

# INTRODUCCIÓN A PYTHON

## LECCIÓN 1

# LECCIÓN 0

Applied Mathematics and Actuary Training



## PRESENTACIÓN

LECCIÓN 0 – Introducción

# P R E S E N T A C I O N E S

## Preguntas Básicas

Nombre / Apodo  
Ocupación (explicado)  
Background (académico / laboral)  
Nivel de programación / inglés  
Hobbies/Datos Curiosos

# PRESENTACIÓN EMANUEL MEJÍA

FES Acatlán UNAM – Licenciatura en Actuaría

IPADE Business School – Maestría en Dirección de Empresas para Ejecutivos con Experiencia (MBA/MEDEX)

University of California, Berkeley – Master of Information and Data Science

1

## Certificados:

- SOA – Exámenes P, FM , MFE
- HBS – Credential of Readiness
- Scrum Alliance – Certified Scrum Master
- GMAC – GMAT Focus (Perc 98)
- ETS – TOEFL iBT (Score 115/120)
- BMV Educación – Asesor de Estrategias de Inversión

2

## Experiencia de más de 10 años

### *Desarrollo de emprendimientos de tecnología*

- 2014                      Grupo Converse de México
- 2015 - Actual        Firedrop
- 2020 - Actual        Laboratorios Verktann



# CONTACTO

## Correo electrónico:

- [emanuelmejia@berkeley.edu](mailto:emanuelmejia@berkeley.edu)
- [emanuel@firedrop.mx](mailto:emanuel@firedrop.mx)

## Redes:

- [www.linkedin.com/in/emanuel-mejia-firedrop](https://www.linkedin.com/in/emanuel-mejia-firedrop)
- <https://github.com/emanuelmejia>





# DINÁMICA DE CLASE

LECCIÓN 0 - Introducción

# DINÁMICA DE CLASE

...los tiempos han cambiado... y también para dar clases

- El maestro es un **facilitador** del aprendizaje.
- Los **medios electrónicos** permiten obtener información **actualizada** de cualquier índole (validarla).
- Curso en **zoom**: Destinar un **espacio** para que se puedan concentrar en su.
- No hay necesidad de desconectarse de sus **dispositivos móviles**.
- Habrá presentaciones en Power Point y se harán ejercicios prácticos (comprobar instalaciones).
- Las presentaciones y **cuadernos se enviarán** posterior a la clase,
- Se exhorta a los alumnos a que **hagan todos los ejercicios** en conjunto con el instructor y en su caso tomar notas.
- Los **comentarios y sugerencias** son bienvenidos.

A programar se aprende... PROGRAMANDO!



# LECCIÓN 0

Applied Mathematics and Actuary Training



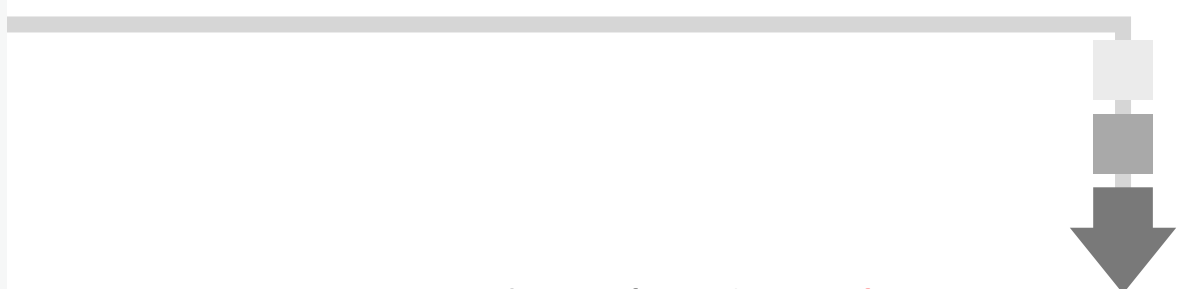
## CONCEPTOS BÁSICOS

LECCIÓN 1 – Conociendo Python



# ¿QUÉ ES UN PROGRAMA?

Gran diversidad de tareas  
que deben ser ejecutadas  
con precisión

- 
- Manipulación de **números** y **operaciones matemáticas**.
  - Registrar **nuevas entradas** de datos.
  - **Guardar datos** de forma segura.
  - **Elaborar gráficos** que apoyen a la toma de decisiones.
  - **Comunicarnos con servidores** u otros sistemas computacionales.

