



Reto KATA

# Desarrollo Reto Técnico

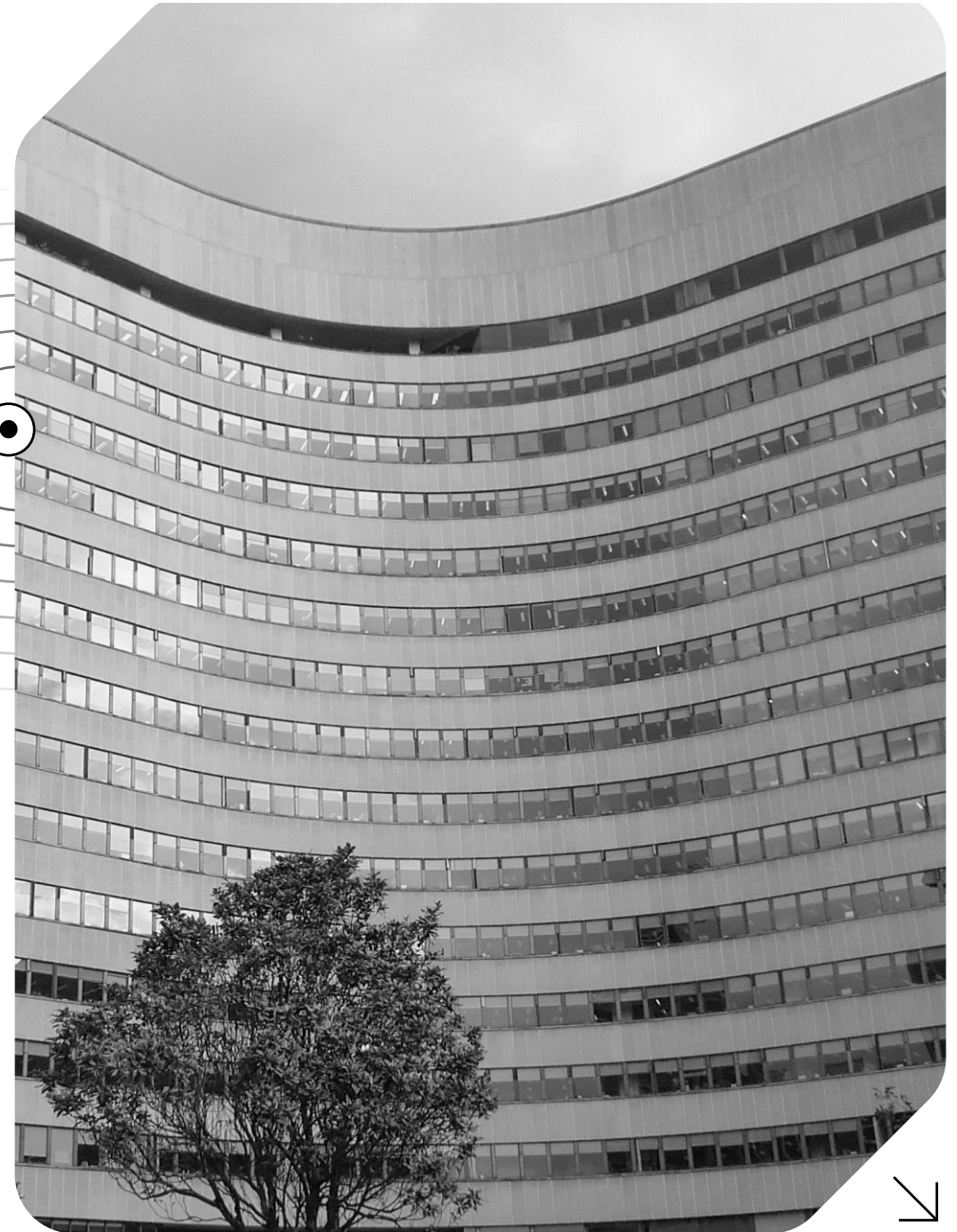
Juan Felipe Rodríguez Galindo  
**@Juferoga**

---



[Ir al índice](#) ♦

[Contacto](#)

