

Inspira Crea Transforma

Programación de Computadores

ST0240

Semestre 2017-1

48 horas semestral

Fundamentos de Programación

ST0286

Semestre 2017-1

48 horas semestral

Agenda

30 m Contenido Previo (Tareas, Lecturas, Ejercicios propuestos o Anterior)

50 m Contenido de Clase

20 m Break

50 m Ejercicios Prácticos

30 m Cierre: Trabajo Individual (Por fuera de Clase)

Presentación: Coordinadora de la Materia

Elizabeth Suescún

Ingeniera Informática Politécnico Jaime Isaza
Caldavid

DSc. Pontificia Universidad Católica de Rio de Janeiro

Oficina: Bloque 19 - 628

Correo: esuescu1@eafit.edu.co



Presentación: Coordinador de la Materia

Juan Francisco Cardona McCormick

Ingeniero de Sistemas Universidad EAFIT

Candidato a Doctor en Universidad de

Oficina: Bloque 19 - 4 piso - 4

Correo: fcardona@eafit.edu.co



Presentación

Profesores:

Sergio Andrés Monsalve Castañeda

Ingeniero de Sistemas

Candidato a Máster en Ingeniería U. EAFIT

Oficina: Bloque 19 - 4 piso - 408

Correo: smonsal3@eafit.edu.co



Juan David Pineda Cárdenas

Ingeniero de Sistemas

Candidato a Máster en Seguridad de las
TIC - Universitat Oberta de Catalunya

Coordinador Técnico - Centro de
Computación Científica APOLO

Oficina: Bloque 19 - 4 piso - 407

Correo: jpineda2@eafit.edu.co



Carlos Alberto Álvarez Henao

Ingeniero Civil

DSc. Mecánica Computacional - UFRJ/Brasil

Correo: calvar52@eafit.edu.co



Luisa Fernanda Villa Montoya

Ingeniera

PhD.

Correo: lvillamo@eafit.edu.co



Presentación

Profesores:

John Jairo Silva Zuluaga

Ingeniero de Sistemas

Candidato a Máster en Ingeniería U. EAFIT

Correo: jsilvazu@eafit.edu.co



César Augusto Ruiz Jaramillo

Magister en Ingeniería U. EAFIT

Correo: cruijar@eafit.edu.co



Myriam Castro Cárdenas

Ingeniera de Sistemas

Magister en Administración U. EAFIT

Oficina: Bloque 19 - 6 piso - 622

Correo: mcastro@eafit.edu.co



Andrés Felipe García Restrepo

Magister en Ingeniería UPB

Estudiante de Doctorado UPB

Correo: afgarciar@eafit.edu.co



Monitores

Nombre	Correo	Horario
Marcos Sierra	@eafit.edu.co	Lunes 1 p.m. - 5 p.m. Martes 3 p.m. - 6 p.m. Lugar:
PENDIENTE	@eafit.edu.co	pendiente

- Pendiente para la Segunda Semana*
- Monitor en Clase
 - Preguntas cortas con cita previa por correo electrónico
 - Horarios de Apoyo

Programa

Semana	Fecha	Contenido	Actividad Evaluativa
1	Enero 23-27	Presentación e Introducción, Lenguajes Compilados vs. Interpretados, IDEs, Tipos y variables, Expresiones, Secuencias, Entrada y Salida 1	
2	Enero 30 - Febrero 3	Condiciones, Ciclos	Taller 1 (5 %)
3	Febrero 6 - 10	Rangos, Funciones	
4	Febrero 13 - 17	Arreglos, Cadenas, Diccionarios, Listas	Taller 2 (5 %)
5	Febrero 20 - 24	Entrada y Salida 2	
6	Febrero 27 - Marzo 3	Excepciones	Taller 3 (10 %)
7	Marzo 6 - 10	Estructuras de Datos (Contenedores): JSON	
8	Marzo 13 - 17	Algoritmos 1: Ordenamiento	Taller 4 (10 %)
9	Marzo 20 - 24	Algoritmos 2: Búsqueda	

Programa

Semana	Fecha	Contenido	Actividad Evaluativa
10	Marzo 27 - 31	Programación Orientada a Objetos	Taller 5 (10 %)
11	Abril 3 - 7	Algoritmos 3	
--	Abril 10 - 14	SEMANA SANTA	NO HAY CLASE
12	Abril 17 - 21	Visualización de Información	
13	Abril 24 - 28	Matlab	
14	Mayo 1 - 5	Matlab	Práctica 1 (15 %)
15	Mayo 8 - 12	Visual Basic for Applications VBA(excel): Programación Orientada a Objetos	
16	Mayo 15 - 19	Visual Basic for Applications VBA(excel)	Práctica 2 (15 %)
17	Mayo 22 - 26	Colchón	
18	Mayo 29 - Junio 2	Finales	Examen Final (20%)

