CURSOS INTERSEMESTRALES

PROTECO

Python Básico

Introcucción e Historia

Junio de 2016

Aspectos generales del curso

- Nombre del curso: Python básico
- Duración: 20 horas
- Horario: 12:30 - 16:30
- Fecha de inicio: 13 de junio
- Fecha de término: 17 junio

Instructores

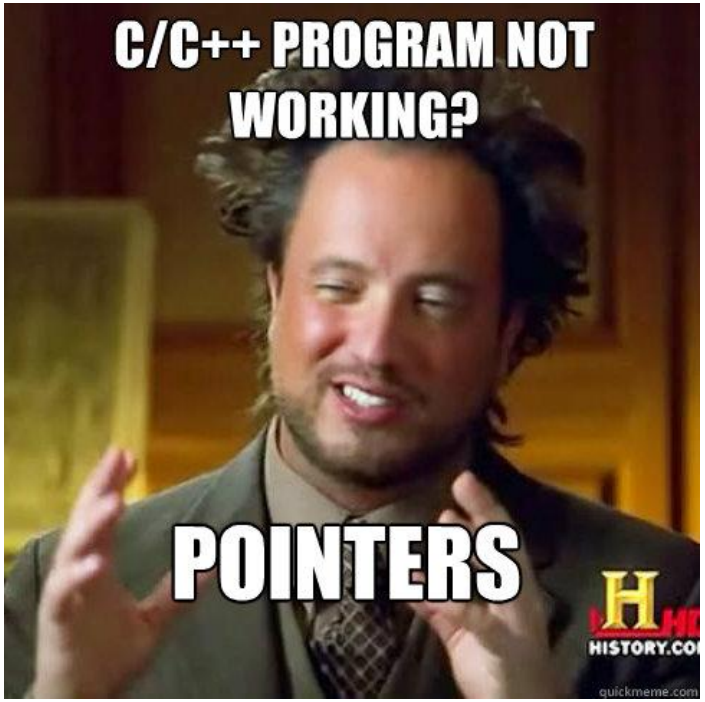
- Instructor titular
  - **Garrido Valencia Alan Rodrigo**
    - Generación 29
    - allanstone19@gmail.com
- Instructor adjunto
  - **Edgar Huerta**
    - Generación 29
    - edgar@proteco.mx
- Instructor auxiliar
  - **Valeria**
    - Generación 31
    - @valeria.proteco@gmail.com

Presentación del curso

- Por favor menciona:
  - Nombre
  - Escuela o trabajo actual
  - Algún otro lenguaje(s) que manejes
  - ¿Por qué quieres aprender Python?
  - ¿Qué esperas de este curso?

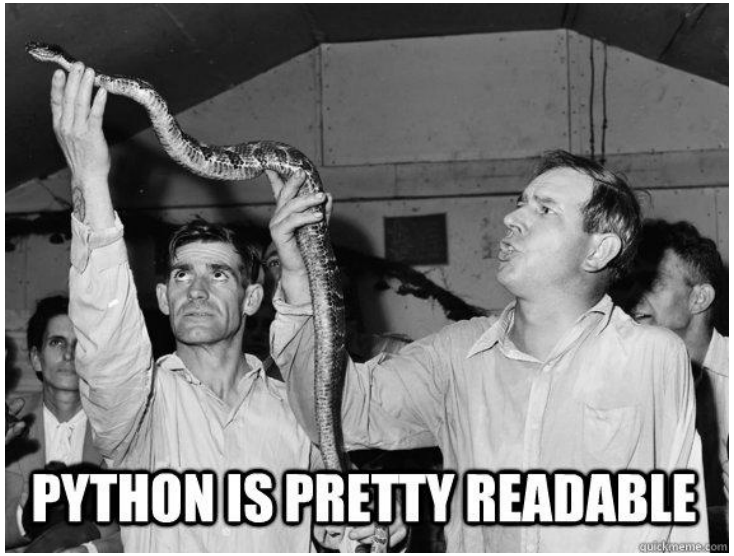
Introducción

- Python es un poderoso lenguaje de programación interpretado y fácil de aprender.



