GOBIERNO DE
EL SALVADOR

SECRETARÍA DE
INNOVACIÓN

CIENCIA DE DATOS

Agenda
Sesión 2/18

Python para ciencia de datos

- ¿Qué es Python?
- ¿Porqué Python para la Ciencia de Datos?
- Instalación de Visual Studio Code
- Instalación de Python 3.9.6
- Recorrido básico de Python.
    - Variables, tipos de datos
    - Operaciones entre variables
    - Estructuras de decisión
    - Estructuras de repetitivas
1. - Estructuras de datos

¿Qué es Python?

Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código. Se trata de un lenguaje de programación multiparadigma, ya que soporta parcialmente la orientación a objetos, programación imperativa y, en menor medida, programación funcional.

Es un lenguaje interpretado, dinámico y multiplataforma.

¿Qué es Python?

Python es un lenguaje de programación de alto nivel, con esta herramienta o lenguaje de programación podemos crear diferentes tipos de aplicaciones y utilizar par diferentes tipos de necesidad.

Python es uno de los lenguajes más populares que fue creado por Guido van Russum y se lanzó 1991.

¿Qué es Python?

El lenguaje de programación de Python puedes crear diferentes tipos de aplicaciones o puedes utilizar en diferentes trabajos o tareas.

¿Qué es Python?

- Desarrollo Web (lado de servidor **django**)
- Análisis de datos
- Aprendizajes automáticos (machine learning)
- Administración de sistemas.
- Testear software o escribir test automatizados
- Prototipos de software
- Desarrollo de escritorio
- Programación de redes
- Desarrollo de juegos (**Pygame**)
- Desarrollo móvil (**Kivy**)
- Otros

Sintaxis de Python

Es muy simple fui diseñado para la legibilidad a comparación con otros lenguajes de programación, Python en su codificación es muy limpio no usa punto y comas para terminar una línea de código, sino que emplea una nueva línea.

Python no usa llaves para indicar el ámbito que pertenece por ejemplo una función a una clase o en el momento de trabajar con las condicionales o los bucles, simplemente utiliza dos puntos Y para indicar el ámbito que pertenece ya sea una condición un bucle utiliza simplemente la sangría en blanco.

Ejemplo de Código de Python

```
#Ejemplo de Variables
x = 1
y = 2

if z > 2:
        print("Five is greater than two!")


#Ejemplo de captura de datos
nombre = input("Escriba su nombre: ")
print("Bienvenido " + nombre)
```

¿Porqué Python para la Ciencia de Datos?

