THE
WEEKEND
STARTS!
HERE.

CMPU4060-E
Object Oriented Software Development 1

Dr Andrew Hines

Room KE-G 026A
School of Computing
Dublin Institute of Technology

Today

- Course Content - description, syllabus, learning outcomes
- Assessment and Credits - how much work is there to do?
- Why do it at all? - about me and you
- Structure of the module - Lectures and Labs
- Assignments and Study - Self directed learning

Module Description

This module provides the learner with the fundamental skills of programming and object oriented programming

Module Aims

- To provide the learner with strong fundamental programming
- To provide the learner with object-oriented programming skills
- To ensure the learner has the necessary skills to design and develop an application using an object-oriented language

Learning Outcomes

On completion of this module, the student should be able to:

1. Design an object-oriented software application
2. Implement a software application using an object-oriented programming language utilising core object-oriented programming concepts, and develop problem solving skills as part of this process
3. Test and debug an object-oriented software application
4. Implement basic algorithms and data structures using an object-oriented programming language
5. Select and evaluate appropriate methods, including algorithms and patterns, for the implementation of object-oriented solutions.

Module Content

How we will deliver the Learning Outcomes?

- Fundamentals of Programming (40%)
  - Types, variables and operators
  - Control structures
  - Code style and quality
- Object Oriented Programming (40%)
  - Objects and classes
  - Methods
  - Inheritance and polymorphism
- Exception handing Data Structures and Algorithms (20%)
  - Collections
  - Basic data structures and algorithms e.g. 1D and 2D arrays, searching and sorting
  - Analysis of algorithms

Assessment

- Continuous Assessment is carried out throughout the semester
- Plagiarism will be monitored closely in all assignments
- All assessments will be submitted via the webcourses site:
  ▸ http://www.dit.ie/lttc/webcourseslogin/
- Additional course content will be posted on webcourses - check it regularly!

Assessment

- 1 semester= 30 ECTS credits
- 5 credits for each other core computing module
- 10 credits for this module
    - Breakdown the time:
    - At 25-30 hours effort per credit = 250 - 300 hours course work
    - 24 hours of lectures
    - 48 hours of labs = 178 hours of your own work!
    - But remember: No exam. No revision. No essays.

About Me

- Undergraduate study in TCD Engineering
- Spent a decade in the software industry
  - Software product companies in the airline and financial sectors
  - Various roles from developer to software architect to Director of Software Development
- MSc in Technology Management (NITM, UCD)
- MBA (Smurfit Business School)
- PhD in Electronic Engineering (Speech and Hearing Signal Processing, EE Dept, TCD)

About You

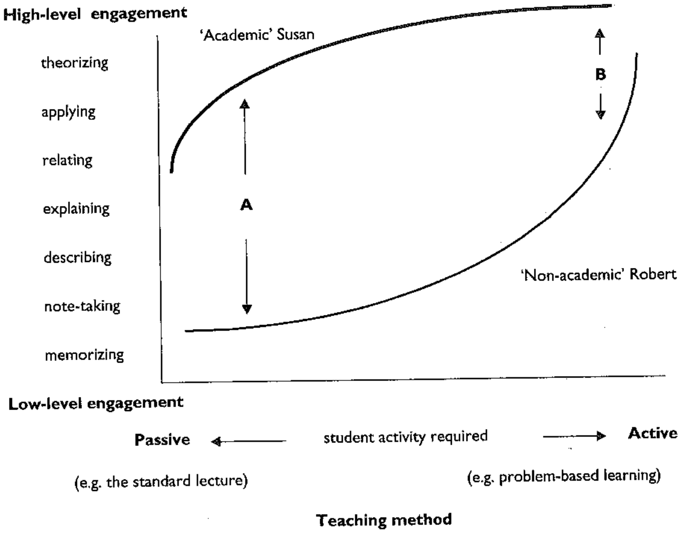
About You

Zero to Hero

Apprentices

Learn through doing.
Practice, Practice, Practice

How Do You Learn



Why is This Hard?

- I cannot precisely explain why it is hard, only that it is indeed hard.
- Typical quote: “Never have I worked so hard and gotten so low a grade.”

An Analogy

Let us say that you have signed up to study French poetry (how about Marot) in the original language.

