

CLASE AUXILIAR #0

Introducción a Python

Francisco Gutiérrez F. (frgutier@dcc.uchile.cl)

CCI001 - Computación I (Sección 3)

14 de marzo, 2011

PLAN DE LA CLASE

- Cómo Instalar Python
- Intérprete Python
- Scripts
- Funciones
- Algunos Problemas

CÓMO INSTALAR PYTHON

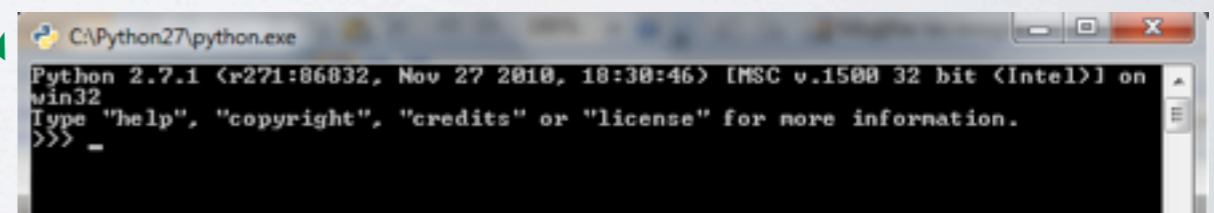
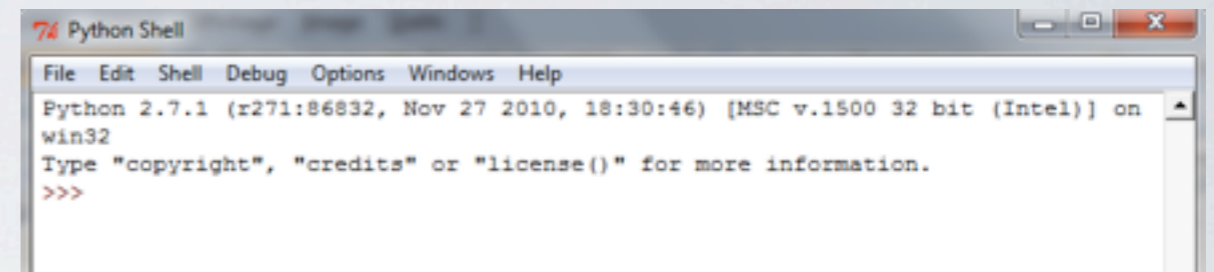
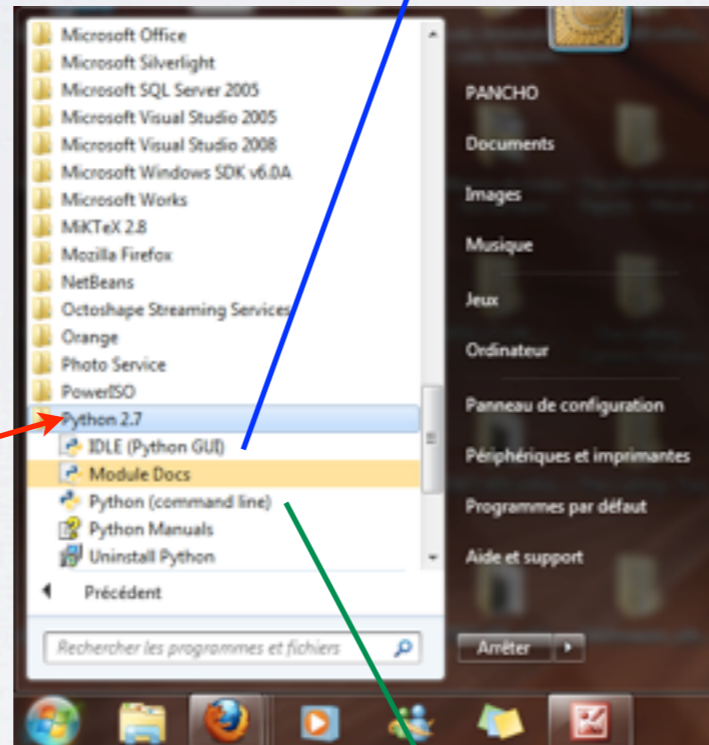
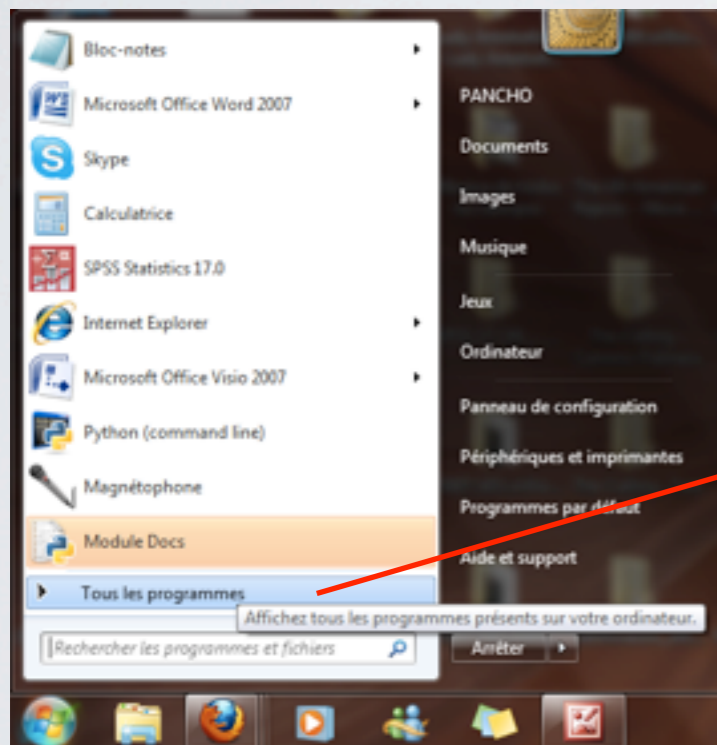
- Descargar el paquete de instalación
 - ➔ <http://www.python.org/download>
- USAREMOS LA VERSIÓN 2.x (actualmente 2.7)
- Ver archivos en UCursos para los pasos de instalación