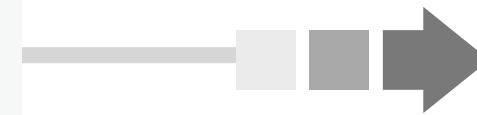
Para estas tareas hacemos uso de DIVERSOS computadores

## PROGRAMA

Conjunto de INSTRUCCIONES para ejecutar un trabajo en un LENGUAJE que sea comprensible para una computadora

# PROGRAMA VS ALGORITMO

Pensar en la manera de resolver un problema



- Pasos a Seguir
- El método a ejecutar
- Puede explicarse en español.



## Algoritmo

Conjunto de pasos para resolver un problema

## Programa

Instrucciones codificadas en un lenguaje de programación

## EJEMPLO DE ALGORITMO

Encontrar la palabra más larga en una lista de palabras

Inventamos un algoritmo SIMPLE, con los siguientes **PASOS**:

1. Iniciamos con una “palabra” ELEGIDA de longitud 0.
2. Pasamos a la SIGUIENTE PALABRA de la lista.
3. Si esta nueva palabra tiene una LONGITUD MAYOR a la ELEGIDA, la convertiremos en nuestra NUEVA ELEGIDA. De lo contrario, permanecemos con la misma.
4. REPETIMOS los pasos 2 y 3 con TODAS LAS PALABRAS de la lista.
5. Al finalizar, nuestra palabra ELEGIDA, será la de mayor longitud.

## PROGRAMA CODIFICADO

Encontrar la palabra más larga en una lista de palabras

Traducimos el algoritmo anterior para implementarlo en lenguaje Python mediante las siguientes líneas de código:

```
1 oracion = input("Ingresa palabras separadas por espacios: ")
2 palabras = oracion.split()
3 elegida = ""
4 for palabra in palabras:
5     if len(palabra) > len(elegida):
6         elegida = palabra
7 print("La palabra más larga es", guess)
```

- Las **líneas 1 y 2** obtienen los datos y los guardan en una variable llamada palabras
- La **línea 3** corresponde al **paso 1** del algoritmo
- Las **líneas 4-6** corresponden a los **pasos 2-4**.
- La **línea 7** imprime el resultado correspondiente al **paso 5**.

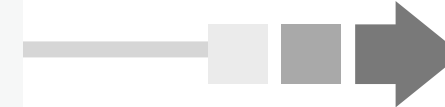
# SENTENCIAS DE PROGRAMACIÓN

Un lenguaje de programación  
es MÁS PRECISO  
que un lenguaje humano



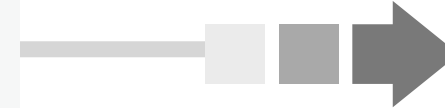
- No hay lugar para ambigüedades
- La computadora no adivina a lo que nos referimos

Cada lenguaje tiene una  
SINTAXIS ESTRUCTA



- Existen reglas para escribir instrucciones
- Sentencias de Programación o Declaraciones de Programa
- Python es más amigable, pero no es la excepción

Existe una  
Interpretación ÚNICA



- La frase for palabra in palabras tiene una única interpretación
- Cualquier computadora que hable “el lenguaje” hará lo mismo cuando vea esta línea.

No puede intercambiarse  
EL ORDEN de las palabras



- Un humano puede deducir el significado
- Una computadora no podrá ejecutarlo



# LECCIÓN 0

Applied Mathematics and Actuary Training



## LENGUAJES DE PROGRAMACIÓN - CARACTERÍSTICAS

LECCIÓN 1 – Conociendo Python

# DIFERENTES LENGUAJES

El mismo algoritmo puede  
codificarse en varios  
lenguajes de programación

- El algoritmo es abstracto – Es **diferente de la forma de comunicarlo** a la computadora
- Cada lenguaje de programación tiene **diferente sintaxis**.
- Elegir un lenguaje específico tiene **ventajas y desventajas**.