Introducción

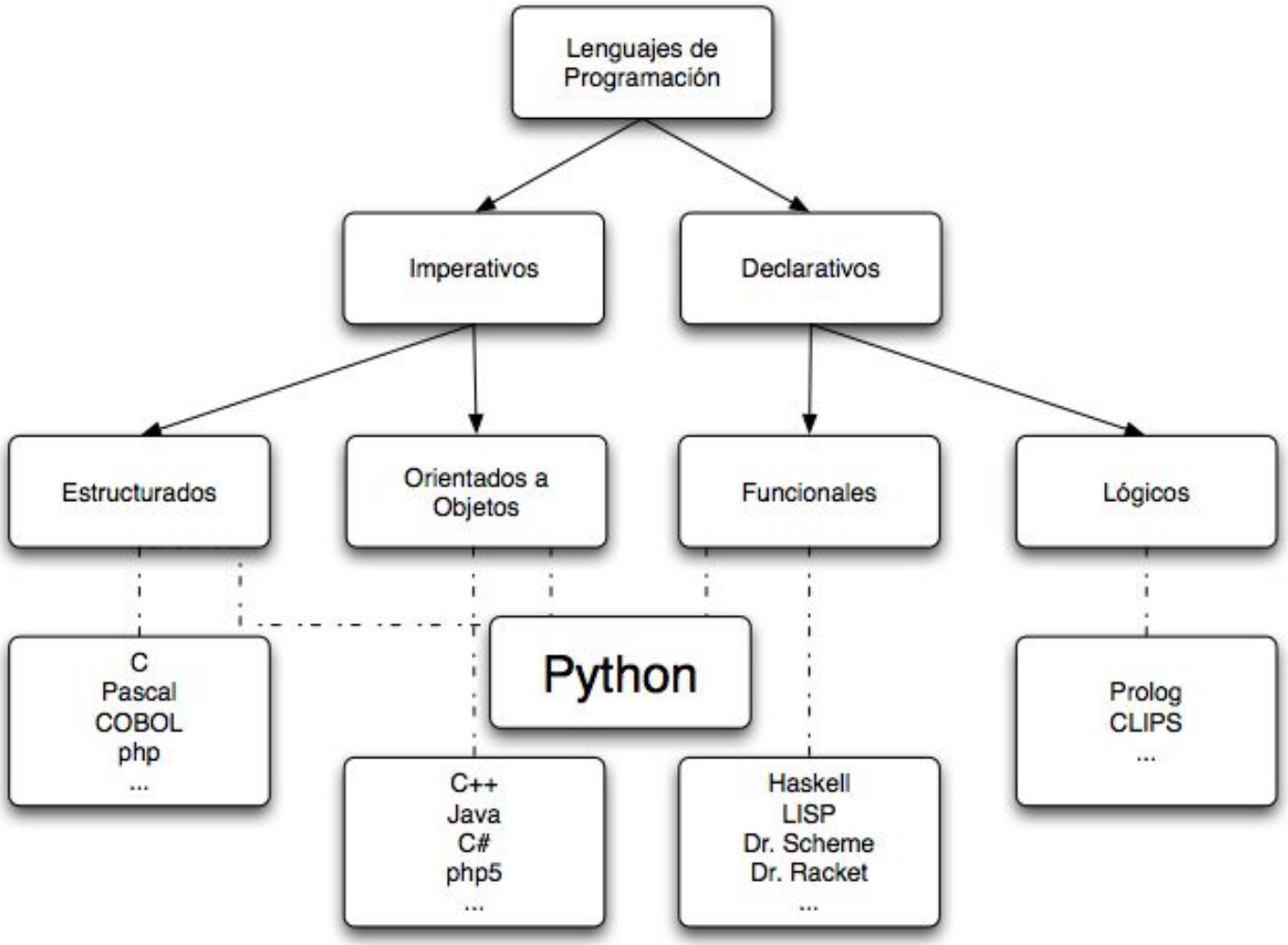
- Es un lenguaje cuya filosofía hace hincapié en una sintaxis que favorezca un código legible.