INTÉRPRETE PYTHON

- Permite utilizar de Python de manera rápida
- Responde automáticamente a las instrucciones
- Windows: Inicio / Programas / Python
- 2 «sabores»: Interfaz Gráfica - Command Line
- Linux - Mac: Ir a la Consola (Terminal) y ejecutar **python**

INTÉRPRETE PYTHON

Windows



INTÉRPRETE PYTHON

Mac / Linux

The diagram illustrates the process of starting the Python interpreter. It consists of two terminal window screenshots. The top window, titled 'Terminal — bash — 80x24', shows a standard shell prompt: 'MacBook-de-Francisco-GUTIERREZ:~ fgutierrez\$'. A vertical arrow points from this prompt down to the second window. The second window, titled 'Terminal — python2.7 — 80x24', shows the same prompt with the word 'python' entered and highlighted by a red circle. Below the prompt, the terminal displays the Python version and build information: 'Python 2.7.1 (r271:86882M, Nov 30 2010, 10:35:54) [GCC 4.2.1 (Apple Inc. build 5664)] on darwin'. It also provides instructions: 'Type "help", "copyright", "credits" or "license" for more information.' and shows the prompt changing to '>>>'.

```
Terminal — bash — 80x24
MacBook-de-Francisco-GUTIERREZ:~ fgutierrez$

Terminal — python2.7 — 80x24
MacBook-de-Francisco-GUTIERREZ:~ fgutierrez$ python