You have two problems:

- you don't speak French
- you don't know much about poetry

Clement Marot 1496-1544

- *Ma mignonne*
- *Je vous donne*
- *Le bon jour;*
- *Le séjuour*
- *C'est prison.*
- *Guérison*
- *Recouvrez,*
- *Puis ouvrez*
- *Votre porte*
- *Et qu'on sorte*
- *Vitement,*
- *Car Clément*
- *Le vous mande.*
- *Va, friane*
- *De ta bouche,*
- *Qui se couche*
- *En danger*
- *Pour mange*
- *Confitures;*
- *Si tu dures*
- *Trop malade,*
- *Couleur fade*
- *Tu Prendras,*
- *Et perdras*
- *L'embonpoint.*
- *Dieu te doint*
- *Santé bonne,*
- *Ma mignonne.*

Crappy-Literal Translation

- *My sweet/cute [one] (feminine)*
- *I [to] you (respectful) give/bid/convey*
- *The good day (i.e., a hello, i.e., greetings).*
- *The stay/sojourn/visit (i.e., quarantine)*
- *[It] is prison.*
- *Cure/recovery/healing (i.e., [good] health)*
- *Recover (respectful imperative),*
- *[And] then open (respectful imperative)*
- *Your (respectful) door,*
- *And [that one (i.e., you (respectful)) should} go out*
- *Fast[ly]/quick[ly]/rapid[ly],*
- *For/because Clement*
- *It (i.e., thusly) [to] you(respectful) commands/orders.*
- *Go (familiar imperative), fond-one/enjoyer/partaker*
- *Of your (familiar) mouth,*
- *Who/which herself/himself/itself beds (i.e., lies down)*
- *In danger;*
- *For/in-order-to eat*
- *Jams/jellies/confectionery.*
- *If you (familiar) last (i.e., stay/remain)*
- *Too sick/ill,*
- *[A] color pale/faded/dull*
- *You)familiar) will take [on],*
- *And [you (familiar)] will waste/lose*
- *The plumpness/stoutness/portliness (i.e., well-fed look).*
- *[may] God [to] you (familiar) give/grant*
- *Health good,*
- *My sweet/cute [one] (feminine).*

Decent Trans, S.Jamar

- *My sweet dish,*
- *You I wish*
- *A good day.*
- *Where you stay,*
- *Is a jail.*
- *Though so pale,*
- *Leave your bed,*
- *Regain red.*
- *Ope' your door*
- *Stay not, poor*
- *Child; gain strength*
- *And at length,*
- *Steve does urge,*
- *Please emerge.*
- *Then go eat*
- *Jam so sweet.*
- *Lying ill*
- *Means you will*
- *become too thin -*
- *Merely skin*
- *Cov'ring bone;*
- *Regretted tone.*
- *Eat again,*
- *Avoid the fen.*
- *God grant thee*
- *Be healthy.*
- *This I wish,*
- *My sweet dish.*

How Does this Apply?

You have two related problems:

- the "syntax" of French is something you have to learn
- the "semantics" of poetry is something you have to learn

You have two problems you have to solve at the same time.

Programming, Syntax and Semantics

- You have to learn the “syntax” of a particular programming language
  - many details about the language, how to debug and use it
- You have to learn about “problem solving” and how to put it down on “computer.”
- There probably is no better way. It’s hard!

Computers & Problem Solving?

This is both the problem and difficulty of computers.

- The promise (perhaps the hope) of computers is that, somehow, we can embed our own thoughts in them. To some extent we can!
- The problem is the difficulty of doing so, and the stringent requirements, the real rigor, required to put simple “thoughts” into a working program.

Focus of Computer Science

There are two foci for computer science

- Learning the difficult task of truly "laying out" a problem-solving task
- Providing tools to make this process as easy (though it will never be "easy") as possible.

Your focus should be on problem-solving, and adding rigor/focus to your ability to do problem-solving.

Good Program (1)

What makes a good program?

- a program is a reflection of the writer and their thoughts
- First, you must have some thoughts! The difficulty for most people is to figure out what has to be done, the problem solving, before writing a program

Rule 1

- Think before you program!

Good Program (2)

- It will be said repeatedly that the goal of a program **<u>is not</u>** to run, but to be read.
- A program communicates with other people as well. It stands as a document to be read, repaired and, yes, run

Rule 2

- A program is a human-readable essay on problem solving that also happens ot execute on a computer.

Why Python?

Why are we learning with python?