Introducción

- Es un lenguaje multiparadigma, ya que soporta orientación objetos, programación imperativa y, en menor medida, programación funcional.
- Un paradigma representa un enfoque particular o filosofía para diseñar soluciones. Los paradigmas son diferentes en la forma de abstraer los elementos involucrados en un problema.
- Generalmente lo usamos porque nos gusta o se nos facilita

Paradigmas que maneja Python

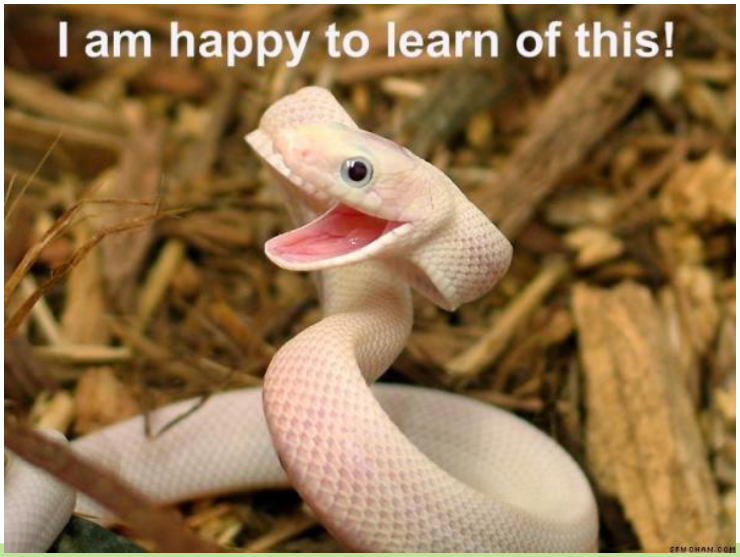


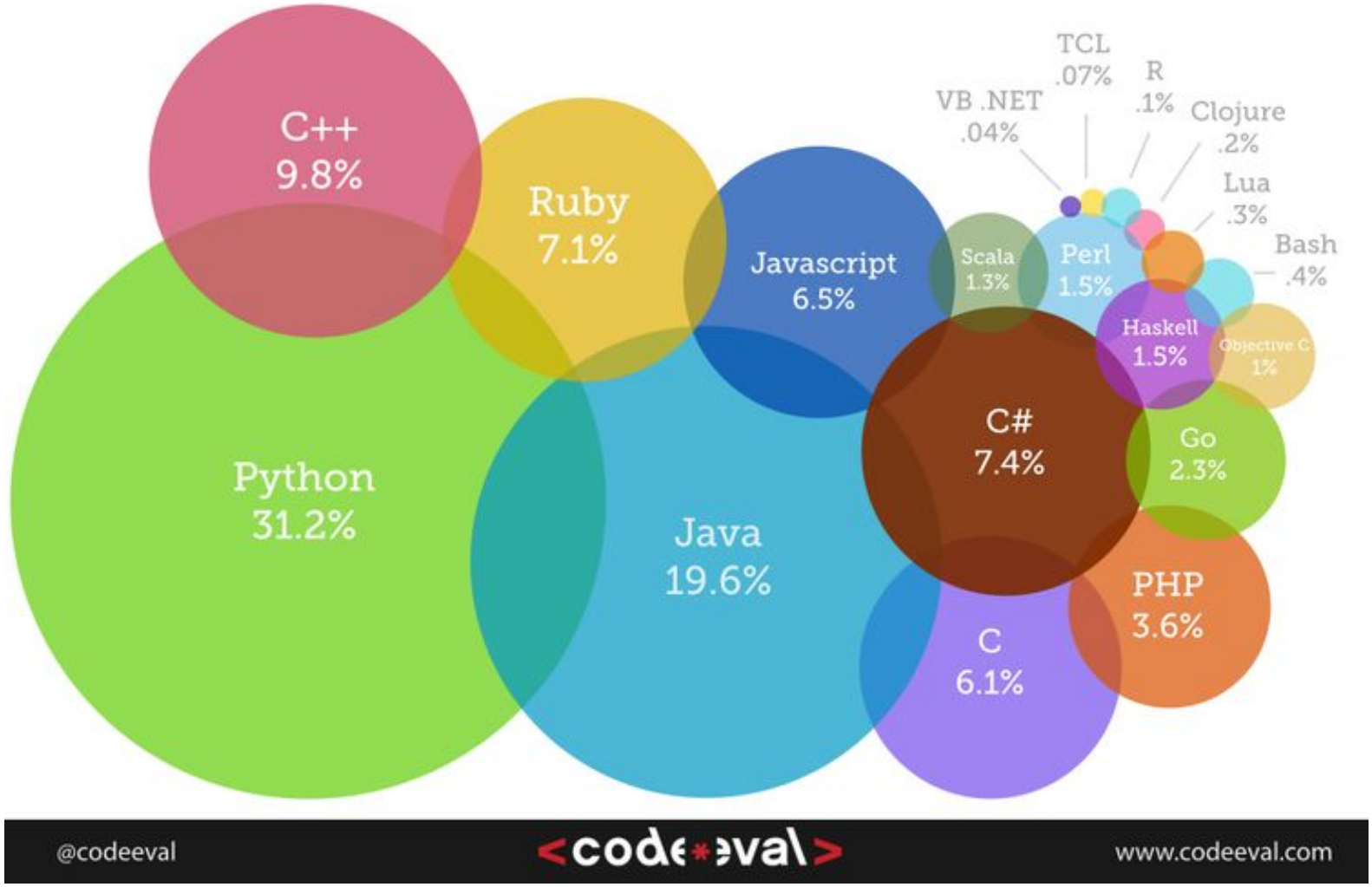
Python fue creado en 1991 por Guido van Rossum en el Centro para las Matemáticas y la Informática, como un sucesor del lenguaje de programación ABC. Guido nombró al lenguaje por su afición al humorista Monty Python.