Python 2.7.1 (r271:86882M, Nov 30 2010, 10:35:54)
[GCC 4.2.1 (Apple Inc. build 5664)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

INTÉRPRETE PYTHON

- Es rápido, interactivo y automático...
- PERO no nos permite guardar lo realizado para ejecutarlo en una nueva sesión (como por ejemplo, cuando tengan que hacer tareas)
- SOLUCIÓN: hacer un script

SCRIPTS

- Archivos de texto en un formato especial que contienen el código del programa (y las funciones que hagamos en él) para ejecutarlo posteriormente con el intérprete de Python
- Se crean con cualquier editor de texto y llevan la extensión **.py**

SCRIPTS

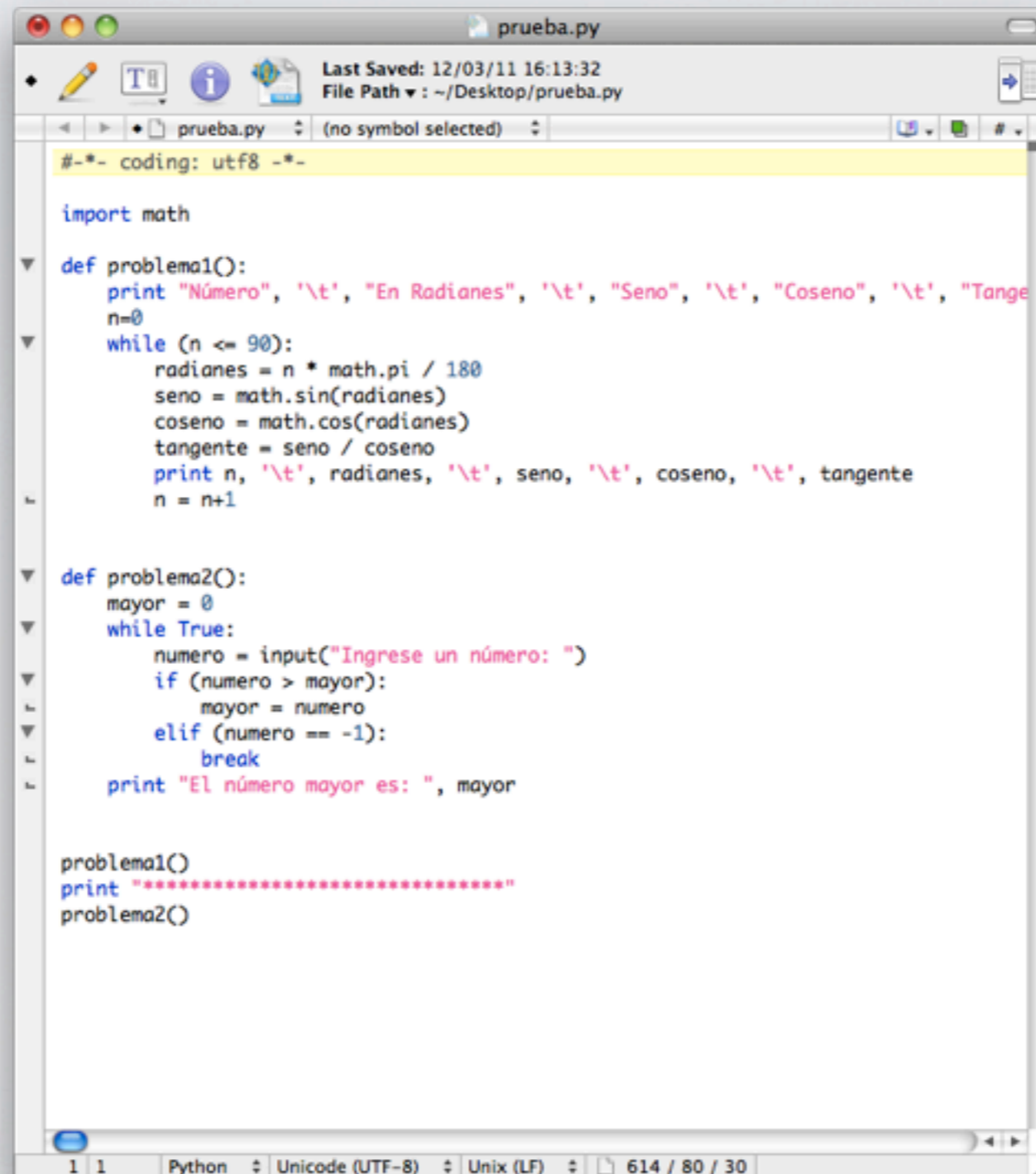
- «*Se crean con cualquier editor de texto [...]*» Sí, pero siempre nos podemos simplificar un poco la vida :-)
- Existen editores de texto para programar que traen incorporada una funcionalidad de **coloración sintáctica**, es decir, las palabras clave son resaltadas con otro color y nos ayudan a ordenar mejor el código.

SCRIPTS

- Mi recomendación (personal):

Windows	<i>Notepad++</i>	http://notepad-plus-plus.org/
Mac OS X	<i>TextWrangler</i>	Disponible en la App Store
Linux	<i>jEdit / Kate</i>	Ya incluidos en algunas distribuciones

SCRIPTS



