# ACTIVIDAD 1: PYTHON

Aprendiz: José Diaz

1. **¿QUÉ ES PYTHON**

Python es un lenguaje de programación de alto nivel, orientado a objetos, con una semántica dinámica integrada, principalmente para el desarrollo web y de aplicaciones informáticas.

Es muy atractivo en el campo del Desarrollo Rápido de Aplicaciones (RAD) porque ofrece tipificación dinámica y opciones de encuadernación dinámicas.

Python es relativamente simple, por lo que es fácil de aprender, ya que requiere una sintaxis única que se centra en la legibilidad. Los desarrolladores pueden leer y traducir el código Python mucho más fácilmente que otros lenguajes.

Por tanto, esto reduce el costo de mantenimiento y de desarrollo del programa porque permite que los equipos trabajen en colaboración sin barreras significativas de lenguaje y experimentación.

1. **HISTORIA DE PYTHON**

1989, Guido Van Rossum, quien trabajaba en el CWI (un centro de investigación holandés), decidió empezar un proyecto como pasatiempo dándole continuidad a ABC, un lenguaje de programación que se desarrolló en el CWI.

ABC fue desarrollado a principios de los 80s como alternativa a BASIC, fue pensado para principiantes por su facilidad de aprendizaje y uso. Su código era compacto pero legible.

El proyecto no trascendió ya que el hardware disponible en la época hacía difícil su uso. Así que Van Rossum le dió una segunda vida creando Python.

A Guido Van Rossum le gustaba mucho el grupo Monty Python, por esta razón escogió el nombre del lenguaje. Actualmente Van Rossum sigue ejerciendo el rol central decidiendo la dirección de Python.

En 1991, Van Rossum publicó el código de la versión 0.9.0 en alt.sources. En esta versión ya teníamos disponibles clases con herencias, manejo de excepciones, funciones y los tipos modulares.

En esta versión aparece un sistema de módulos adoptado de Modula-3, un lenguaje de programación estructurado y modular, el cual Guido describe como una de las mayores unidades de programación de Python. Por ejemplo, el modelo de excepciones de Python es parecido al de Modula-3

Para 1994 se creó comp.lang.python, un foro de discusión de Python que marcó un hito en su popularidad y multiplicó su cantidad de usuarios.

1. **CARACTERISTICAS DE PYTHON**

* Programación Orientada a objetos (POO):

Al igual que otros lenguajes populares como Java, C++ o JavaScript, Python es un lenguaje orientado a objetos. Un lenguaje orientado a objetos es aquel en el que el código se organiza en unidades denominadas clases y objetos. Esto permite representar conceptos cotidianos en un programa. En definitiva, a través de Python podemos expresarnos de forma similar a como lo haríamos en la vida real.

* Lenguaje interpretado:

Existen dos tipos de lenguajes: compilados o interpretados como en el caso de Python. No es necesario compilar (transformar el lenguaje de los programas informáticos a uno equivalente) cuando se trabaja con Python, ya que los intérpretes que se utilizan con este lenguaje se encargan de ejecutar dichos programas a través de scripts propios.

* Multiplataforma:

Python está disponible en los principales sistemas operativos como Linux, Windows, UNIX, Mac OS, etc. Es decir, este lenguaje puede ser ejecutado en casi cualquier sistema operativo siempre que se cuente con un intérprete adecuado para ello.

* Tipado dinámico:

El tipado dinámico de Python permite que las variables no tengan que especificar su tipo. Estas adoptan un tipo automáticamente en función del valor que se les asigne mientras el lenguaje esté funcionando.

* Lenguaje open source:

Python es un lenguaje de código abierto, por lo que no se requieren licencias de pago para comenzar a trabajar con él.

* Ampliamente respaldado:

Sus características y funcionalidades hacen que este lenguaje sea muy interesante. Por ello, Python ha generado una comunidad de usuarios muy grande a su alrededor que puede ser de utilidad cuando queremos encontrar información o pedir ayuda para desarrollar cualquier tipo de programa o algoritmo.

* Es polivalente:

Como ya hemos comentado, Python se utiliza para infinidad de proyectos y aplicaciones diferentes. Actualmente destacan los siguientes:

* Aprendizaje automático (Machine Learning)
* Inteligencia Artificial (IA)
* Big data y Análisis de datos
* Operaciones matemáticas
* Visualización de datos
* Programación de apps
* Desarrollo web
* Desarrollo de videojuegos
* Gestión financiera

1. **VENTAJAS Y DESVENTAJAS DE PYTHON**

## Ventajas de Python

* Una gran comunidad para ayudarte:

Python está sustentado en una gran comunidad de usuarios que explora sus posibilidades frecuentemente. Esta comunidad tan activa permite que usuarios de todos los niveles encuentren los mejores tutoriales, consejos y claves para empezar a utilizarlo.

Es decir, hablamos de un lenguaje accesible para personas con cualquier nivel de conocimientos. Además, es un software gratuito y de código abierto.

* Sintaxis sencilla:

El lenguaje de codificación de Python tiene una sintaxis muy sencilla y utiliza palabras en inglés.