Why Python (1): Simpler

- Python is a "simpler" language than C++
- Simpler means:
  - Fewer alternatives (one way to do it)
  - Better alternatives (easier to accomplish common tasks)
- This allows us to focus less on the language and more on problem solving

Why Python(2): Interactive

- C++ requires an intermediate step before you can run a program, compiling.
- Python allows you to type program statements into the python window and *<u>see results immediately</u>*
- Better for experimenting (which you *<u>need</u>* to do)

Why Python(3): User Base

- While we want to (and will) teach the fundamentals of computer science, we want what you learn to be "useful"
- Python is used in many areas to solve problems related to that field. Many packages are available to help for a particular area

Why Python (4): Useful

- C++ is a good language, especially for majors. It teaches a level of detail that is needed
- Python is more generally "useful", you can do things with it quickly. If you only take this course in CS, you will learn something fundamental *and* practical.

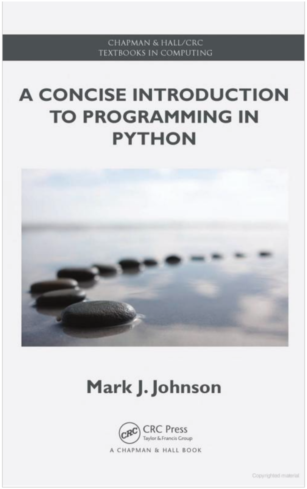
Computational Thinking

Having finished this course, we want you to have the following thought in your subsequent college and career.

“Hey, I’ll just write a program for that”. For us, that is “computational thinking”

Python allows this to happen more readily.

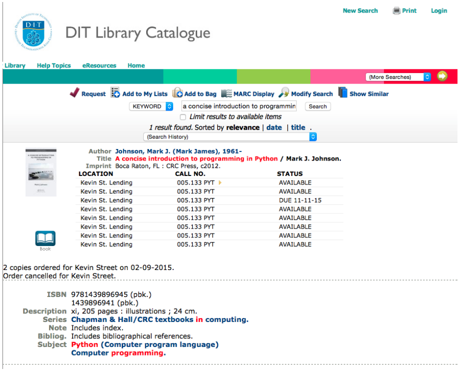
Textbook



A Concise Introduction to Programming in Python. CRC Press, 02/2012.

Mark J. Johnson

Textbook: Available in the Library



1 result found. Sorted by **relevance** | date | title .

Author Johnson, Mark J. (Mark James), 1961-
Title A concise introduction to programming in Python / Mark J. Johnson.
Imprint Boca Raton, FL : CRC Press, c2012.

LOCATION	CALL NO.	STATUS
Kevin St. Lending	005.133 PYT	AVAILABLE
Kevin St. Lending	005.133 PYT	AVAILABLE
Kevin St. Lending	005.133 PYT	DUE 11-11-15
Kevin St. Lending	005.133 PYT	AVAILABLE
Kevin St. Lending	005.133 PYT	AVAILABLE
Kevin St. Lending	005.133 PYT	AVAILABLE
Kevin St. Lending	005.133 PYT	AVAILABLE
Kevin St. Lending	005.133 PYT	AVAILABLE

2 copies ordered for Kevin Street on 02-09-2015.
Order cancelled for Kevin Street.

ISBN	9781439896945 (pbk.)
	1439896941 (pbk.)
Description	xi, 205 pages : illustrations ; 24 cm.
Series	Chapman & Hall/CRC textbooks in computing.
Note	Includes index.
Bibliog.	Includes bibliographical references.
Subject	Python (Computer program language)
	Computer programming.

There are 10-12 copies available in Kevin St. Library

Not a mandatory text!

An excellent way to keep up and work outside of class

Online Resources

- http://learnpythonthehardway.org/book/intro.html
- https://www.codecademy.com/tracks/python
- http://www.tutorialspoint.com/python/index.htm

Computer Systems and Software

- CPU: Central Processing Unit
- RAM: Random Access Memory
- volatile: if you turn off the computer it is gone
- SSD/HDD: Solid state drive, hard disk drive
- persistent storage: saves data even if you power off the machine
- high level language: Python, C++ etc.
- instruction set: assembly language - basic commands: load data, perform arithmetic, store data
- machine language: what the CPU can processs

What happens inside a computer when your program runs?

Languages

Level	Language	Purposes
Higher	Python	Scripts
	Java	Applications
	C, C++	Applcations, Systems
	Assembly Languages	Specialized Tasks
Lower	Machine Languages	

From: Johnson, Mark J.. A Concise Introduction to Programming in Python. CRC Press, 02/2012. VitalBook file.

