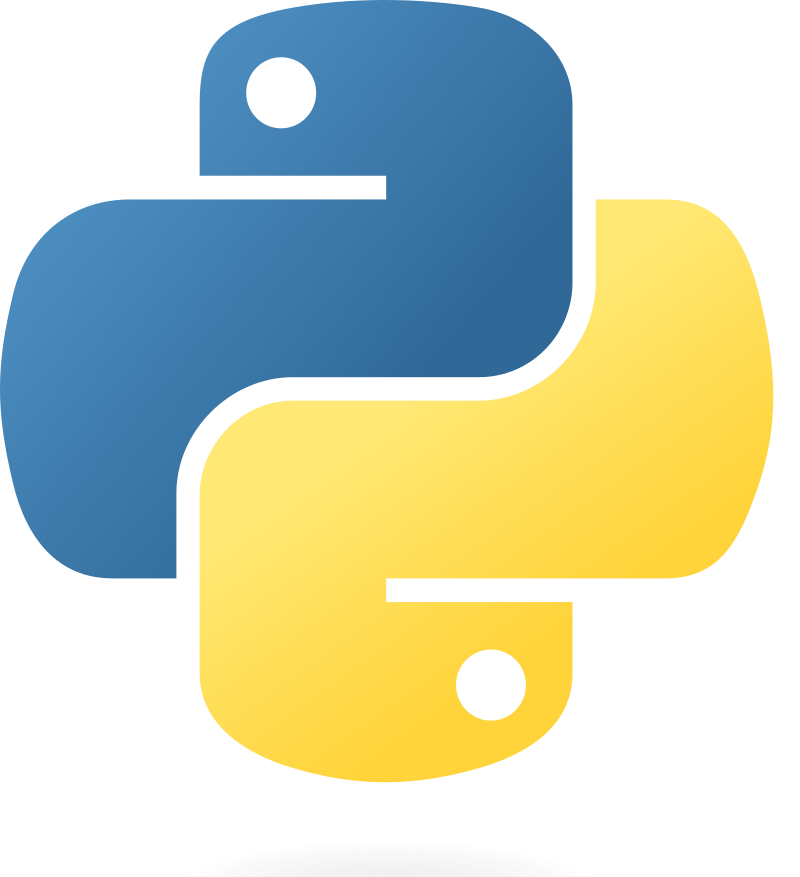
**ACTIVIDAD #1 DE PYTHON:**

Fundamentos de python

1.Qué es python

R/

En términos técnicos, 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.

Además, soporta el uso de módulos y paquetes, lo que significa que los programas pueden ser diseñados en un estilo modular y el código puede ser reutilizado en varios proyectos. Una vez se ha desarrollado un módulo o paquete, se puede escalar para su uso en otros proyectos, y es fácil de importar o exportar.

Por otro lado, uno de los **beneficios más importantes de Python** es que tanto la **librería estándar** como el **intérprete** están disponibles gratuitamente, tanto en forma binaria como en forma de fuente.

Tampoco hay exclusividad, ya que Python y todas las herramientas necesarias están disponibles en todas las plataformas principales. Por lo tanto, es una opción **multiplataforma**, bastante tentadora para los desarrolladores que no quieren preocuparse por pagar altos costos de desarrollo.

En definitiva, es un lenguaje de programación relativamente fácil de aprender, y las herramientas necesarias están disponibles para todos de forma gratuita. Esto hace que sea accesible para casi todo el mundo. Si dispones de tiempo para aprender, conseguirás crear esos proyectos que tienes en mente.

2.Historia de python(creador, año, versiones)

R/-

Python fue creado por Guido van Rossum, un programador holandés a finales de los 80 y principio de los 90 cuando se encontraba trabajando en el sistema operativo Amoeba. Primariamente se concibe para manejar excepciones y tener interfaces con Amoeba como sucesor del lenguaje ABC.

