Ejercicios de
Tipos de datos estructurados

Programación — DAW

Ricardo Pérez López
IES Doñana

Curso 2024/2025

1 Devolver el último elemento

Crear una función que devuelva el valor del último elemento de una lista o cadena.

Ejemplos

```
last_ind([0, 4, 19, 34, 50, -9, 2]) == 2
last_ind("The quick brown fox jumped over the lazy dog") == "g"
last_ind([]) == None
```

Notas

- Las listas/cadenas pueden ser de cualquier tamaño.
- Devuelve `None` si la lista/cadena está vacía.

Fuente

https://edabit.com/challenge/mcC546MLnBjNLXTb8

2 Asignar personas a profesiones

Tienes dos listas. Una muestra los nombres de las personas (`names`), mientras que el otro muestra sus profesiones (`jobs`). Tu tarea es crear un diccionario que muestre a cada persona con su respectiva profesión.

Persona	Profesión
Annie	Profesor
Steven	Ingeniero
Lisa	Doctor
Osman	Cajero

Ejemplos

```
names = ["Dennis", "Vera", "Mabel", "Annette", "Sussan"]
jobs = ["Butcher", "Programmer", "Doctor", "Teacher", "Lecturer"]

assign_person_to_job(names, jobs) = {
  "Dennis": "Butcher",
  "Vera": "Programmer",
  "Mabel": "Doctor",
  "Annette": "Teacher",
  "Sussan": "Lecturer"
}
```

Notas

- Las dos listas tienen la misma longitud.
- El índice de un nombre en la lista `names` es el mismo que el índice de la profesión de la persona en la lista `jobs`, como se muestra en la tabla anterior.

Fuente

`https://edabit.com/challenge/XPnTdP9eDkwqAQeWq`

3 Calcular pérdidas totales

¡Acabas de regresar a casa y descubres que han robado tu mansión! Dado un diccionario de los artículos robados, devolver el monto total del robo (número). Si no se robó nada, devolver la cadena `"Lucky you!"`.

Ejemplos

```
calculate_losses({
  "tv" : 30,
  "skate" : 20,
  "stereo" : 50,
}) == 100
calculate_losses({
  "painting" : 20000,
}) == 20000
calculate_losses({}) == "Lucky you!"
```

Notas

- El valor del artículo siempre es positivo.

Fuente

https://edabit.com/challenge/kSmkcYLcWRnEEwXwX

4 Cantidad total de caracteres únicos

Dadas dos cadenas, crear una función que devuelva el número total de caracteres únicos de la cadena combinada.

Ejemplos

```
count_unique("apple", "play") == 5
# "appleplay" tiene 5 caracteres únicos:
# "a", "e", "l", "p", "y"
count_unique("sore", "zebra") == 7
count_unique("a", "soup") == 5
```

Notas

- Cada palabra contendrá al menos una letra.
- Todas las palabras estarán en minúsculas.

Fuente

https://edabit.com/challenge/oTJaJ895ubqqpRPMh

5 ¿Cuál es tu tipo?

Python tiene tres tipos principales de estructuras de datos formadas por objetos más pequeños:

- Listas, escritas con corchetes `[]`, como `[1, 2, 4, 8]`.
- Tuplas, escritas entre paréntesis `()`, como `(7, 8, 9)`.
- Conjuntos, escritos con llaves `{}`, como `{2, 3, 5, 7, 11, 13}`.

Cada uno de estos tipos tiene sus propias propiedades y peculiaridades especiales que vale la pena conocer, pero este ejercicio solo consiste en transformar estos tipos de datos entre sí.

Esto se puede hacer de la siguiente manera:

- `tuple([1, 2, 4, 8])` devuelve `(1, 2, 4, 8)`
- `list({2, 3, 5, 7, 11})` devuelve `[2, 3, 5, 7, 11, 13]`
- `set((1, 2, 4))` devuelve `{1, 2, 4}`

Dadas dos estructuras de datos, `data1` y `data2`, devolver `data2` convertido al tipo de `data1`.

Ejemplos

```
convert([1, 2, 4, 8], (7, 8, 9)) == [7, 8, 9]
convert((7, 8, 9), [1, 2, 4, 8]) == (1, 2, 4, 8)
convert([1, 2, 4, 8], {2, 3, 5, 7, 11, 13}) == [2, 3, 5, 7, 11, 13]
convert({2, 3, 5, 7, 11, 13}, [1, 2, 4, 8]) == {8, 1, 2, 4}
```

Notas

- Es posible que haya notado que el último ejemplo da `{8, 1, 2, 4}` en lugar de `{1, 2, 4, 8}`. Esto tiene que ver con el hecho de que en los conjuntos el orden no importa, por lo que Python considera que `{8, 1, 2, 4}` y `{1, 2, 4, 8}` es el mismo conjunto.
- En los casos de prueba que no tendrá que preocuparse acerca de las órdenes: las respuestas siempre tendrán la orden dada por la aplicación de las funciones `list()`, `tuple()`, `set()`.

Fuente

`https://edabit.com/challenge/GPRZauknkEbyRz5qM`

6 Añade su nombre

Dados tres argumentos (un diccionario `obj`, una clave `name` y un valor `value`) devolver un diccionario con ese nombre y valor (como pares clave-valor).

Ejemplos

```
add_name({}, "Brutus", 300) == {"Brutus": 300}
add_name({"piano": 500}, "Brutus", 400) == {"piano": 500, "Brutus": 400}
add_name({"piano": 500, "stereo": 300}, "Caligula", 440)
    == {"piano": 500, "stereo": 300, "Caligula": 440}
```

Notas

- El argumento `value` será un número.

Fuente

https://edabit.com/challenge/owmKG47zYRJw72aiD

7 Purgar y organizar

Dada una lista de números, escribir una función que devuelva una lista que…

1. Elimine todos los elementos duplicados.
2. Esté ordenado de menor a mayor valor.

Ejemplos

```
unique_sort([1, 2, 4, 3]) == [1, 2, 3, 4]
unique_sort([1, 4, 4, 4, 4, 4, 3, 2, 1, 2]) == [1, 2, 3, 4]
unique_sort([6, 7, 3, 2, 1]) == [1, 2, 3, 6, 7]
```

Fuente

https://edabit.com/challenge/EZMCpHaNFg2Yfsnxx

8 Unos verdaderos, ceros falsos

Crear una función que devuelva una lista de valores *booleanos*, a partir de un número dado. Repitiendo el dígito del número uno a la vez, agregar `True` si el dígito es `1` y `False` si es `0`.

Ejemplos

```
integer_boolean("100101") == [True, False, False, True, False, True]
integer_boolean("10") == [True, False]
integer_boolean("001") == [False, False, True]
```

Notas

- Los números sólo pueden ser `0` y `1`.

Fuente

https://edabit.com/challenge/58DYAThA2dxnAsMpL

9 Obtener nombres de estudiantes

Crear una función que tome un diccionario de nombres de estudiantes y devuelva una lista de nombres de estudiantes en orden alfabético.

Ejemplos

```
get_student_names({
  "Student 1" : "Steve",
  "Student 2" : "Becky",
  "Student 3" : "John"
}) == ["Becky", "John", "Steve"]
```

Fuente

https://edabit.com/challenge/HvkPdhijquecKASdF