Guido van Rossum

Características

- Ideal para scripts multiplataforma por su naturaleza interpretada, gracias a su elegante sintaxis, tipado fuerte y dinámico y facilidad de aprendizaje fue el lenguaje más usado del 2015.




Most Popular Coding Languages of 2015
C++
9.8%
Ruby
7.1%
Javascript
6.5%
TCL
.07%
VB .NET
.04%
R
.1%
Clojure
.2%
Lua
.3%
Bash
.4%
Scala
1.3%
Perl
1.5%
Haskell
1.5%
Objective C
1%
C#
7.4%
Go
2.3%
Python
31.2%
Java
19.6%
PHP
3.6%
C
6.1%
@codeeval
<code*eval>
www.codeeval.com

Ventajas

- Fácil de usar
  - Los tipos se asocian a objetos no a variables
  - Opera en un muy alto nivel de abstracción
  - Las reglas sintácticas son muy sencillas

Python es muy conveniente para el desarrollo rápido de aplicaciones.

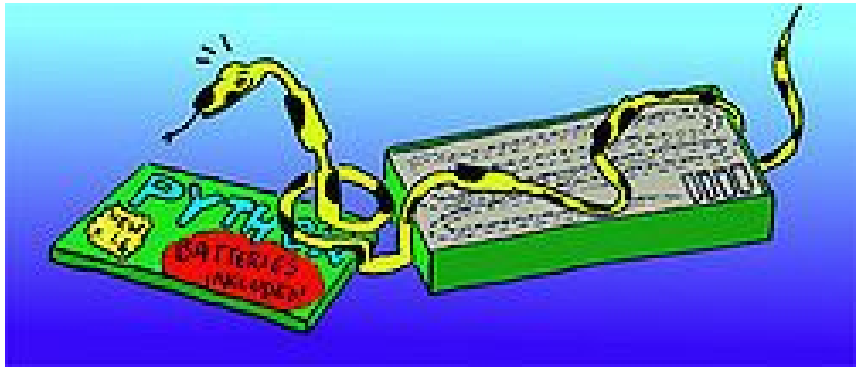
Ventajas

- Expresividad
  - Python es un lenguaje muy expresivo

    - Una línea en Python puede hacer más que una línea en cualquier otro lenguaje.
    - Menor líneas de código -> Menor tiempo en escribir y ejecutar.
    - Menor líneas de código -> Mayor facilidad para mantener y depurar los programas.