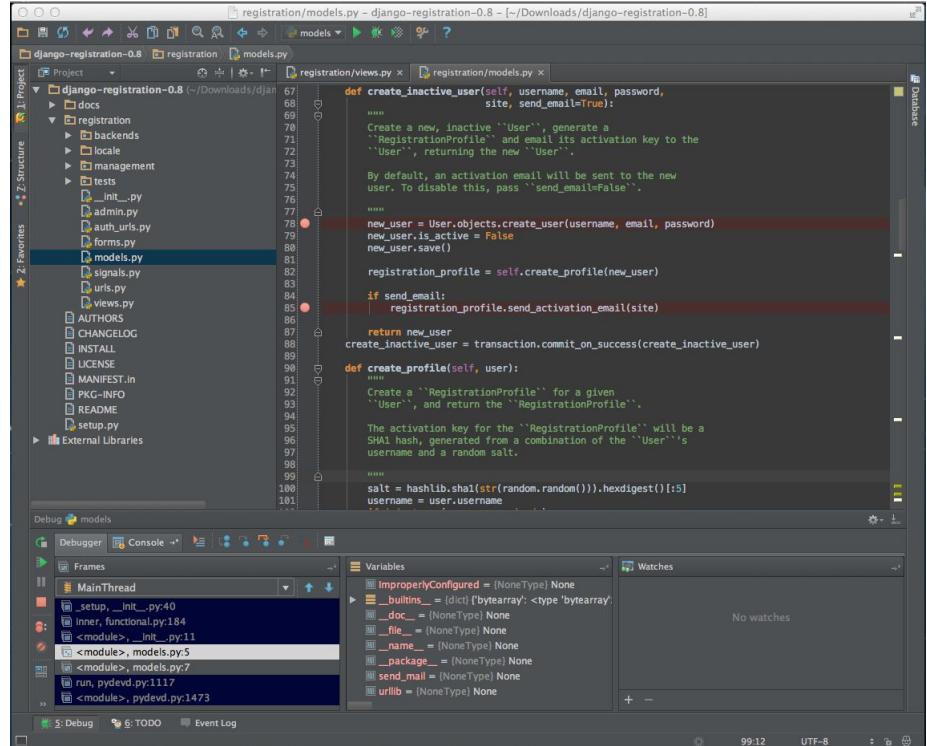
Evaluación

Actividad	Valor	Semana
Taller 1	5%	Semana 2
Taller 2	5%	Semana 4
Taller 3	10%	Semana 6
Taller 4	10%	Semana 8
Taller 5	10%	Semana 10
Práctica 1	15%	Semana 14
Práctica 2	15 %	Semana 16
Práctica Final	20%	Semana 18
Seguimiento	10%	A lo largo del semestre

Editores e IDE's

Integrated Development Environment

- Oficial: Pycharm -
<https://www.jetbrains.com/pycharm-edu/>
- Alternativo: Python Anywhere -
<https://www.pythonanywhere.com>
- Otros:
 - Jupyter - <http://jupyter.org>
 - IdeOne - <http://ideone.com>
 - Sublime - <https://www.sublimetext.com>
 - Atom - <https://atom.io>



The screenshot shows the PyCharm IDE interface. The code editor displays a Python file named `models.py` from the `registration` package of the `django-registration-0.8` project. The code defines several methods for creating users and profiles. The project structure on the left shows files like `__init__.py`, `admin.py`, and `views.py`. The bottom toolbar includes tabs for Debug, Debugger, Console, and others. The Variables and Watches panes are visible on the right, showing current variable values.

```
def create_inactive_user(self, username, email, password, site, send_email=True):
    ...
    Create a new, inactive ``User``, generate a ``RegistrationProfile`` and email its activation key to the ``User``, returning the new ``User``.

    By default, an activation email will be sent to the new User. To disable this, pass ``send_email=False``.

    ...
    new_user = User.objects.create_user(username, email, password)
    new_user.is_active = False
    new_user.save()

    registration_profile = self.create_profile(new_user)

    if send_email:
        registration_profile.send_activation_email(site)

    return new_user
create_inactive_user = transaction.commit_on_success(create_inactive_user)

def create_profile(self, user):
    ...
    Create a ``RegistrationProfile`` for a given ``User``, and return the ``RegistrationProfile``.

    The activation key for the ``RegistrationProfile`` will be a SHA1 hash, generated from a combination of the ``User``'s username and a random salt.

    ...
    salt = hashlib.sha1(str(random.random()).hexdigest()[:5])
    username = user.username
    ...

salt = hashlib.sha1(str(random.random()).hexdigest()[:5]
username = user.username
```

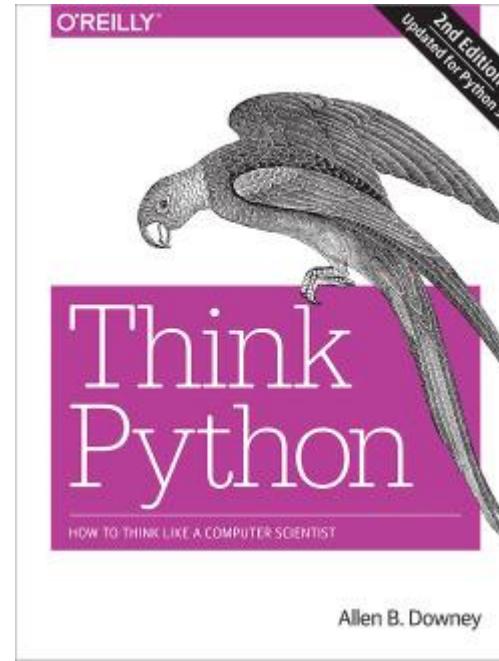
Libros Guía

- Farrell, Joyce.
Introducción a la
programación Lógica
y Diseño. 7a Ed.
México: Cengage
Learning, 2013.