## Algoritmo

Conjunto de pasos para resolver un problema

Programa en  
C++

Programa en  
Python

Programa en  
R



# ALTO NIVEL VS BAJO NIVEL

- Más legible – Abstracto del Hardware
- Más fácil de programar
- Menos eficiente
- Menor conocimiento técnico (hardware)
- Menos líneas de código
- Posibilidad de: Automatización de Tareas, utilizar librerías y funciones.
- Menos legible – Cercano a código máquina
- Más difícil de programar
- Más eficiente
- Mayor conocimiento técnico (hardware)
- Más líneas de código
- Control sobre: Asignación de memoria, Reparto de procesador, Hardware

ALTO  
NIVEL

```
# NEURAL NETWORKS!
model = tf.keras.Sequential()
# flatten the 128x128x3 images into 1-D vectors
model.add(keras.layers.Flatten())
# add the specified hidden layers with the specified activation
for hidden_layer_size in hidden_layer_sizes:
    model.add(tf.keras.layers.Dense(units=hidden_layer_size, activation=activation))
```

Python | Visual Basic | Ruby

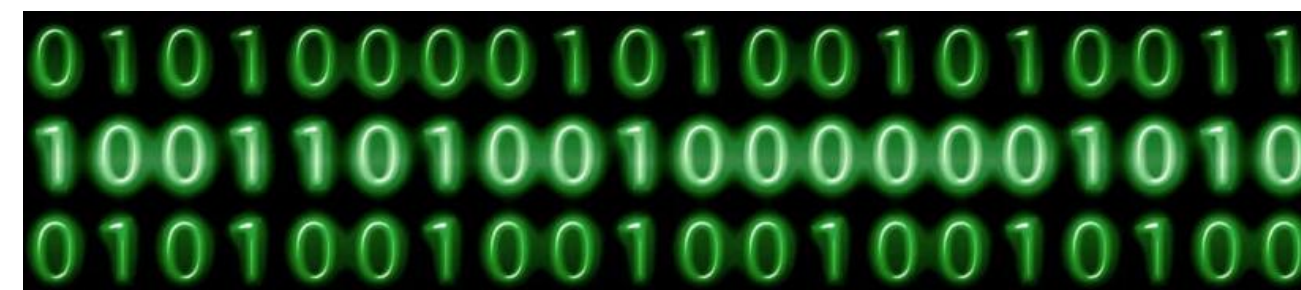
R | JavaScript

Java | Perl

C/C++

Ensamblador

Código Máquina



BAJO  
NIVEL

# PARADIGMAS

Estilos o enfoques fundamentales para escribir, estructurar y ejecutar programas.

Paradigma	Descripción breve	Lenguajes comunes
Procedimental	Basado en secuencias estructuradas de instrucciones y funciones.	C, R, <b>Python</b> , PascalJava (parcialmente)
Orientado a objetos (OOP)	Organiza el código en objetos que encapsulan datos y comportamientos.	Java, C++, <b>Python</b> , Ruby, R (parcialmente)
Funcional	Usa funciones puras, evita el estado mutable y los efectos secundarios.	Haskell, Lisp, Scala, <b>Python</b> , R
Declarativo	El programador declara únicamente <i>qué</i> quiere hacer, no <i>cómo</i> .	SQL, HTML, CSS
Reactivo	Diseñado para trabajar con flujos de datos y cambios en tiempo real.	RxJS, React (JavaScript)
Lógico	Basado en reglas lógicas y hechos; el programa responde a consultas lógicas.	Prolog
Orientado a eventos	Basado en la respuesta a eventos (clicks, teclas, señales, etc.).	JavaScript, Node.js, C#
Concurrente / Paralelo	Permite la ejecución simultánea de múltiples procesos o hilos.	Go, Rust, Java, Erlang



# LECCIÓN 0

Applied Mathematics and Actuary Training



## PYTHON ONE-O-ONE

LECCIÓN 1 – Conociendo Python





## BREVE HISTORIA DE PYTHON

- Creado por el holandés Guido van Rossum a finales de los 80.
- Primera versión pública en 1991.
- Inspirado en el lenguaje ABC.
- Diseñado para ser claro y legible.
- Dirigido hasta 2018 personalmente por Guido.
- Desde 2019 dirigido por un consejo de cinco personas (renovado anualmente).
- Administrado por la Python Software Foundation.
- La versión Python 2 cayó en obsolescencia en 2020.
- Actualmente se utiliza la versión Python 3.