Fácil de utilizar

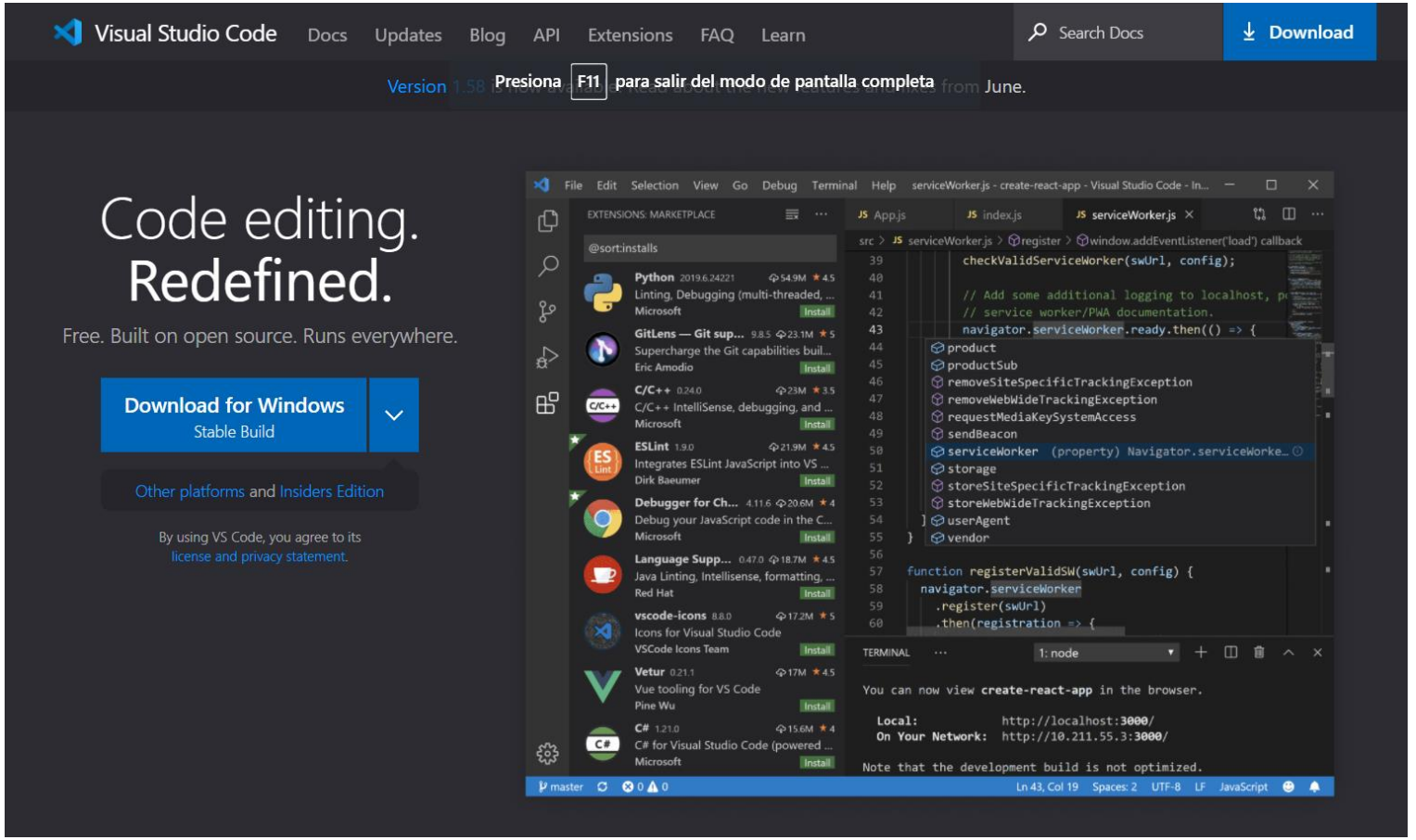
Fácil de entender

Fácil de enseñar

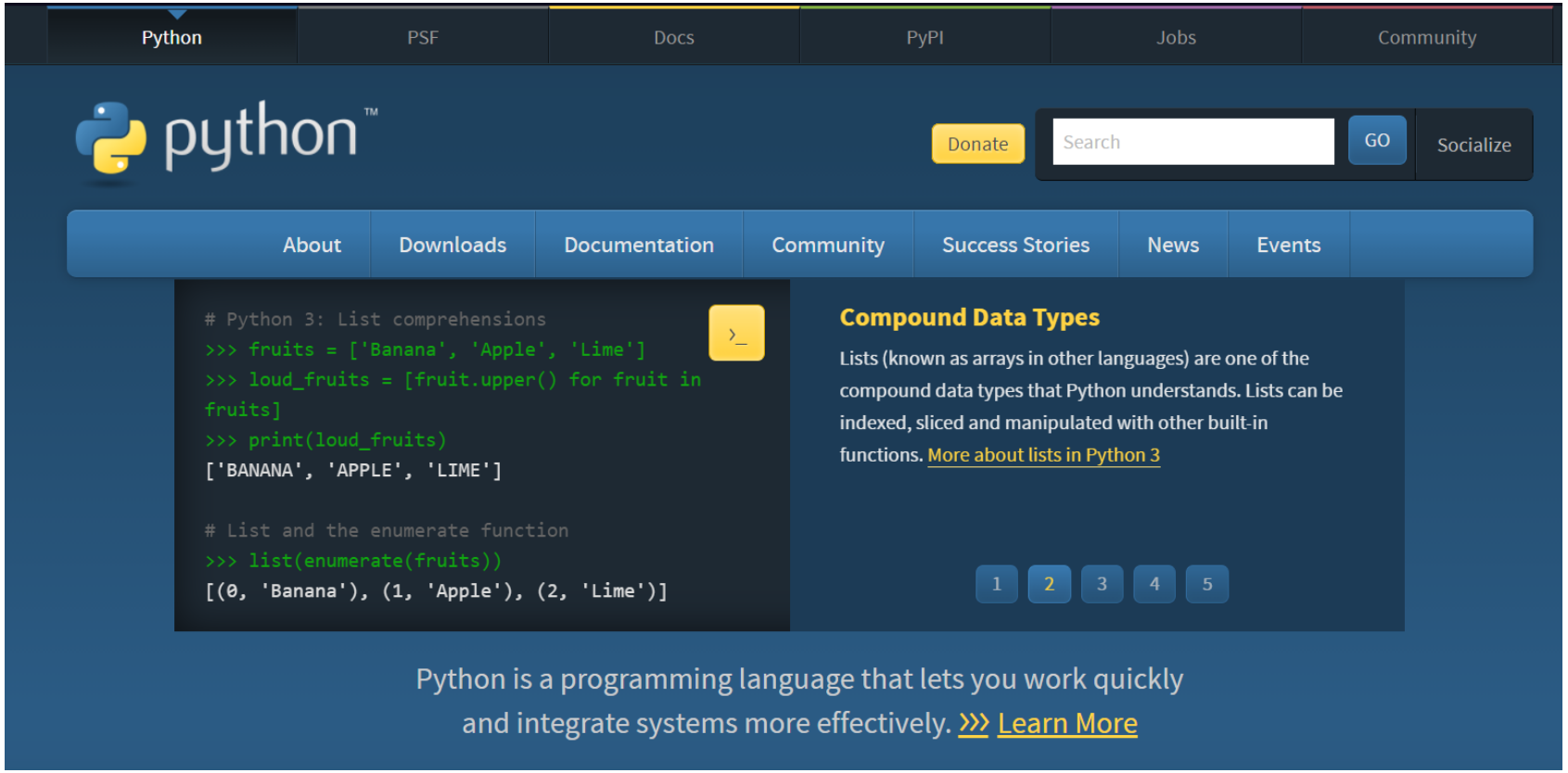
Fácil de aprender

Fácil de obtener, instalar y desplegar

Instalación de Visual Studio Code y Python



Enlace para descarga:
https://code.visualstudio.com



Enlace para descarga:
https://www.python.org

Recorrido básico de Python

Variables de Python

En Python, las variables se crean cuando le asigna un valor:

```
x = 5
y = "Hello, World!"
```

Tambien podemos definir como queremos que sean estas variables, por ejemplo con los numeros:

```
x = str(3)
y = int(3)
z = float(3)
```

```
print(x)
print(y)
print(z)
```

Obtenga el tipo

Puede obtener el tipo de datos de una variable con la type() función.

```
x = 5
y = "John"
print(type(x))
print(type(y))
```

Recorrido básico de Python

Comentarios

Python tiene la capacidad de comentar con el propósito de documentar el código.

Los comentarios comienzan con un # y Python representará el resto de la línea como un comentario:

```
#This is a comment.
print("Hello, World!")
```

Recorrido básico de Python

Operadores de Python

Los operadores se utilizan para realizar operaciones sobre variables y valores.