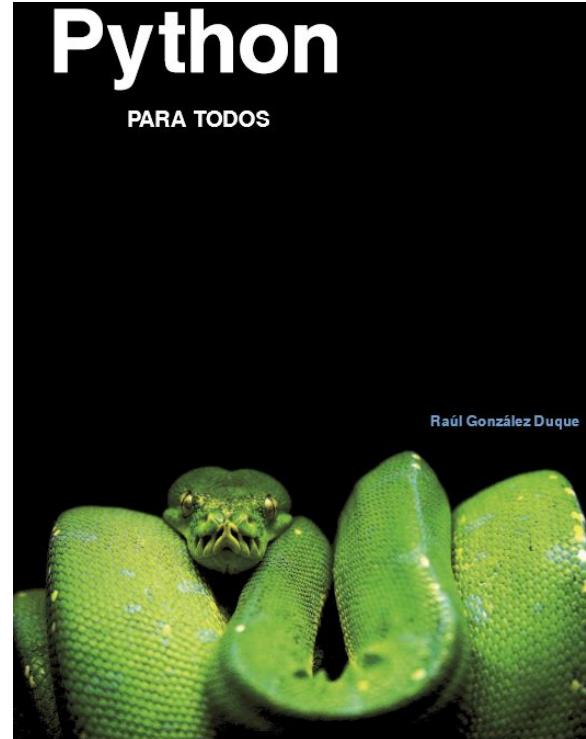
Libros Guía

- Downey, Allen B.
Think Python. 2a Ed.
Green Tea Press.
- Descargable gratuitamente desde
<http://greenteapress.com/wp/think-python-2e/>



Libros Guía

- Raul Gonzalez Duque.
Python para todos.
- Se puede descargar la versión más reciente de este libro gratuitamente en:
<http://mundogeek.net/tutorial-python/>



Objetivos generales del curso

Al finalizar este curso, el estudiante:

- Tendrá bases lógicas y técnicas para la programación de un computador;
- Poseerá conocimientos suficientes que le permitirán organizar y escribir instrucciones para un computador, es decir, programarlo de una manera eficaz mediante la utilización de un lenguaje de programación;
- Erradicará la creencia que llevan algunas personas de que la programación es privilegio sólo de pocos, y así se sentirá bastante satisfecho de tener un computador a “sus órdenes”;
- Comprenderá que no saber programar un computador, implica una disminución, en un gran porcentaje, del producto del trabajo realizado con el mismo, a pesar del software tan desarrollado existente en el mercado.

Todos Deberían Aprender a Programar - www.code.org



Creo que uno de los errores más comunes sobre la informática y la programación en general es que

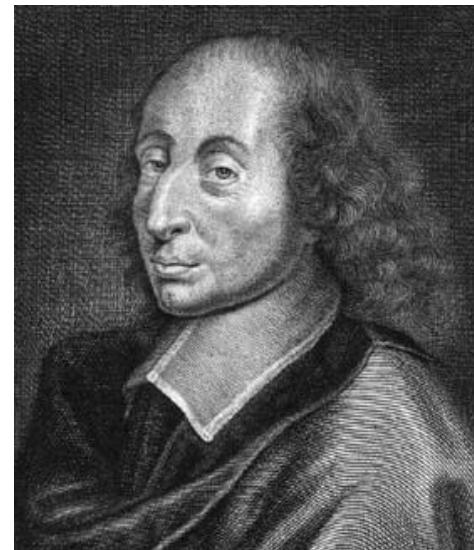
Historia de la Computación

Blaise Pascal (1623 - 1662)

Matemático, físico, filósofo y escritor.

Contribuciones a la matemática, estadística,
historia natural y calculadoras mecánicas.

Después de una experiencia religiosa profunda,
abandonó la matemática y la física para dedicarse
a la filosofía y la teología.

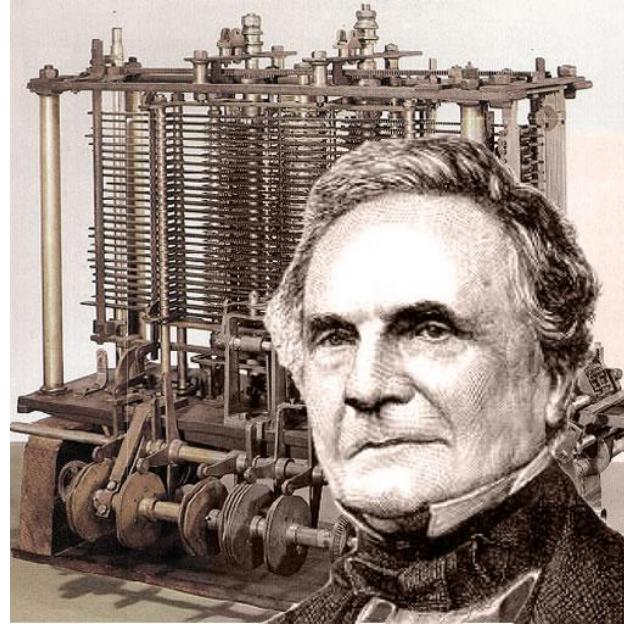


Tomado de: <http://www-history.mcs.st-and.ac.uk/>

Historia de la Computación

Charles Babbage (1791- 1871)

Conocido como el padre de la computación por su contribuciones al diseño básico del computador a través de su “máquina analítica”.