Empezar





 “La mejor manera de  
Predecir  el futuro  
es creándolo”

Empezar



# Índice



Introducción



Análisis



Diseño



Implementación



Extras



Agradecimientos



Fin



Repositorio

Inicio



Contacto ♦



# Introducción

Soy **Juan Felipe Rodríguez**, un desarrollador comprometido con soluciones eficientes. Mi experiencia en instituciones como la Agencia Catastral de Cundinamarca, el Instituto Geográfico Agustín Codazzi y domino estudio me ha enseñado a equilibrar precisión técnica y eficiencia. Veo esta oportunidad como el siguiente paso para aplicar mis habilidades y contribuir a su evolución digital.





# Enunciado Del problema



1

Adaptar un arreglo según las condiciones e invertir el orden de este



2

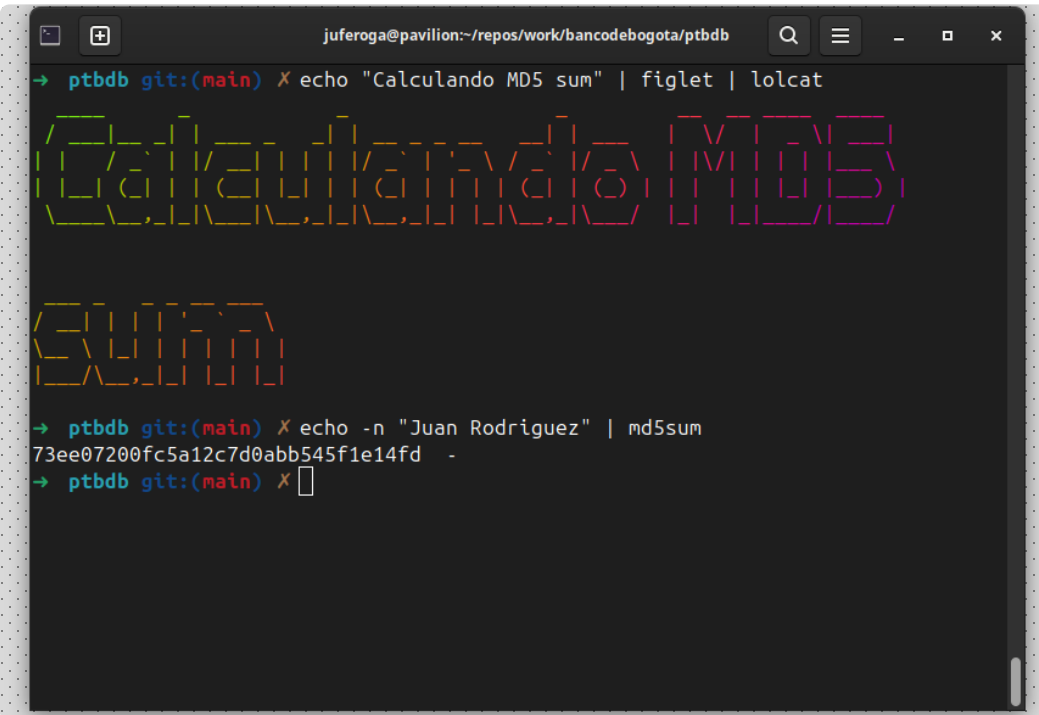
Modificar un arreglo elevando cada elemento al cuadrado y eliminando los que no cumplan la condición



3

Encontrar el mínimo monto no alcanzable con las monedas dadas en un arreglo





```
juferoga@pavilion:~/repos/work/bancodebogota/ptbdb
→ ptbdb git:(main) X echo "Calculando MD5 sum" | figlet | lolcat
Calculando MD5 sum
73ee07200fc5a12c7d0abb545f1e14fd -
→ ptbdb git:(main) X
```

Adquisición del hash

*Autoría propia*

# Preparación

Para el inicio del reto se tendrá en cuenta realizar el procedimiento para adquirir el hash y tomar el primer número para asignarlo como valor de “s”.





# Análisis

Having a list of  $n$  numbers with digits in range  $[0, S]$ , where  $n \leq 100$ , switch all list positions in  $O(n)$  time.