Python divide los operadores en los siguientes grupos:

- Operadores aritméticos
- Operadores de Asignación
- Operadores de comparación
- Operadores lógicos
- Operadores de identidad

Recorrido básico de Python

Operadores aritméticos de Python

Los operadores aritméticos se utilizan con valores numéricos para realizar operaciones matemáticas comunes:

Operador	Nombre	Ejemplo
+	Adición	x + y
–	Subtracción	x – y
*	Multiplicación	x * y
/	División	x / y
%	Modulo	x % y
**	Exponente	x ** y
//	División de Piso	x // y

Recorrido básico de Python

Operadores de asignación de Python

Los operadores de asignación se utilizan para asignar valores a las variables:

Operador	Ejemplo de uso	Igual a
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x – 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
//=	x //= 3	x = x // 3
**=	x **= 3	x = x ** 3
&=	x &= 3	x = x & 3
|=	x |= 3	x = x | 3
^=	x ^= 3	x = x ^ 3
>>=	x >>= 3	x = x >> 3
<<=	x <<= 3	x = x << 3

Recorrido básico de Python

Operadores de comparación de Python

Los operadores de comparación se utilizan para comparar dos valores:

Operator	Example
==	x == y
!=	x != y
>	x > y
<	x < y
>=	x >= y
<=	x <= y

Recorrido básico de Python

Condiciones de Python

Python admite las condiciones lógicas habituales de las matemáticas:

Es igual a: a == b

No es igual a: a! = B

Menor que: a <b

Menor o igual a: a <= b

Mayor que: a> b

Mayor o igual a: a> = b

Recorrido básico de Python

Declaraciones If

Una «instrucción if» se escribe utilizando la palabra clave if .

```
a = 33
b = 200
if b > a:
  print("b is greater than a")
```

Recorrido básico de Python

Elif

La palabra clave elif es la forma que tiene Python de decir «si las condiciones anteriores no eran verdaderas, pruebe esta condición».

```
a = 33
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
```

Recorrido básico de Python

Else

La palabra clave else captura cualquier cosa que no sea detectada por las condiciones anteriores.

```
a = 200
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
else:
  print("a is greater than b")
```

Recorrido básico de Python

Operador Lógico **AND y OR**

La palabra clave **and** es un operador lógico y se usa para combinar declaraciones condicionales:

```
a = 200
b = 33
c = 500
if a > b and c > a:
  print("Both conditions are True")
```

Recorrido básico de Python

La **or** palabra clave es un operador lógico y se usa para combinar declaraciones condicionales:

```
a = 200
b = 33
c = 500
if a > b or a > c:
  print("At least one of the conditions is True")
```

Recorrido básico de Python

Anidado de If

Puede tener if declaraciones dentro de if declaraciones, esto se llama declaraciones anidadas if .

```
x = 41

if x > 10:
  print("Above ten,")
  if x > 20:
    print("and also above 20!")
  else:
    print("but not above 20.")
```

Estructuras Repetitivas de Python

Python tiene dos comandos de bucle primitivos:

while

for

El bucle while

Con el bucle While podemos ejecutar un conjunto de instrucciones, siempre y cuando se cumpla una condición.

```
i = 1
while i < 6:
  print(i)
  i += 1
```

El bucle for

Un bucle for se usa para iterar sobre una secuencia (que es una lista, una tupla, un diccionario, un conjunto o una cadena).

Esto es menos parecido a la palabra clave for en otros lenguajes de programación y funciona más como un método iterador que se encuentra en otros lenguajes de programación orientados a objetos.

Con el ciclo for podemos ejecutar un conjunto de declaraciones, una vez para cada elemento de una lista, tupla, conjunto, etc.

Recorrido básico de Python

El bucle for

```
fruits = ["apple", "banana", "cherry"]

for x in fruits:
  print(x)
```

Recorrido básico de Python

Colecciones de Python (matrices)

Hay cuatro tipos de datos de recopilación en el lenguaje de programación Python:

- **List** es una colección ordenada y modificable. Permite miembros duplicados.
- **Tuple** es una colección ordenada e inmutable. Permite miembros duplicados.
- **Set** es una colección que no está ordenada ni indexada. No hay miembros duplicados.
- **Dictionary** es una colección ordenada * y modificable. No hay miembros duplicados.

RESUMEN DE SESIÓN

GOBIERNO DE
EL SALVADOR

SECRETARÍA DE
INNOVACIÓN