# DATOS CURIOSOS

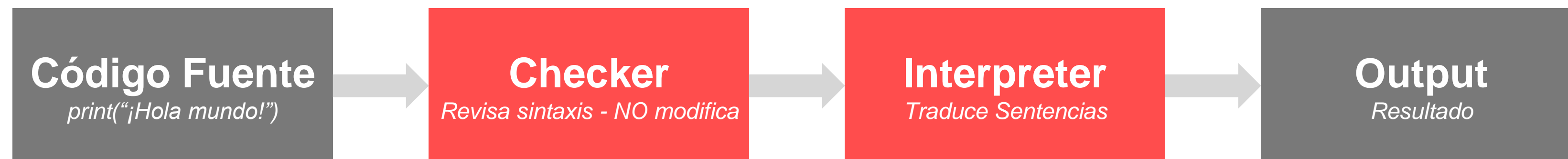
- El **logo son 2 serpientes**, basadas en la representación maya.
- El **nombre no viene del animal** sino de una serie cómica británica.
- Fue creado durante unas vacaciones de Navidad, como **proyecto personal**.
- La sintaxis es tan clara que **parece pseudocódigo**.
- Si escribes **import this** obtienes el “**Zen de Python**”.
- Actualmente es el **tercer lenguaje** de programación **más utilizado**.
- Es el lenguaje favorito para **enseñar a programar**.



# EL LENGUAJE PYTHON

- Se encuentra entre los lenguajes de más alto nivel.
- Desarrollar en Python usualmente es muy rápido.
- Su desarrollo se lleva a cabo de **manera abierta y colaborativa**.
- Multiparadigma: Soporta programación procedimental, orientada a objetos, funcional.
- Es un lenguaje interpretado, se ejecuta línea por línea.

## LENGUAJE INTERPRETADO



## LENGUAJE COMPILADO





## VENTAJAS

- Multiplataforma
- Amplia comunidad
- Extensa biblioteca estándar
- Gran ecosistema de bibliotecas externas
- Multiparadigma
- Fácil integración con otros lenguajes
- Sintaxis simple y legible