If the input number contains a digit greater or equal than  $S$ , you will delete the digit from the number, for example with  $S=6$ , 61 becomes 1, and 6 will be deleted from the array. The result should be printed in console/terminal. Please, don't use built-in sort of your language.

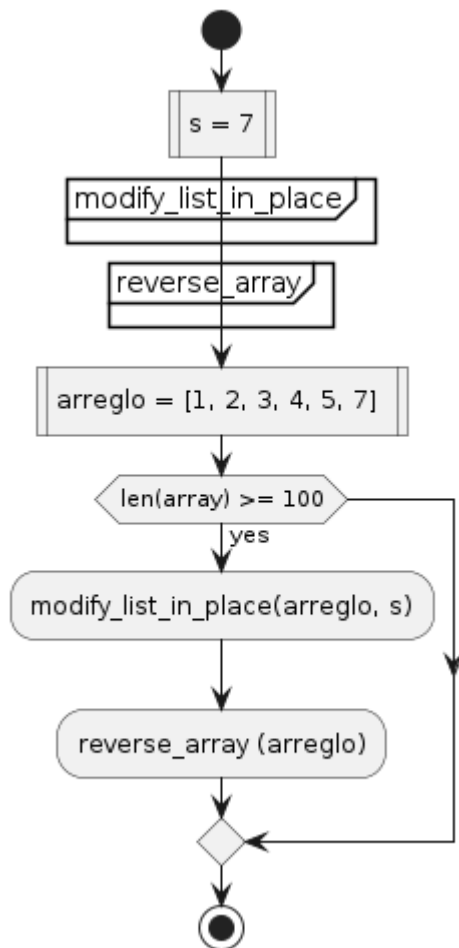
## RETO 1

- Como entrada se debe admitir una lista de no más de 100 elementos dentro de un rango de 0 a 7 para el presente caso.
- Como el algoritmo debe tener una complejidad  $O(n)$  se debe recorrer todo el arreglo.
- Se debe de invertir el orden del arreglo.
- Si uno de los números del arreglo contiene un dígito o igual a 7, se eliminará este.

1

2

3



# Diseño Algoritmo General

Se crean las dos funciones `modify_list_in_place`, la cual se encargará de modificar el arreglo eliminando los dígitos que sean iguales a "s". Y la función `reverse_array`, que modificará el arreglo invirtiéndola. Por último, se llamarán antes de comprobar la longitud del arreglo.