Compilation and Interpretation

Compiled Code

Convert the high level language version into a machine readable executable version

Interpreted Languages

Converted on the fly without an executable program

A Python Program

```
from math import pi

r = 12
area = pi * r ** 2
print("Area of a circle with radius",
      r,"is", area)
```

Variables, Statements and Syntax

Variables

Names that refer to data in memory - can be anything but make them sensible!* In python: A-Z, a-z, _, and 0-9**

*except for reserved *keywords* - more later **except for the first character of the variable name

Variables, Statements and Syntax

Variables

Names that refer to data in memory - can be anything but make them sensible!* In python: A-Z, a-z, _, and 0-9**

*except for reserved *keywords* - more later **except for the first character of the variable name

Statements

A program is a sequence of *statements* that Python *interprets* and executes

Variables, Statements and Syntax

Variables

Names that refer to data in memory - can be anything but make them sensible!* In python: A-Z, a-z, _, and 0-9**

*except for reserved *keywords* - more later **except for the first character of the variable name

Statements

A program is a sequence of *statements* that Python *interprets* and executes

Syntax

Every statement must have the correct *syntax* or form in which it is written.

Variables, Statements and Syntax

Variables

Names that refer to data in memory - can be anything but make them sensible!* In python: A-Z, a-z, _, and 0-9**

*except for reserved *keywords* - more later **except for the first character of the variable name

Statements

A program is a sequence of *statements* that Python *interprets* and executes

Syntax

Every statement must have the correct *syntax* or form in which it is written.

Warning!

Python has different syntax in different versions

Another Python Program

```
# madlib.py
#

adjective = input("Enter an adjective: ")
noun = input("Enter an noun: ")
verb = input("Enter an verb: ")
adverb = input("Enter an adverb: ")
print("A", adjective, noun,"should never",
    verb, adverb)
```

Types of Statements

The example uses 3 types of statements:

Assignment

Used to assign (give) a `variable` a value from an `expression` (left = right)

Print

Used to create output

Import

Used to access standard libraries

Data Types, Expressions and Comments

Data Types

The example contains three types: `Strings`, `Integers` and `Floats`

Data Types, Expressions and Comments

Data Types

The example contains three types: `Strings`, `Integers` and `Floats`

Expressions

Something to evaluate, e.g. input, numeric using arithmetic operations ($+$,$-$,$*$,$**$,/,//)

Data Types, Expressions and Comments

Data Types

The example contains three types: `Strings`, `Integers` and `Floats`

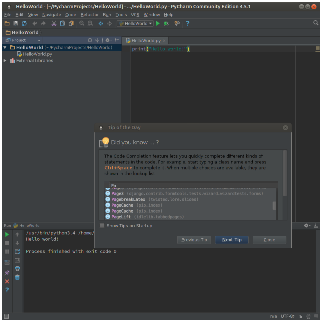
Expressions

Something to evaluate, e.g. input, numeric using arithmetic operations ($+$,$-$,$*$,$**$,$/$,$//$)

Comments

Comments begin with a # for single line comments. They are not interpreted - everything after the # is ignored

PyCharm


HelloWorld - [~/PycharmProjects/HelloWorld] - .../HelloWorld.py - PyCharm Community Edition 4.5.1
File Edit View Navigate Code Refactor Run Tools VCS Window Help
HelloWorld
Project
HelloWorld (~/PycharmProjects/HelloWorld)
HelloWorld.py
External Libraries
Tip of the Day
Did you know ... ?
Show Tips on Startup
Previous Tip
Next Tip
Close
Run HelloWorld
n/a UTF-8

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

The Code Completion feature lets you quickly complete different kinds of statements in the code. For example, start typing a class name and press Ctrl+Space to complete it. When multiple choices are available, they are shown in the lookup list.

```
Pa
Page3  (django.contrib.formtools.tests.wizard.wizardtests.forms)
PagebreakLatex  (twisted.lore.slides)
PageCache  (pip.index)
PageCache  (pip.index)
PageLift  (idlelib.tabbedpages)
```

```
/usr/bin/python3.4 /home/
Hello world!

Process finished with exit code 0
```

PyCharm is an Integrated Development Environment (IDE)

It is used for programming in Python. It provides code analysis, a graphical debugger, and integrated unit testing and source control.

Today's Lab: 11:00 - 13:00

This aim of this lab is to familiarise yourself with:

1. Logging on to lab machines
2. Accessing webcourses (Caveat: you may not be registered in the module yet)
3. Running PyCharm
4. Creating a project, a python file, and running a simple program
5. Using online learning tutorials for Python

Next Thursday's Lab (24 September)

We will recap on some of these items with new exercises and introduce some concepts that will be discussed in the next lecture

Next Lab

We will recap on some of these items with new exercises and introduce some concepts that will be discussed in the next lecture

Next week's Lecture

Details on assessments, project etc.

THE
WEEKEND
STARTS!
HERE.