Tomado de: <http://www.greatthoughtstreasury.com/>

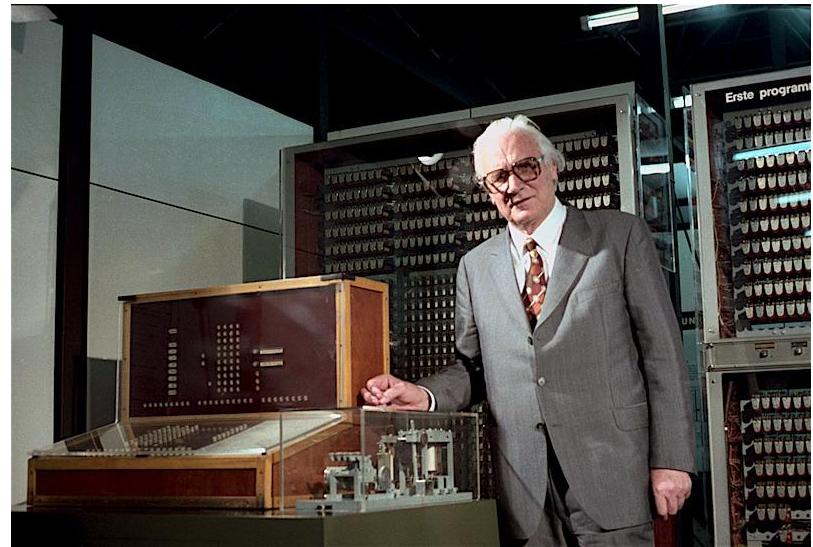
Historia de la Computación

Konrad Zuse (1910 - 1995)

Alemán, ingeniero civil, inventó ante de la segunda guerra el computador electromecánico conocido como Z1, el cual fue destruido por bombas en tiempos de guerra.

Desarrolló otras dos máquinas antes de que la guerra terminara pero no pudo convencer al gobierno nazi que apoyara su trabajo. Voló a Zurich donde desarrolló la Z4. Desarrolló un lenguaje de programación básico conocido como Plankalkul, con el cual desarrolló un juego de ajedrez.

Es considerado el inventor del primer computador digital programable.



Tomado de: <https://www.german-way.com/>

Historia de la Computación

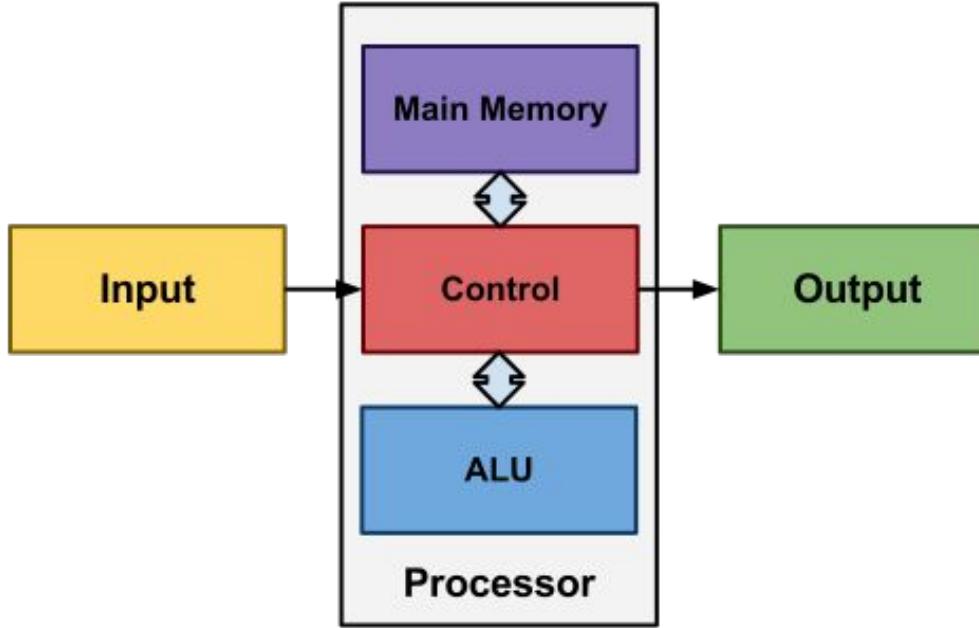
John von Neumann (1903 - 1957)

Matemático Húngaro con grandes aportes a física cuántica, análisis funcional, teoría de conjuntos, teoría de juegos, ciencias de la computación, economía, análisis numérico, cibernetica, hidrodinámica, estadística, criptografía entre otros.

Gracias a él tenemos la arquitectura de von Neumann, arquitectura que prevalece hasta los computadores de nuestros días.



Tomado de: <https://www.wikipedia.org/>



Tomado de: <http://cs.uwec.edu/>

Historia de la Computación

Alan Turing (1912 - 1954)

Considerado el padre de la computación moderna. Tenemos la máquina universal gracias a él, también conocida como la máquina de turing.

Cómo matemático, aplicó el concepto de algoritmo a los computadores digitales.

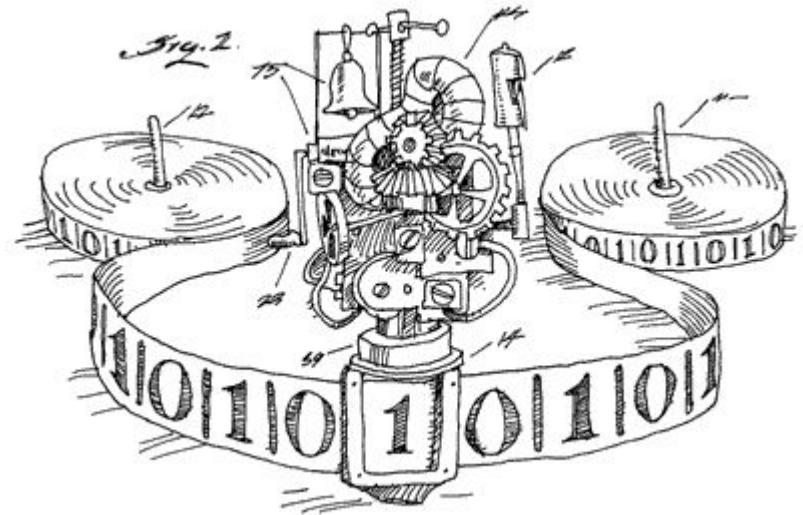
Su investigación es acerca de la inteligencia artificial.