1

2

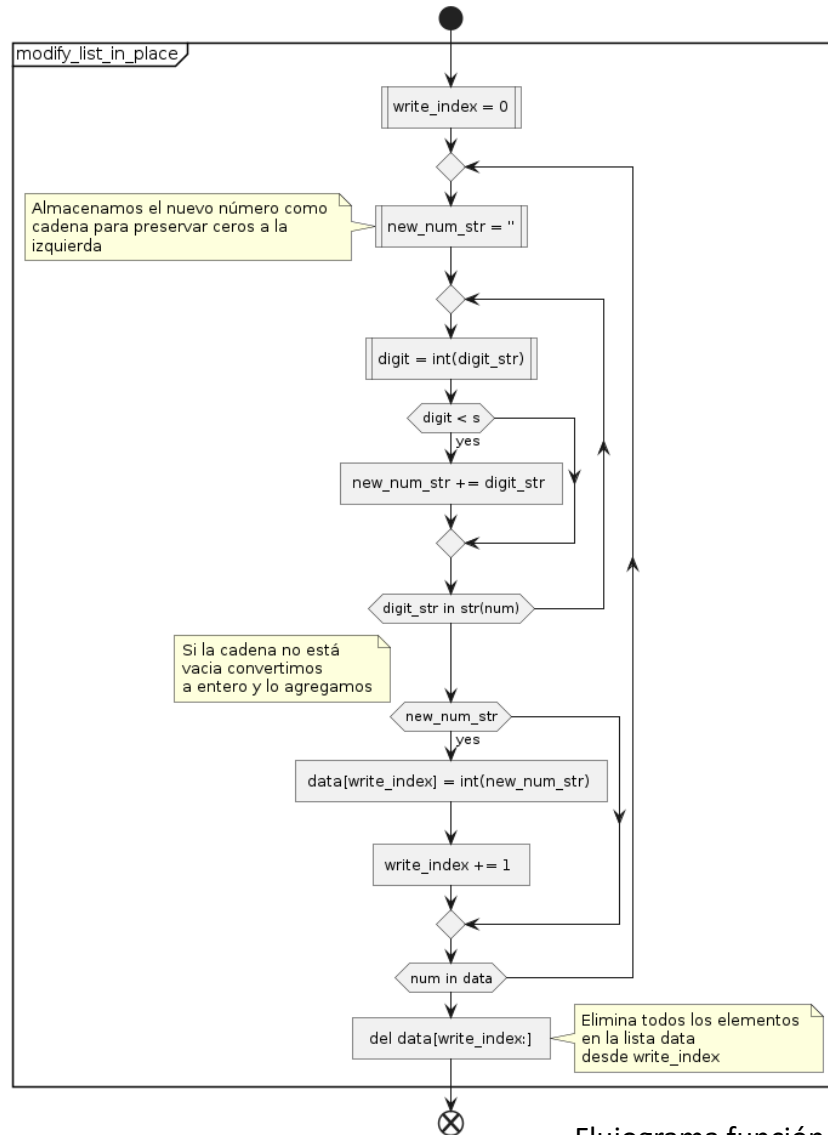
3

Flujograma algoritmo general

*Autoría propia*

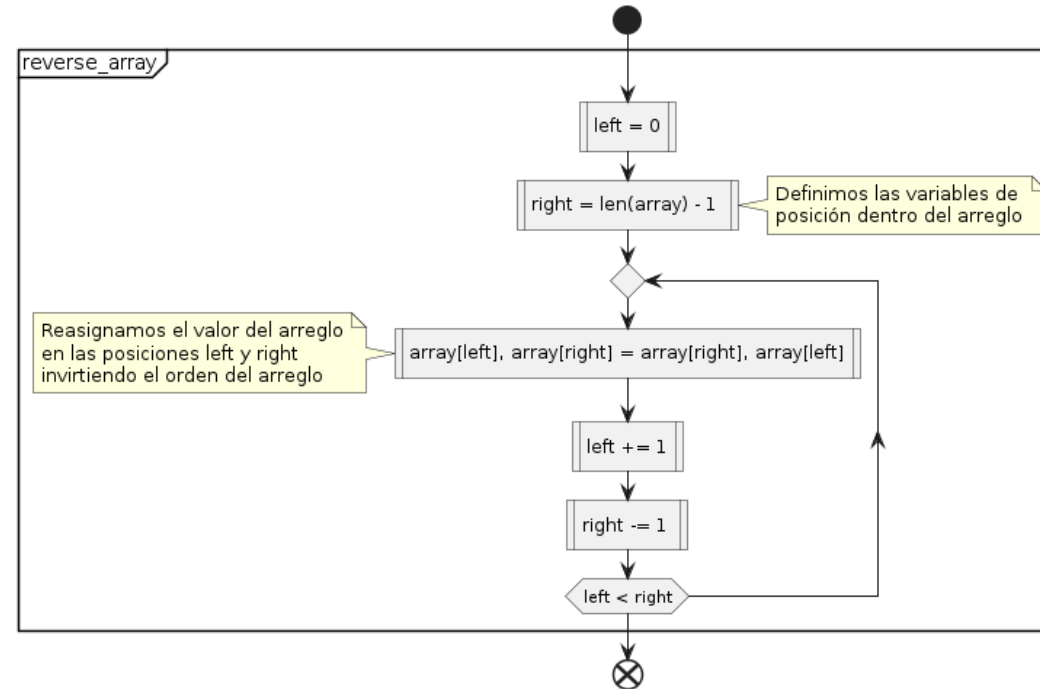






Flujograma función modificar

*Autoría propia*



Flujograma función reversar arreglo

*Autoría propia*

1

2

3

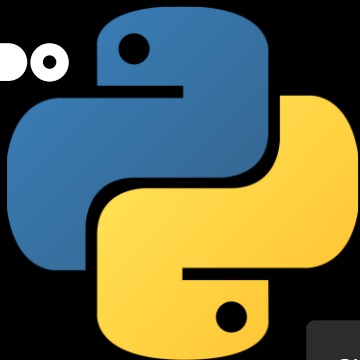




# Implementación

The screenshot shows a GitHub Actions workflow run for the repository 'Juferoga / ptbdb'. The workflow is named 'docs: improve icon #7'. The run is titled 'test-1' and succeeded 2 days ago in 7s. The workflow consists of three jobs: 'test-1', 'test-2', and 'test-3'. The 'test-1' job is selected, showing its logs. The logs indicate that the job was set up, checked out, and then executed a script. The script output shows the execution of a Python main.py file, which prints the original array, the result of the modify\_list\_in\_place function, and the result of the reverseArray function. The output for the first test case is: 'Array original [1, 2, 3, 4, 5, 7]', 'Función modify\_list\_in\_place y resultado [1, 2, 3, 4, 5]', and 'Función reverseArray y resultado [5, 4, 3, 2, 1]'. The