Ventajas

- Legibilidad
  - Facilidad de lectura
- Filosofía de Baterías incluidas
  - La librería estándar de Python es muy amplia



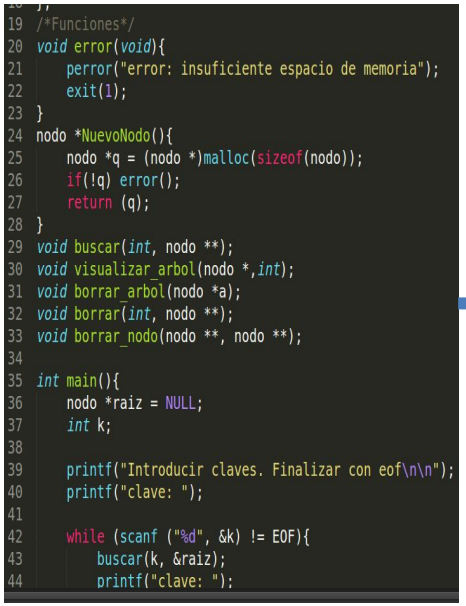
Ventajas

- Multiplataforma
  - Dado que es interpretado, el mismo código puede ejecutarse en cualquier plataforma que tenga instalado un intérprete de Python.

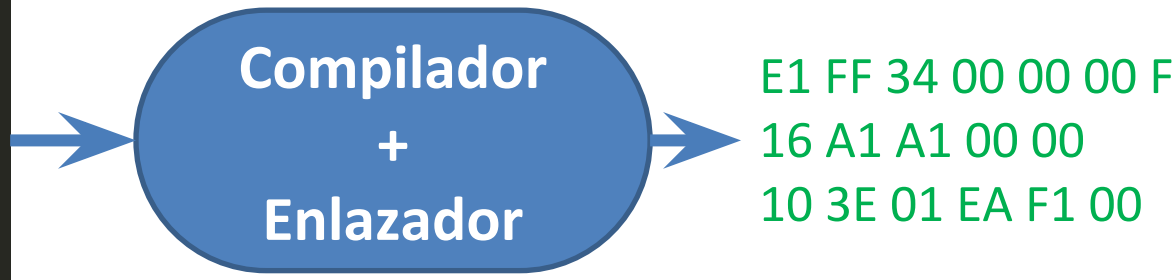
- Open Source
  - Está desarrollado con el modelo open source y está disponible libremente bajo una licencia pública general de GNU .

¿Cómo funciona Python?

- **Compiladores**



```
/*Funciones*/
void error(void){
    perror("error: insuficiente espacio de memoria");
    exit(1);
}
nodo *NuevoNodo(){
    nodo *q = (nodo *)malloc(sizeof(nodo));
    if(!q) error();
    return (q);
}
void buscar(int, nodo **);
void visualizar_arbol(nodo *,int);
void borrar_arbol(nodo *a);
void borrar(int, nodo **);
void borrar_nodo(nodo **, nodo **);

int main(){
    nodo *raiz = NULL;
    int k;

    printf("Introducir claves. Finalizar con eof\n\n");
    printf("clave: ");

    while (scanf ("%d", &k) != EOF){
        buscar(k, &raiz);
        printf("clave: ");
```