Fué uno de los pioneros en el concepto de computador digital.



Tomado de: <https://k31.kn3.net>

Alan Turing describió una máquina que debería leer una serie de unos y ceros desde una cinta. Estos unos y ceros describen los pasos que se requieren para llevar a cabo la solución de un problema en particular o realizar cierta tarea. La máquina de Turing debería leer cada uno de los pasos y realizarlos en secuencia, resultando en la respuesta adecuada.



Tomado de: <http://www.worldofcomputing.net/>

Inspira Crea Transforma

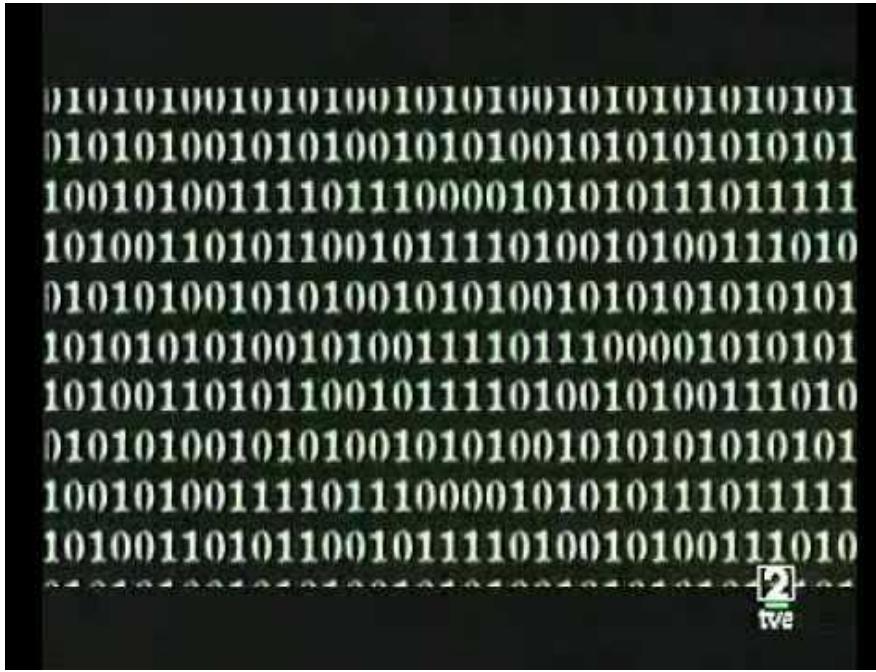
**UNIVERSIDAD
EAFIT®**

Discusión

¿Porqué es importante la computación en la historia
de la humanidad?

¿Pros y Contras?

Tarea: Documental Código Linux



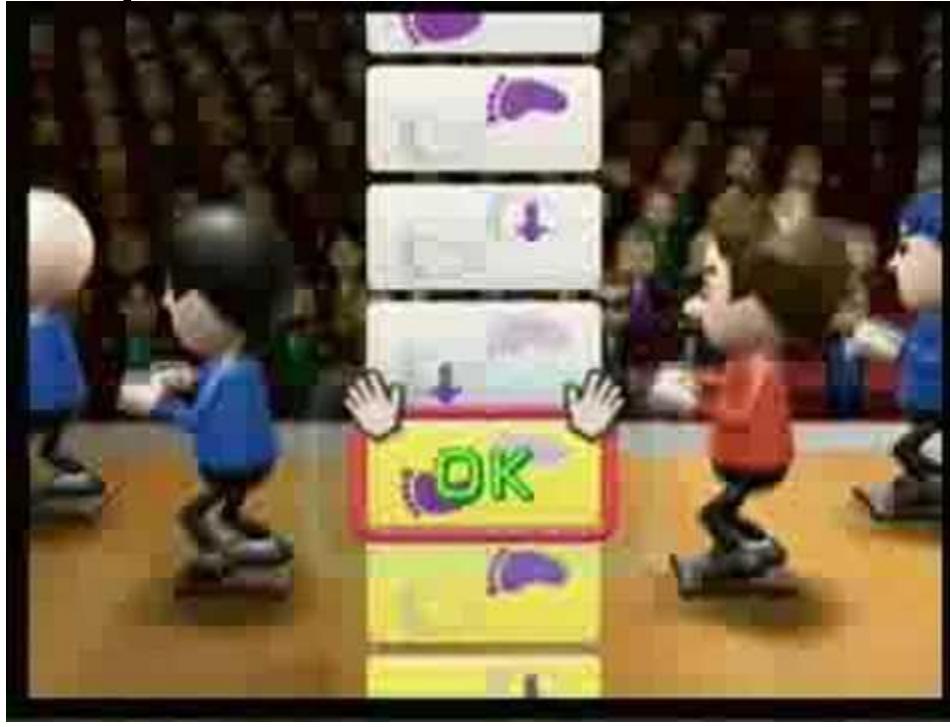
¿Por qué programar?

