



Python Crash Course

TB022 - Esteban - Riesgo - Kristal



Temario necesario

El conocimiento de los siguientes temas va a ser necesarios para poder cursar esta materia

Funciones

Condiciones - Ciclos

Secuencias (Strings-Listas)

Diccionarios

Archivos

Excepciones

Objetos

En las siguientes diapositivas vamos a ver/repasar estos temas de forma básica. Vamos a dejar links a explicaciones más completas provenientes del curso de Fundamentos TB021 - Essaya.



Instalación

Python ya viene instalado en cualquier sistema Unix moderno.



Funciones

Información completa: [link](#)

```
def funcion(arg1: str, arg2: str = "valor_default"):  
    return arg1 + arg2
```

```
funcion("1 ") => "1 valor_default"  
funcion("1 ", "2") => "1 2"  
funcion(1, "2") => TypeError
```

Ciclos

Información completa: [link](#)

```
while <condicion>:  
    <hacer algo>
```

```
x = 0  
while x < 10:
```

```
    print("El valor de x es:", x)  
    x += 1
```

```
for <var> in range(hasta,desde,paso):  
    <hacer algo>  
  
for x in range(10):  
    # de 1 en 1 empezando por 0 sin pasarse de 10  
    print("El valor de x es:", x)  
  
for x in range(10,1,-2):  
    # de 2 en 2 empezando por 1 sin pasarse de 10  
    print("El valor de x es:", x)
```

Secuencias

Información completa: [link](#) y [link](#)

```
cadena: str = "Esto es un string"
```

Inmutable, no se puede cambiar

```
lista: list[str] = ["Esto", "es", "una", "lista", "de", "strings"]
```

Mutable, se puede
cambiar
lista[i] = valor

```
for i in range(len(lista)):  
    palabra = lista[i]  
    print(palabra)
```

```
for palabra in lista:  
    print(palabra)
```

```
for i in range(len(cadena)):  
    letra = cadena[i]  
    print(letra)
```

```
for letra in cadena:  
    print(letra)
```



Interpolación de cadenas

Podemos imprimir algo así

```
print("El valor de x es:", x)
```

o podemos hacerlo interpolando la cadena para que se más cómodo

```
print(f"El valor de x es: {x}")
```



Diccionarios

Conjunto asociativo de claves y valores.

Para cada clave, hay un único valor que le corresponde.

Argentina => .com.ar

Uruguay => .com.uy

Brasil => .br

España => .es

```
{  
    "Argentina": ".com.ar",  
    "Uruguay": ".com.uy",  
    "Brasil": ".br",  
    "España": ".es",  
}
```



Diccionarios

Información completa: [link](#)

Sintaxis y operaciones en Python

```
dict = {  
    "Argentina": ".com.ar",  
    "Uruguay": ".com.uy",  
    "Brasil": ".br",  
    "España": ".es",  
}  
  
if "Argentina" in dict:  
    print("Está en el diccionario")  
  
for clave in dict:  
    print(f"Clave: {clave}, Valor: {dict[clave]}")  
  
    dict["Argentina"] # => ".com.ar"  
    dict["Uruguay"] # => ".com.ar"  
    dict["Argentina"] = ".ar" # => Son mutables  
    dict["Marruecos"] # => KeyError  
  
    for clave, valor in dict.items():  
        print(f"Clave: {clave}, Valor: {valor}")
```



Archivos

Información completa: [link](#)

Leer y escribir archivos de texto

```
with open(<path>) as f:  
    for linea in f:  
        ...  
  
    with open(<path>, 'w') as file:  
        file.write(<string> + "\n")  
  
with open(<path>) as f:  
    linea = f.readline()  
    while linea != '':  
        ...  
        linea = f.readline()
```

Excepciones

Forma en la que se manejan errores excepcionales en el lenguaje.

Cuando se ejecuta una línea de código que no puede ser resuelta por presenta un error, se lanza una excepción.

5 / 0

l = [1,2,3]

l[10]

"3" + 2

```
def f():
    g()
    def g():
        h()
        def h():
            raise ValueError("Error de ejecución")
```

Excepciones

Forma en la que se manejan errores excepcionales en el lenguaje.

```
def f():
    g()

def g():
    h()

def h():
    raise ValueError("Error de ejecución")

def main():
    try:
        f()
    except Exception as e:
        print(f"Hubo un error: {e}")
```



Objetos



Fin