```
prueba.py
Last Saved: 12/03/11 16:13:32
File Path: ~/Desktop/prueba.py

#-*- coding: utf8 -*-

import math

def problema1():
    print "Número", '\t', "En Radianes", '\t', "Seno", '\t', "Coseno", '\t', "Tange"
    n=0
    while (n <= 90):
        radianes = n * math.pi / 180
        seno = math.sin(radianes)
        coseno = math.cos(radianes)
        tangente = seno / coseno
        print n, '\t', radianes, '\t', seno, '\t', coseno, '\t', tangente
        n = n+1

def problema2():
    mayor = 0
    while True:
        numero = input("Ingrese un número: ")
        if (numero > mayor):
            mayor = numero
        elif (numero == -1):
            break
    print "El número mayor es: ", mayor

problema1()
print "*****"
problema2()
```

1 1 Python Unicode (UTF-8) Unix (LF) 614 / 80 / 30

SCRIPTS

- ¿Cómo ejecutar un Script de Python?
- Generalmente, basta con hacer doble click al archivo.
- Si no, en la consola hay que ir al directorio donde esté el archivo y escribir: **python <nombreArchivo.py>**

MI «PRIMER» SCRIPT

- Dado un número ingresado por el usuario, el computador entrega su sucesor

Diálogo a seguir:

Ingresa un numero: *172*

Su sucesor es: *173*

MI «PRIMER» SCRIPT

Algoritmo

pedir número
leer número

sucesor \rightarrow numero + 1

imprimir sucesor

Python

```
print "Ingrese un numero: "
```

```
numero = input()
```

```
sucesor = numero + 1
```

```
print "El sucesor es: ", sucesor
```

FUNCIONES

- Cuando queremos resolver un problema, generalmente hay partes de código que reutilizaremos más de una vez. Esto lo conocemos como una **función**.
- En un sentido estricto, es igual que una función matemática: *dado uno o más valores de entrada, devolvemos otro.*

FUNCIONES

- En Python definimos una función con la palabra clave **def** y devolvemos un valor al programa principal con la palabra clave **return**
- Si consideramos el ejemplo anterior, tendríamos:

```
def sucesor(numero):
```

```
    numeroSucesor = numero + 1
```

```
return numeroSucesor
```

FUNCIONES

- OJO CON LA SINTAXIS: no olvidar los **dos puntos al final de la declaración** y la **indentación**
- Python es uno de los pocos lenguajes de programación donde **LA INDENTACIÓN DE LOS BLOQUES ES OBLIGATORIA**
- Como recomendación, usen nombres de funciones y variables que sean representativas al problema (es más fácil leer y entender lo que hace el código, tanto ustedes como alguien que lee por primera vez lo que hicieron)

FUNCIONES

- Es posible crear un archivo con una lista de funciones y usarlas en un programa escrito en un archivo distinto
- Para ello, usamos la palabra clave **import** (es decir, «importamos» las funciones implementadas en otra parte para usarlas en este programa
- Ejemplos que ya han visto en clases: **math** y **random**

¿CONSULTAS?

PROBLEMAS DE APLICACIÓN

- Dado un número de 3 cifras ingresado por el usuario, devolver el resultado de la suma de este número con su forma invertida (Si recibo 123, devuelvo $123 + 321 = 444$)
- Dados los valores de dos lados de un triángulo cualquiera y el ángulo (en grados) entre ambos, calcular el perímetro y el área de dicho triángulo

PARA IR MÁS ALLÁ...

- «*How to think like a Computer Scientist: Learning with Python*»

Allen Downey, Jeffrey Elkner, Chris Meyers

Este libro les puede servir como complemento a las clases de cátedra. Trata prácticamente todos los temas que veremos durante el semestre y viene con algunos ejemplos y ejercicios propuestos.

Lo pueden descargar gratuitamente (licencia GNU) desde:

<http://greenteapress.com/thinkpython/thinkpython.html>

PRÓXIMA CLASE

- Viernes 18 de marzo - 8:30 horas
- CÁTEDRA: Condicionales (If)