- Los computadores hacen cosas por nosotros
- Debemos aprender cómo hablar con ellos para hacer que hagan lo que necesitamos
- Si ellos se equivocan es porque nosotros nos equivocamos
- Ya no somos usuarios, ahora somos programadores.
- El medio por el cual nos comunicamos y damos instrucciones a los computadores es por medio de los lenguajes de programación.



Tomado de: <http://www.newenglandcollegeonline.com/>

Programas para Humanos



Inspira Crea Transforma

UNIVERSIDAD
EAFIT®

Programas para Máquinas

The screenshot shows a desktop application window with two panes. The left pane is a 'Python Shell' window with a blue header bar containing 'File Edit Shell Debug Options Windows Help'. Below the header, it displays the Python version and build information: 'Python 3.3.0 (v3.3.0:bd8afb90ebf2, Sep 29 2012, 10:57:17) [MSC v.1 D64] on win32'. It also shows a command-line session starting with '>>> print("Hello World")' followed by 'Hello World' and an empty line. The right pane is titled '*FirstApp' with a file icon. It contains a single line of code: 'print("Hello World")'. Above this code, there is a timestamp: 'Created on Apr 17, 2012' and a comment: '@author: pro'.

Most Popular Coding Languages of 2016

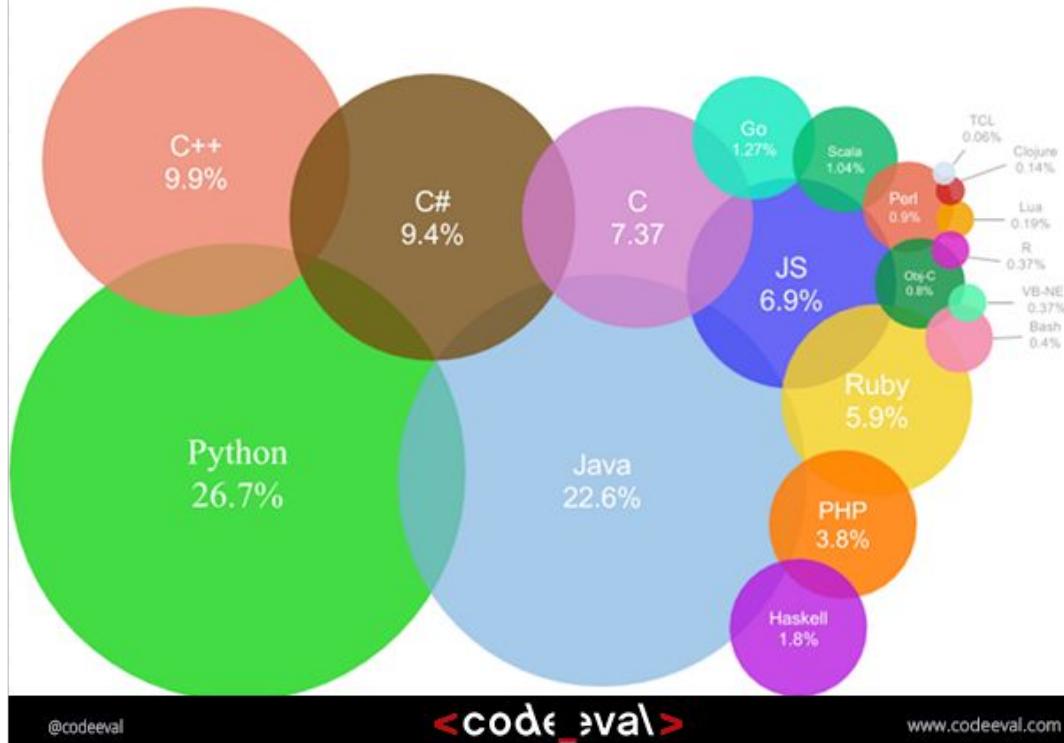
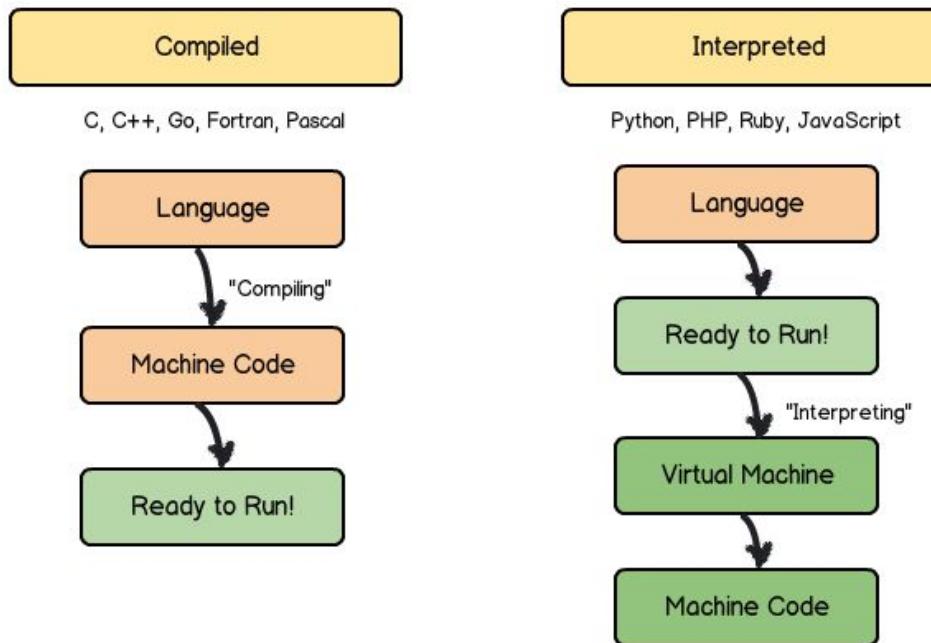