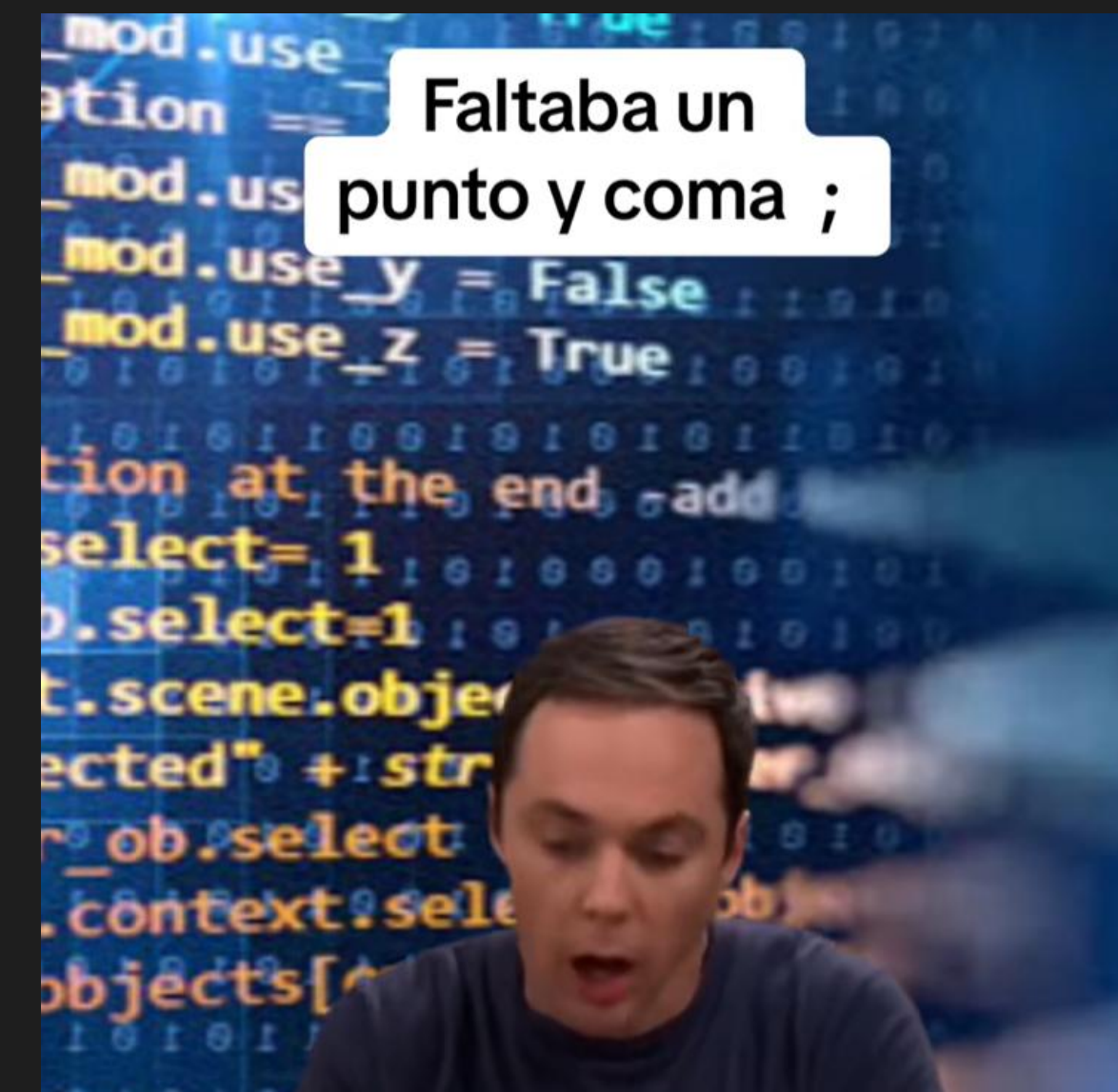


## DESVENTAJAS

- Bajo rendimiento
- Alto consumo de memoria
- Gestión de errores en tiempo de ejecución
- No ideal para aplicaciones móviles y videojuegos

Pero sobre todo...

- No se requiere punto y coma (;) al final de cada línea



# DISTRIBUCIONES DE PYTHON

Paquetes de software que contienen el intérprete de python



- Contienen la biblioteca estándar.
- Suelen contener otros paquetes y herramientas.
- Facilitan la instalación y gestión en diferentes SO.

Distribución	Ventajas	Desventajas
CPython	<ul style="list-style-type: none"><li>• Implementación estándar, escrita en C.</li><li>• Mayor compatibilidad y estandarización.</li><li>• Personalizable.</li></ul>	<ul style="list-style-type: none"><li>• Conservador en cuanto a optimizaciones.</li><li>• No incluye compilador JIT nativo.</li><li>• Conjunto básico de herramientas.</li></ul>
Anaconda	<ul style="list-style-type: none"><li>• Incorpora las librerías más utilizadas.</li><li>• Sistema de gestión de paquetes (conda).</li><li>• Integra distintas herramientas, interfaz Navigator.</li></ul>	<ul style="list-style-type: none"><li>• Su tamaño es considerablemente mayor que otros.</li><li>• Problemas con versiones de librerías.</li><li>• Espacio en disco inutilizado en versiones anteriores.</li></ul>
Active Python	<ul style="list-style-type: none"><li>• Incorpora las librerías más utilizadas.</li><li>• Sistema de gestión de paquetes más popular (pip).</li><li>• Librerías con dependencias de terceros</li></ul>	<ul style="list-style-type: none"><li>• Más lenta que otras distribuciones.</li><li>• Mayor consumo de memoria.</li><li>• Actualizaciones no frecuentes, candado en proyectos.</li></ul>
Jython	<ul style="list-style-type: none"><li>• Interoperabilidad directa con Java.</li><li>• Ejecución en JVM.</li><li>• Acceso a librerías y framewors inusuales.</li></ul>	<ul style="list-style-type: none"><li>• Solo compatible con versiones Python 2.X</li><li>• No puede acercar Python a Android.</li><li>• Curva de aprendizaje para Java.</li></ul>

# ¿QUÉ ES JUPYTER NOTEBOOK?

## IDE - Entorno de Desarrollo Integrado

- Aplicación que ayuda a desarrollar código eficientemente
- Muestra una ejecución de código a través de un navegador.
- Es el sucesor del proyecto IPython.
- Kernels de diferentes lenguajes (Python, R, Julia, Ruby, Matlab...)

# INTRODUCCIÓN A PYTHON

Conociendo Python | LECCIÓN 1