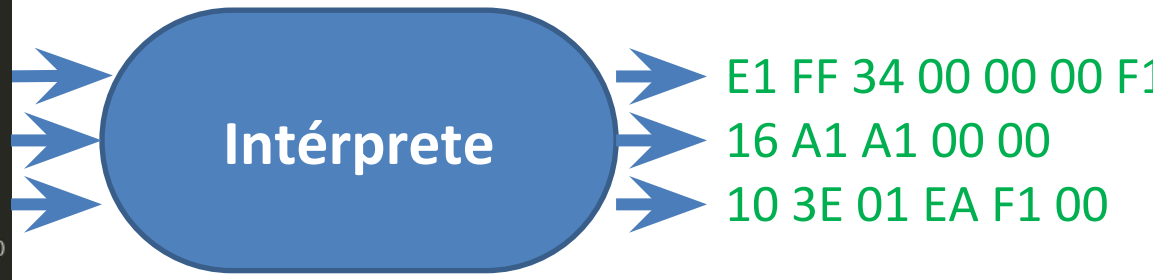


Código completo se analiza y se reportan todos los errores y advertencias. El compilador genera el código máquina mientras que el enlazador se encarga de poner las funciones externas para generar un ejecutable específico para ese sistema.

¿Cómo funciona Python?

- **Intérpretes**

```
class Animal():
    def __init__(self,especie,tipo,nombre):
        self.especie=especie
        self._tipo=tipo
        self.__nombre=nombre

    def setEspecie(self,especie):
        self.especie=especie

    def getEspecie(self):
        return self.especie

    def setTipo(self,tipo):
        self.tipo=tipo

    def getTipo(self):
        return self.tipo



    def reproducir(self):
        print(self._nombre, "se esta reproduciendo")


    def moverse(self):
        if self._tipo=="acuatico":
            print("Yo nado")
        elif self._tipo=="terreste":
            print("Yo camino o me arrastro")
```

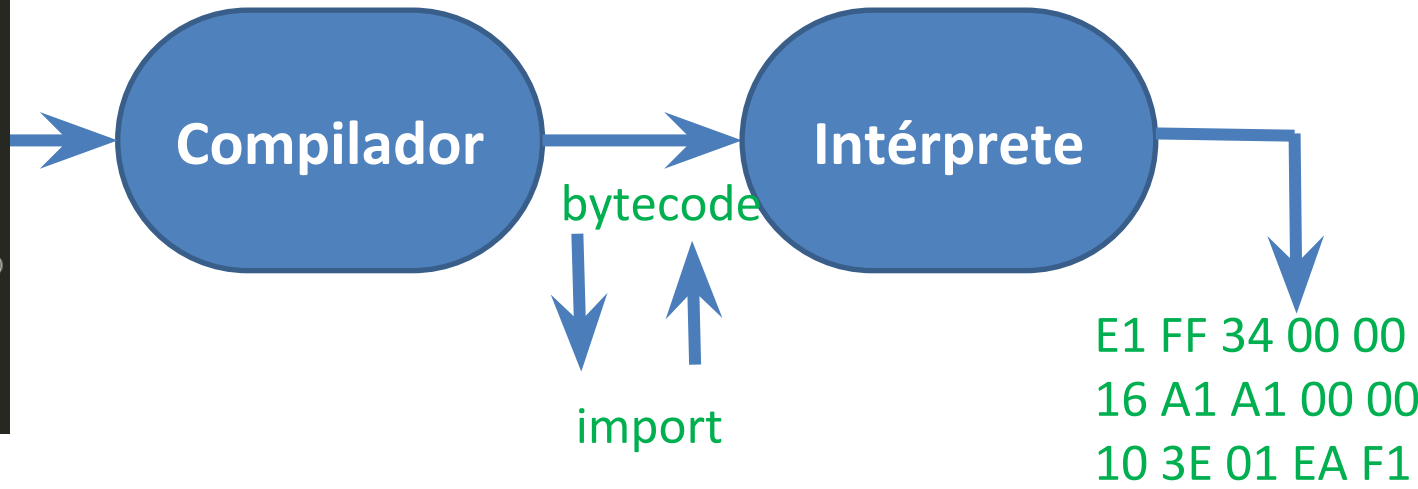


Código se analiza y ejecuta línea por línea, por lo regular compila a un lenguaje intermedio (bytecode) y después lo traduce al código máquina. Si alguna línea falla no interrumpe la operación y sigue ejecutando las demás.

¿Cómo funciona Python?