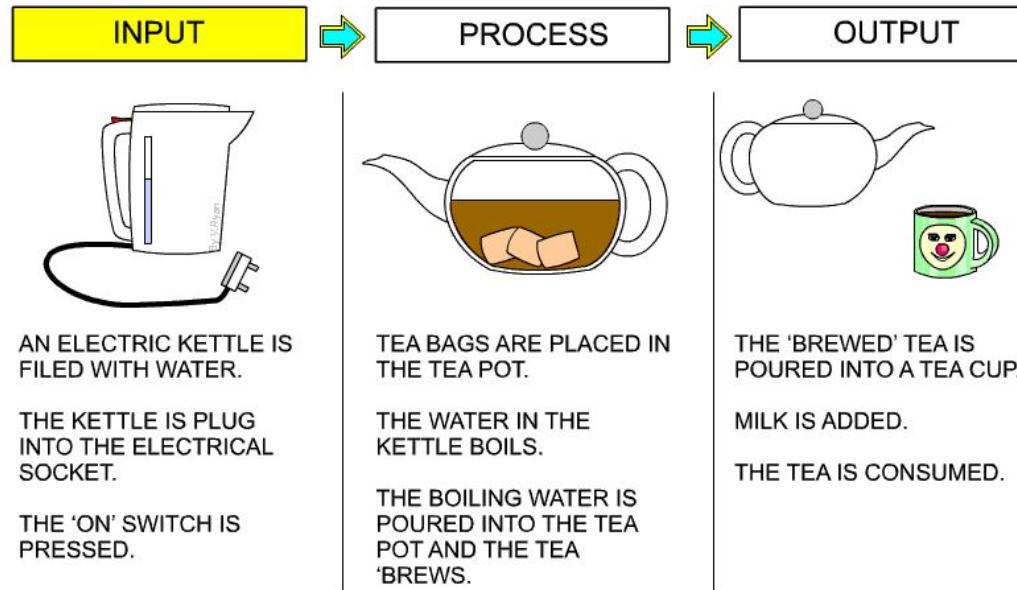
Imagen tomada de [artículo](#) de Universia

Diferencia entre Lenguajes Compilados e Interpretados



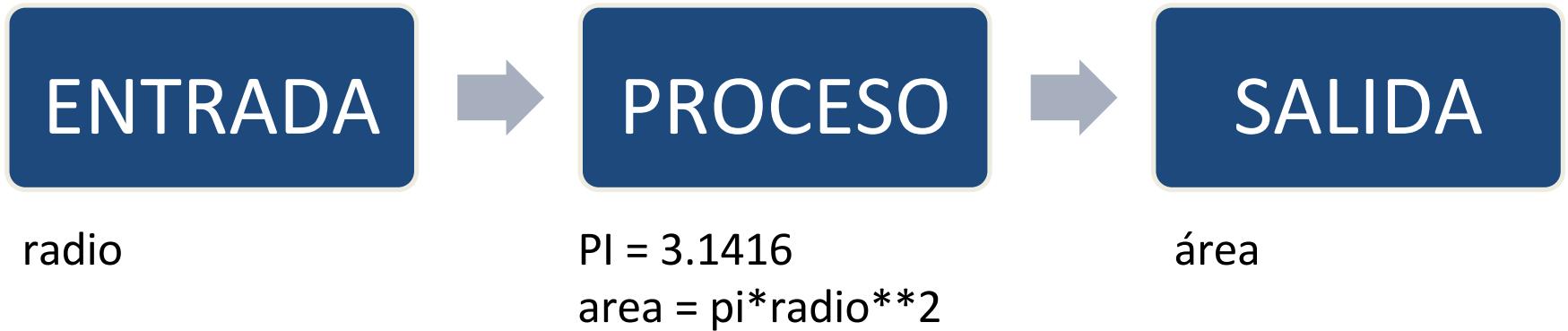
Tomado de: <http://shawnbiddle.com/>

SAMPLE SYSTEMS DIAGRAM



Tomado de: <http://www.technologystudent.com/>

Ejemplo. Calcular el área de un círculo ($A=\pi r^2$).



Sintaxis vs. Semántica

- Semántica / Significado
 - Violación semántica: “El sol sube arriba spotify hace dos días sin netflix”
 - Es español pero no significa nada!
- Sintaxis / Forma
 - Violación sintáctica: estoy yo de pie
 - Puedes inferir qué es pero no está bien dicho, no sigue las reglas del español



Tipos y Variables

- **Números:** (Enteros, Reales, Octal, hexadecimal) Se pueden realizar operaciones matemáticas
- **Cadenas (strings):** Concatena letras y signos para formar cadenas de caracteres
- **Booleanos:** Falso / Verdadero
- **Listas:** Agrupa elementos de diferentes propiedades
- **Tuplas:** ídem, pero de comportamiento diferente
- **Diccionarios:** ídem

Tipos y Variables

- **Variable:** es un espacio para almacenar datos modificables:

```
nombre_de_la_variable = valor_de_la_variable
```