* Legibilidad:

Al ser un lenguaje de alto nivel, las líneas de código escritas en Python también son fáciles de leer e interpretar para rápidamente entender el objeto del fragmento de código analizado.

* Escalabilidad:

Puedes empezar un programa en Python sin tener que preocuparte por la difícil tarea de reescribir o adaptar el código a otras plataformas a medida que vas creciendo. Es decir, este lenguaje te permitirá aumentar la complejidad de tus programas a lo largo del tiempo.

* Aplicaciones en Big Data:

El último punto fuerte que destacamos de Python es la gran utilidad para el mundo del Big Data, del Data Science y de la Inteligencia Artificial. Además, este lenguaje posee librerías muy potentes de cálculo matricial y analítica de datos como las mencionadas anteriormente: Pandas, Numpy, Scikit Learn o Keras entre otras.

## Desventajas de Python

1. Lentitud

La lentitud de Python se debe principalmente a su naturaleza dinámica y versatilidad. No obstante, hay formas de optimizar las aplicaciones de Python aprovechando la sincronización, entendiendo las herramientas de creación de perfiles y considerando el uso de múltiples intérpretes.

2. Consumo de memoria

Otra de las desventajas de Python es que, en el caso de que una tarea requiera mucha memoria, Python no es la mejor opción. El consumo de memoria de Python es muy alto, y esto se debe a la flexibilidad de los tipos de datos.

3. No es perfecto para todo

El hecho de que sea un lenguaje bastante versátil, no quiere decir que sea eficiente en todos los campos en los que se puede ejecutar. En el caso del desarrollo de aplicaciones móviles, Python no es la mejor opción para ello, ya que existen otros lenguajes más especializados y que ofrecen mejores resultados.

1. **¿COMO DEFINIR VARIABLES Y CONSTANTES EN PYTHON?**

## Definición de variables

Las variables en Python se crean cuando se definen por primera vez, es decir, cuando se les asigna un valor por primera vez. Para asignar un valor a una variable se utiliza el operador de igualdad (=). A la izquierda de la igualdad se escribe el nombre de la variable y a la derecha el valor que se quiere dar a la variable.

En el ejemplo siguiente se almacena el número decimal 2.5 en una variable de nombre x (como se comenta en el apartado anterior, realmente habría que decir que se crea la etiqueta x para hacer referencia al objeto número decimal 2.5). Fíjese en que los números decimales se escriben con punto (.) y no con coma (,).

>>> x = 2.5

La variable se escribe siempre a la izquierda de la igualdad. Si se escribe al revés, Python genera un mensaje de error:

>>> 2.5 = x

SyntaxError: can't assign to literal

Para que IDLE muestre el valor de una variable, basta con escribir su nombre:

>>> x = 2.5

>>> x

2.5

Si una variable no se ha definido previamente, escribir su nombre genera un mensaje de error:

>>> x = -10

>>> y

Traceback (most recent call last):

File "<pyshell#1>", line 1, in <module>

y

NameError: name 'y' is not defined

Una variable puede almacenar números, texto o estructuras más complicadas (que se verán más adelante). Si se va a almacenar texto, el texto debe escribirse entre comillas simples (') o dobles ("), que son equivalentes. A las variables que almacenan texto se les suele llamar cadenas (de texto).

>>> nombre = "Pepito Conejo"

>>> nombre

'Pepito Conejo'

Si no se escriben comillas, Python supone que estamos haciendo referencia a otra variable (que, si no está definida, genera un mensaje de error):

>>> nombre = Pepe

Traceback (most recent call last):

File "<pyshell#0>", line 1, in <module>

nombre = Pepe

NameError: name 'Pepe' is not defined

>>> nombre = Pepito Conejo

SyntaxError: invalid syntax

## Definición de constantes

El uso de las constantes no está directamente ligado a Python, pues hay constantes en todos los lenguajes de programación. Las constantes son valores que no cambian a través del tiempo en la ejecución de tu sistema pero que necesitamos conocer. Además, no es necesario utilizar las constantes forsozamente en el desarrollo de cualquier software, esto quiere decir que puedes utilizar las variables comunes aunque tu sepas que ese valor no cambiara.

Algunos ejemplos de uso de constantes normalmente utilizadas por los desarrolladores:

RUTA\_API\_SERVIDOR = "https://grupocodesi.com/rest"

CORREO\_ADMIN = "correo@sitio.com"

DIRECCION = "Metro Puebla!

CURSOS = "Cursos de python"

COSTO = 2900

Como puedes darte cuenta el uso de las constantes son valores que son siempre fijos por ejemplo en la ruta del api del servidor es casi imposible que se cambie dicha ruta y podra consumir ese servicio web, esto no quiere decir que no se podra cambiar, probablemente en una nueva version del servicio para mejorarlo podría cambiar a lo largo del tiempo

En una constante su valor o contenido no cambia, pero en Python por su intento extremo de simplicidad si permite que el programador cambie el valor de las constantes, la única diferencia es que las constantes van en mayúscula por ejemplo CONSTANTE1, desde el punto de vista funcional no existe ninguna diferencia en Python entre las constantes y variables, pero se recomienda que no se intente modifica el valor de una constante posteriormente, solo como buena práctica.