VERSIONES DE PYTHON

* Comienzo de la implementación - December, 1989
* Publicación interna en CWI - 1990
* Python 0.9.0 - 20 de febrero de 1991 (publicado en alt.sources)[15](https://es.wikipedia.org/wiki/Historia_de_Python#cite_note-history-15)​
  + Python 0.9.1 - febrero de 1991
  + Python 0.9.2 - otoño de 1991
  + Python 0.9.4 - 24 de diciembre de 1991
  + Python 0.9.5 - 2 de enero de 1992 (solo para Macintosh)
  + Python 0.9.6 - 6 de abril de 1992
  + Python 0.9.7 beta - 1992
  + Python 0.9.8 - 9 de enero de 1993
  + Python 0.9.9 - 29 de julio de 1993
* Python 1.0 - enero de 1994
  + Python 1.5 - 31 de diciembre de 1997
  + Python 1.6 - 5 de septiembre de 2000
* Python 2.0 - 16 de octubre de 2000
  + Python 2.1 - 17 de abril de 2001
  + Python 2.2 - 21 de diciembre de 2001
  + Python 2.3 - 29 de julio de 2003
  + Python 2.4 - 30 de noviembre de 2004
  + Python 2.5 - 19 de septiembre de 2006
  + Python 2.6 - 1 de octubre de 2008
  + Python 2.7 - 3 de julio de 2010
* Python 3.0 - 3 de diciembre de 2008
  + Python 3.1 - 27 de junio de 2009
  + Python 3.2 - 20 de febrero de 2011
  + Python 3.3 - 29 de septiembre de 2012
  + Python 3.4 - 16 de marzo de 2014
  + Python 3.5 - 8 de febrero de 2015
  + Python 3.6 - 23 de diciembre de 2016
  + Python 3.7 - 12 de junio de 2018
  + Python 3.8 - 14 de octubre de 2019
  + Python 3.9 - 5 de octubre de 2020

3.Características de python

R/

Algunos de los mayores cambios anunciados para Python 3.0 fueron:

* Cambiar la instrucción (print ) de modo que sea una función "built-in", no una instrucción. Esto ha hecho que sea más fácil cambiar un módulo para usar una función print diferente, así como hacer la sintaxis más regular. En Python 2.6 y 2.7 esta característica puede ser habilitada usando from \_\_future\_\_ import print\_function.
* Moviendo reduce (pero no map o filter) fuera del espacio de nombres "built-in" dentro de "functools".
* Adicionar soporte para anotación opcional en funciones que pueda ser usar para la declaración informal de tipos u otros propósitos.
* Unificar los tipos str/unicode, representando texto, e introduciendo un tipo immutable bytes; y un más correspondiente tipo mutable bytearray, ambos de los cuales representan "arrays" de [bytes](https://es.wikipedia.org/wiki/Byte);
* Eliminar [compatibilidad hacia atrás](https://es.wikipedia.org/wiki/Compatibilidad_hacia_atr%C3%A1s), incluidas clases "old-style", excepciones "string", e implícitos "imports" relativos.

Las subsecuentes publicaciones de la serie Python 3.x han incluido sustanciales nuevas características; todas al corriente del desarrollo que el lenguaje ha tenido en la serie 3.x.

4.Ventajas y desventajas de python

R/

VENTAJAS:

Lenguaje de alto nivel

Polivalente y de paradigmas

Bibliotecas y *frameworks*

Portabilidad

Gratis y de código abierto

Baja curva de aprendizaje

Comunidad fuerte

DESVENTAJAS:

Lentitud

Consumo de memoria

Desarrollo móvil

5.Cómo se definen las variables y los tipos de variables

R/

Son los componentes básicos para tratar todo tipo de datos o variables y contienen valores de datos puros y simples.

Python tiene **cuatro tipos primitivos:** enteros, flotantes, booleanos y cadenas o Strings.

* Los tipos de variables enteros en Python (int) se utilizan para representar datos numéricos, específicamente números enteros. Estos pueden ser tanto positivos como negativos.

year = 2021

dia = 7

edad = 26

temperatura = -5

angulo = -45

* Los tipos de datos flotantes en Python (float) se utilizan para representar números de coma flotante o con decimales.

pi = 3.1416

estatura = 1.84

peso = 85.6

temperatura = -5.55

edad = 26.0

* Los tipos primitivos booleanos en python (bool) son tipos de datos binarios, es decir que pueden tomar los valores: *Verdadero* (**True**) y *Falso* (**False**). Son útiles para expresiones con condicionales y de comparaciones.

esta\_frio = **True**

es\_bajo = **False**

* Los tipos de datos String en Python son un arreglo de caracteres que forman cadenas para formar un mensaje o oración generalmente. Se pueden crear usando comillas simples, dobles o triples.

profesor = "Sergio Castaño Giraldo"

web = 'Control Automático Educación'

cursos = **"""**

**1. Python**

**2. Matlab**

**3. MicroPython**

**4. Arduino**

**5. PIC**

**6. Control de Procesos**

**"""**