while, for)
  - **Day 4:** Functions and Input/Output
  - **Day 5:** Modules (*sys*, *os*, *re*, *Bio*) and Do It Yourself day
- 

### What is python?

---

Python is an **interpreted, object-oriented, high-level** programming language, that emphasizes in code readability:

- **Interpreted:** Programs can be run as soon as they are written, no need to compile;
  - **Object-oriented:** The programming paradigm, where "objects" can be somewhat separated from the rest of the program. An object is an entity that has a:
    - **Identity:** Unique identifier of the object
    - **Value:**
    - **Type:** whether it is a string, number, etc.
  - **High-level:** Abstracts from the computer it is being run on;
-

# Two ways of running python code

---

## The interactive mode

- Ideal for **training** and **testing blocks of code** on the fly. During an interactive session, instructions are directly executed. However, the code cannot be saved between different sessions. Some interactive interpreters include:
  - Python's built-in interpreter (<http://docs.python.org/tutorial/interpreter.html>)
  - Python IDLE (<http://docs.python.org/library/idle.html>)
  - Dreampie (<http://dreampie.sourceforge.net/>)

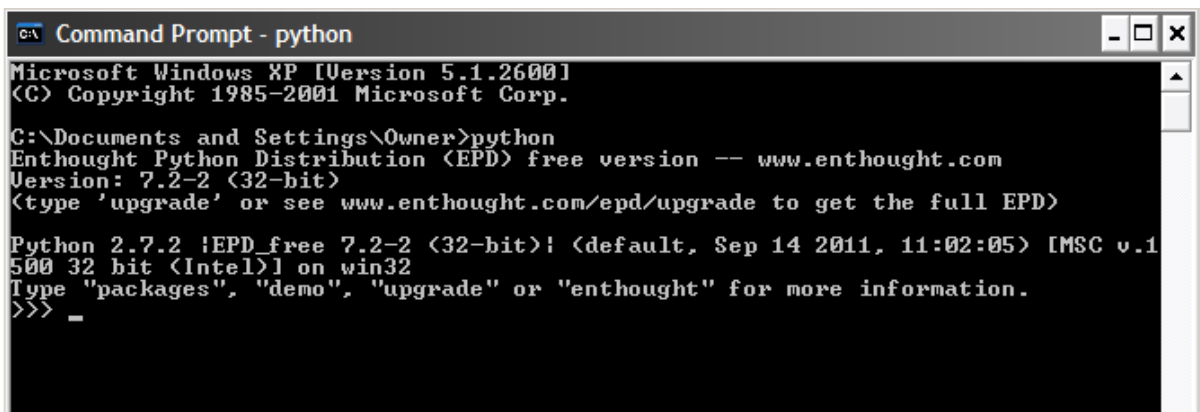
## Script mode

- Used to construct Python programs following this simple procedure:
  - Create and edit a text file with Python code using a text editor (geany (<http://www.geany.org/>), vim (<http://www.vim.org/>))
  - Save the file with the filename extension ".py"
  - Run the script with the Python interpreter

## The interactive mode

---

In its easiest form, the interactive mode can be initialized by simply typing **"python"** in the command line prompt.



```
C:\> Command Prompt - python
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Owner>python
Enthought Python Distribution (EPD) free version -- www.enthought.com
Version: 7.2-2 (32-bit)
(type 'upgrade' or see www.enthought.com/epd/upgrade to get the full EPD)

Python 2.7.2 !EPD_free 7.2-2 (32-bit)! (default, Sep 14 2011, 11:02:05) [MSC v.1
500 32 bit (Intel)] on win32
Type "packages", "demo", "upgrade" or "enthought" for more information.
>>> _
```

- The symbols ">>>" are the prompt of the interactive mode. To execute an instruction, write on the prompt line and press enter.
- The symbols "...\_" are the secondary prompt. They represent continuation lines, i.e., when the instruction is not yet complete and ready to be executed.

- To exit the interactive mode, you can use the keywords "**Ctrl + D**" or type "**quit()**"
- 

## The script mode

---

Writing scripts will make the majority of your coding experience. A python script that contains a set of instructions can be executed in more than one way:

- **Script argument way:**
    - Run your python interpreter in the prompt with the script as an argument (e.g. `python my_script.py`). This mode is preferred for scripts that require arguments to be passed when executed
  - **Double clicking:**
    - Run your script simply by double-clicking it. However, this mode will only be useful when the user input is collected using the prompt feature of the **`input()`** and **`raw_input`** functions, that will be covered in the 4th Day
- 

## Introducing Python programming

---

### Data structures (Day 2)

- Basic operators ("`+`", "`-`", "`/`", "`*`")
- Variable assignment
- Standard data structures (Strings, numbers, lists and dictionary)

### Control flow (Day 3)

- Control flow is used in Python whenever you want to:
    - **Make choices**, using *conditional statements* when conditions are met
    - **Perform repeated actions**, using a set of statements executed *n* times in a *loop* until a condition is met
- 

## Introducing Python programming

---

## Functions (Day 4)

- Blocks of code that perform specific tasks (procedures) repeatedly
- How to create new functions
- Using arguments to bring flexibility to those procedures

## Input/Output (Day 4)

- The three input/output channels: standard input, standard output and standard error
  - Handling input files (opening and parsing) and output files (writing)
  - Interactive scripts that request user input
  - Printing messages to the terminal
- 

# Introducing Python programming

---

## Modules (Day 5)

- What are modules, how do they work and how to use them
  - Overview of some of the most useful modules:
    - *re*, regular expressions
    - *sys*, system tools
    - *os*, operative system tools and commands
    - *biopython*, the bioinformatic's module of choice
- 

## Online resources

---

Some of the most usefull resources that will help you solving problems and getting new ideas during your coding sessions:

- Official Python tutorial (<http://docs.python.org/tutorial/>): From the official Python documentation, this resource covers all basic and advance built-in features of Python.
- Python wiki (<http://wiki.python.org/moin/>): Includes several official Python resources, such as the beginners guide (<http://wiki.python.org/moin/BeginnersGuide>), beginners errors (<http://wiki.python.org/moin/BeginnerErrorsWithPythonProgramming>) and Python books (<http://wiki.python.org/moin/PythonBooks>).

- Stack Overflow (<http://stackoverflow.com/>): An extremely useful language-independent collaboratively edited question and answer site for programmers. Most of the questions and problems that you may face when coding have been probably already answered in this forum.

## **Google is your friend**

- Giving the current flood of digital data, the importance of google's search engine to sort over all information cannot be overstated. The trick, however, is knowing how to put your problem in order to obtain the most useful answers.
-