- **Constante:** define valores fijos, que no requieren ser modificados:

```
NOMBRE_DE_LA_CONSTANTE = valor_de_la_constante
```

Operadores Aritméticos

Símbolo	Significado	Ejemplo	Resultado
+	Suma	$a = 10 + 5$	a es 15
-	Resta	$a = 12 - 7$	a es 5
-	Negación	$a = -5$	a es -5
*	Multiplicación	$a = 7 * 5$	a es 35
**	Exponente	$a = 2 ** 3$	a es 8
/	División	$a = 12.5 / 2$	a es 6.25
//	División Entera	$a = 12.5 // 2$	a es 6.00
%	Módulo	$a = 27 \% 4$	a es 3

Operadores Relacionales

Símbolo	Significado	Ejemplo	Resultado
>	Mayor que	$7 > 2$	Verdadero
<	Menor que	$7 < 2$	Falso
\geq	Mayor o igual que	$7 \geq 2$	Verdadero
\leq	Menor o igual que	$7 \leq 2$	Falso
\neq	Diferente	$7 \neq 2$	Verdadero
\equiv	Igual	$7 \equiv 2$	Falso

INCORRECTO: $0 < x < 100$

Operadores Lógicos

Símbolo	Significado	Ejemplo	Resultado
and	Y	$(7 > 2)$ and $(7 \geq 5)$	Verdadero
or	O	$(7 < 2)$ or $(7 \geq 5)$	Verdadero
not	Negación	not $(7 \geq 2)$	Falso

Precedencia de Operadores

¿Qué valor da como resultado la siguiente operación? (¿Cuál es el valor almacenado en “resultado”?)

$$\text{resultado} = 3^{**2} + 5 - 4 / 8 ^{**} 1 / 3$$

Un Lenguaje de Programación puede utilizarse para resolver una cantidad Infinita de Problemas, desde los más sencillos hasta los más complejos por lo tanto puede llegar a ser bastante extenso.

Siempre habrán dudas y aprender a consultar de forma eficaz y eficiente la documentación oficial es una habilidad necesaria para ser un buen programador

Si en algún momento no se recuerda o no se está seguro de un concepto recuerde que la documentación oficial es la fuente mas certera de información:

<https://docs.python.org/3/reference/operators.html#operator-precedence>

Expresiones

Una expresión es una combinación de constantes, variables o funciones, que es interpretada de acuerdo a las normas particulares de precedencia y asociación para un lenguaje particular.

Como en matemáticas, la expresión es su valor evaluado, es decir, la expresión es una representación de ese valor.

- Expresión relacional: $y > 8$
- Expresión aritmética: $3+2$, $x+1$, ...
- Expresión lógica: $x \text{ OR } y$, $\text{NOT } x$, ...
- Expresión con predicados:
 $P(a) \text{ AND } Q(b)$, ...

Tomado de Wikipedia: [https://es.wikipedia.org/wiki/Expresi%C3%B3n_\(inform%C3%A1tica\)](https://es.wikipedia.org/wiki/Expresi%C3%B3n_(inform%C3%A1tica))

Expresiones

a = 10

b = 5

c = 10

a = a + b - 5

b = a + b - 5

c = a + b - 5

a = a + 5 * b / 2

b = a + 5 * b / 2

c = a + 5 * b / 2

¿Qué valores quedan en las variables a, b y c luego de ejecutar todas las instrucciones?

Ejercicio en Casa

- Spoj
 - Ingresar a el link: <http://www.spoj.com/register/>
 - Registrarse con sus datos y el correo de la universidad.
 - Cuando se haya registrado ingresar a:
<http://www.spoj.com/PCEAFIT/problems/main/sort=0,start=0>
 - Resolver el problema: 1 Life, the Universe, and Everything
 - **Recuerde: todo espacio, mayúscula, minúscula, nueva línea (enter), que no haga parte de la respuesta esperada puede hacer que su solución sea catalogada como incorrecta**
- Crear cuenta en Python Anywhere y asignar al profesor.
 - Crear un hola mundo que salude personalizadamente.

Recursos Adicionales

<https://spoj.com/>

<https://ideone.com/>

<https://www.pythontutorial.net/>

<http://python.swaroopch.com/>

<https://pragprog.com/book/gwpy2/practical-programming>

<https://coderbyte.com/course/learn-python-in-one-week>

<https://developers.google.com/edu/python/>



Para la Próxima Clase/Semana

1. Leer Capítulos 1, 2 y 3 del libro
2. Documentales vistos en clase
3. Instalar y realizar los ejemplos de introducción a la programación en python de PyCharm
4. Exponer problemas que hayan tenido en el foro de la materia.
5. Apoyarse en los contenidos y videos de Google Python Class (Visión General, Configuración e Introducción)



Referencias y Lecturas Adicionales

- History of Computing - <http://slideplayer.com/slide/5126850/>
- Python Google Class - <https://developers.google.com/edu/python/>
- Computer Programming Khan Academy -
<https://www.khanacademy.org/computing/computer-programming>
- <https://code.org>
- <http://www.skulpt.org/>
- <https://repl.it/languages/python3>
- <https://www.khanacademy.org>
- <https://www.codecademy.com/es/learn/python>
- <https://www.codeschool.com/courses/try-python/>
- <https://www.edx.org/course/introduction-computer-science-harvardx-cs50x>
- [¿Qué lenguaje de programación aprender?](#)