

{desafío}
latam_

Estructuras de datos _

Sesión Experimental 1



Itinerario



Activación de conceptos

Desarrollo Desafío

Panel de discusión

/* Activación de conceptos */

¿Cuál de las siguientes no es una llamada a una función?

1. `LinearRegression.fit()`
2. `len("perro")`
3. `pd.shape`
4. `int("2")`
5. `np.array([1,2,3])`

¿Cómo puede corregir el siguiente error?

(considere número = 5)

```
1 incrementar_en_uno()
```

```
-----  
TypeError                                Traceback (most recent call last)  
<ipython-input-12-ea1a5f1be123> in <module>()  
----> 1 incrementar_en_uno()  
  
TypeError: incrementar_en_uno() missing 1 required positional argument: 'numero'
```

1. incrementar_en_uno.numero()
2. numero(incrementar_en_uno)
3. numero.incrementar_en_uno()
4. incrementar_en_uno().numero
5. incrementar_en_uno(numero)

¿Qué palabra se utiliza dentro del bloque de la función para devolver un resultado?

1. give
2. show
3. print
4. return
5. break

¿Cuál es el alcance de una variable global?

1. El espacio `__main__`
2. El espacio de la primera función definida
3. El espacio `__run__`
4. El espacio de cada función llamada
5. El espacio `__principal__`

Listas

- Contenedores de datos
- Mutables
- Se definen con corchetes
- Se separan los elementos con comas

```
1 cadenas = ["perro", "casa", "verde"]  
2 numeros = [1, 2, 3, 4]  
3 mixta = [1.1, 3, True, "perro"]
```


Índices

```
1 cadenas = ["perro", "casa", "verde"]
2 numeros = [1, 2, 3, 4]
3 mixta = [1.1, 3, True, "perro"]
```

```
1 posiciones_indices = ["indice cero", "indice uno", "indice dos"]
2 print(posiciones_indices[0])
3 print(posiciones_indices[1])
4 print(posiciones_indices[2])
```

indice cero
indice uno
indice dos

```
1 animales = ["perro", "gato"]
2 print("El {} está en el índice {}".format(animales[0]))
```

El perro está en el índice 0

```
1 # No se puede solicitar un elemento con un índice no asignado
2 animales[2]
```

```
-----
IndexError                                Traceback (most recent call last)
<ipython-input-4-c9eace203131> in <module>()
      1 # No se puede solicitar un elemento con un índice no asignado
----> 2 animales[2]
```

`IndexError`: list index out of range

Funciones de listas

```
1 animales = ["perro", "gato"]  
2  
3 # append: Agrega un elemento al final  
4 animales.append("erizo")  
5 animales
```

['perro', 'gato', 'erizo']

```
1 # insert: Agrega un elemento en una posición  
2 animales.insert(2, "hurón")  
3 animales
```

['perro', 'gato', 'hurón', 'erizo']

```
1 # pop: saca el último elemento  
2 animales.pop()  
3 animales
```

['perro', 'gato', 'hurón']

```
1 # remove: Elimina el elemento indicado  
2 animales.remove("gato")  
3 animales
```

['perro', 'hurón']

Operaciones en listas

```
1 # "+" concatena listas
2 animales = ['perro', 'gato', 'hurón', 'erizo']
3 colores = ["rosa", "azul", "morado"]
4 animales + colores
```

['perro', 'gato', 'hurón', 'erizo', 'rosa', 'azul', 'morado']

```
1 # "*" multiplica los elementos y los añade
2 colores * 2
```

['rosa', 'azul', 'morado', 'rosa', 'azul', 'morado']

Comprensiones de Listas

```
1 mascotas = animales * 2
2
3 # Comprensión aplicando un for
4 mascotas_mayusculas = [i.upper() for i in mascotas]
5 mascotas_mayusculas
```

['PERRO', 'GATO', 'HURÓN', 'ERIZO', 'PERRO', 'GATO', 'HURÓN', 'ERIZO']

```
1 mascotas = animales * 2
2
3 # Comprensión aplicando un for y un condicional
4 [x.upper() if x != 'gato' else x.lower() for x in mascotas]
```

['perro', 'GATO', 'hurón', 'erizo', 'perro', 'GATO', 'hurón', 'erizo']

Iterar una lista usando enumerate

```
1 mascotas = ['PERRO', 'GATO', 'HURÓN', 'ERIZO', 'PERRO', 'GATO', 'HURÓN', 'ERIZO']
2
3 # index será el índice o posición, y element el elemento iterado
4 for index, element in enumerate(mascotas):
5     print("Iterando en índice {}. Elemento: {}".format(index, element))
```

```
Iterando en índice 0. Elemento: PERRO
Iterando en índice 1. Elemento: GATO
Iterando en índice 2. Elemento: HURÓN
Iterando en índice 3. Elemento: ERIZO
Iterando en índice 4. Elemento: PERRO
Iterando en índice 5. Elemento: GATO
Iterando en índice 6. Elemento: HURÓN
Iterando en índice 7. Elemento: ERIZO
```

Transformación de datos

```
1 numeros_string = ["200", "40", "9090", "3", "777"]
2 numeros_int = []
3
4 # Transformamos en cada iteración y agregamos a la lista vacía
5 # "i" es el elemento iterado, NO el índice
6 for i in numeros_string:
7     numeros_int.append(int(i))
8
9 print(numeros_int)
```

[200, 40, 9090, 3, 777]

Filtrado de datos

```
1 a = [100, 200, 1000, 5000, 10000, 10, 5000]
2 n = len(a)
3 filtered_array = []
4
5 for i in range(n):
6     if a[i] >= 1000:
7         filtered_array.append(a[i])
8
9 filtered_array
```

[1000, 5000, 10000, 5000]

Operaciones funcionales

```
1 numeros = [200, 40, 9090, 3, 777]
2
3 # En lugar de for, usamos map para transformar el dato
4 # map recibe una función y una lista como parámetros
5 map_numeros = map(lambda i: i * 2, numeros)
6
7 # Luego se debe transformar el map a lista
8 list(map_numeros)
```

[400, 80, 18180, 6, 1554]

```
1 # En lugar de for, usamos filter para filtrar datos
2 # filter recibe una función y una lista como parámetros
3 return_even_filter = filter(lambda x: x % 2 == 0, numeros)
4
5 # Se debe transformar a lista
6 list(return_even_filter)
```

[200, 40, 9090]

```
1 # Se debe importar
2 from functools import reduce
3
4 # Se opera sobre a, donde uno de los parámetros (x) actúa como acumulador,
5 # y el otro (y) actúa como iterador
6 reduce(lambda x, y: x + y, numeros)
```

10110

/* Desafío */

/* Panel de discusión */

{desafío}
latam_

*Academia de
talentos digitales*

www.desafiolatam.com