- **Intérprete de Python**

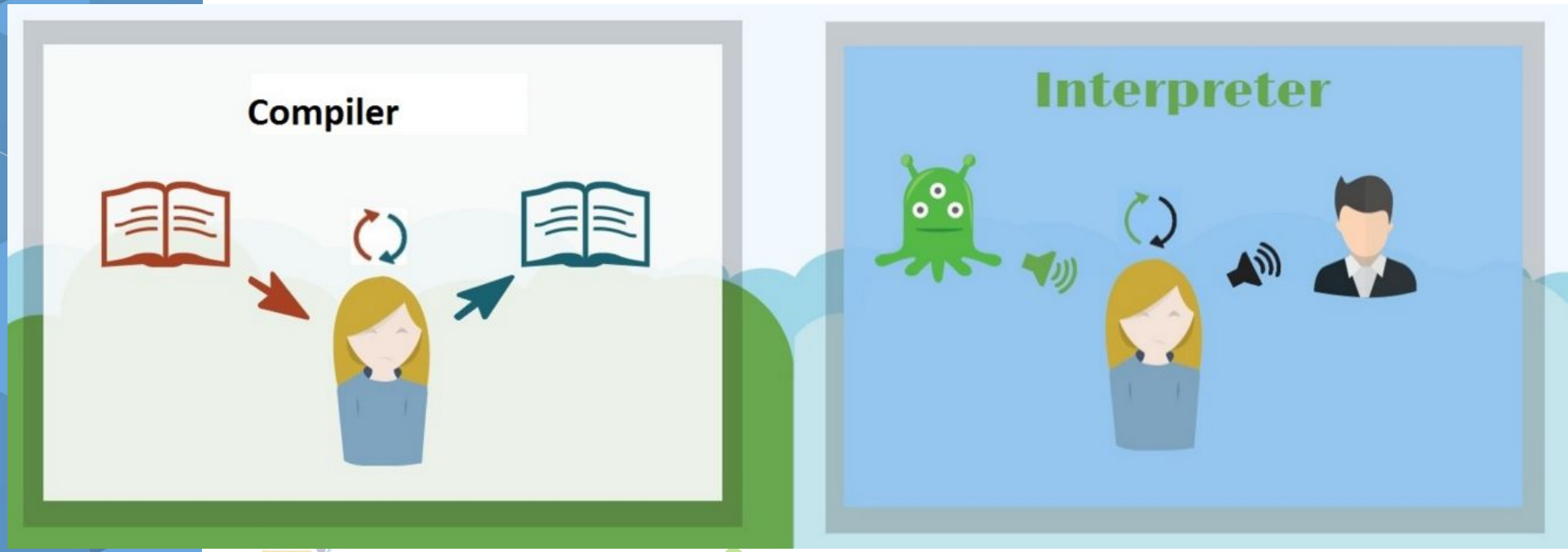
```
class Animal():
    def __init__(self,especie,tipo,nombre):
        self.especie=especie
        self._tipo=tipo
        self.__nombre=nombre

    def setEspecie(self,especie):
        self.especie=especie

    def getEspecie(self):
        return self.especie

    def setTipo(self,tipo):
        self.tipo=tipo

    def getTipo(self):
        return self.tipo


    def reproducir(self):
        print(self._nombre, "se esta reproduciendo")


    def moverse(self):
        if self._tipo=="acuatico":
            print("Yo nado")
        elif self._tipo=="terreste":
            print("Yo camino o me arrastro")
```



Un programa en python se pasa a bytecode primero, si hay alguna referencia que no esté definida el compilador lo indicará, luego la máquina virtual de ese sistema ejecuta el bytecode con las dependencias ya resueltas directamente en código máquina

- **En resumen**

Un compilador traduce el código completo y si no está bien escrito no puede completar la traducción, en cambio un intérprete sólo traduce lo que entiende y se va de corrido



Desventajas

- No es el lenguaje más rápido
- No posee las librerías más extensas
- No tiene revisión de tipos

¿Quién usa Python?

¿Quién usa Python?

- http://www.python.org/about/apps
- http://www.python.org/about/success