output for the second test case is: 'Array original [10, 20, 30, 40]', 'Función modify\_list\_in\_place y resultado [10, 20, 30, 40]', and 'Función reverseArray y resultado [40, 30, 20, 10]'. The output for the third test case is: 'Array original [7]', 'Función modify\_list\_in\_place y resultado [7]', and 'Función reverseArray y resultado [7]'.

Repo.



GitHub





# Análisis

Write a function that takes in a non-empty array of integers sorted in ascending order and returns a new array of the same length with the squares of the original integers also sorted in ascending order.

If the output number is out of the range  $[0, SS]$  (for  $S=6$  the range will be  $[0, 66]$ ), you will delete it of the output array. Please, don't use built-in sort of your language.

## RETO 2

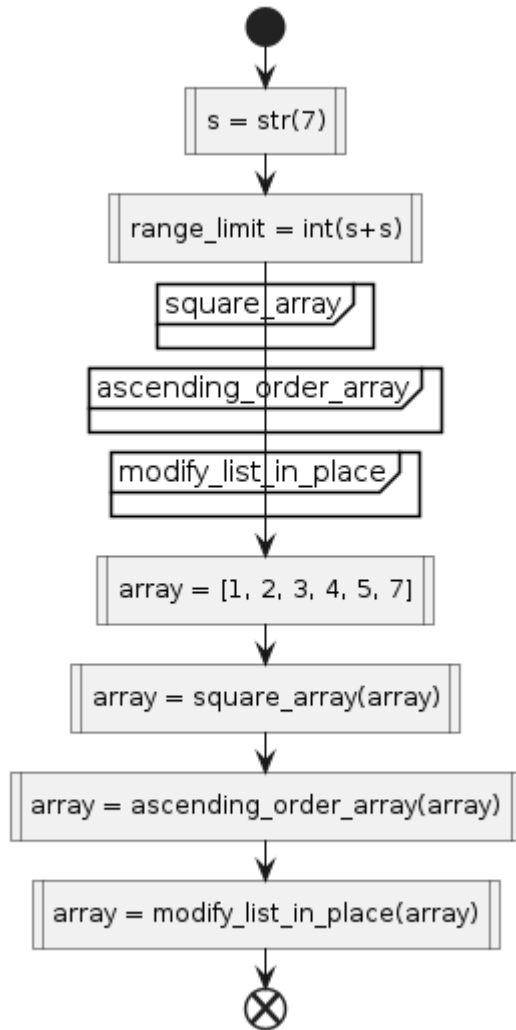
- Se debe escribir una función que reciba un arreglo no vacío de enteros.
- Se debe elevar cada elemento al cuadrado.
- Se debe eliminar los elementos que no estén en el rango  $[0, 77]$ .

1

2

3





Flujograma algoritmo general

*Autoría propia*

# Diseño Algoritmo General

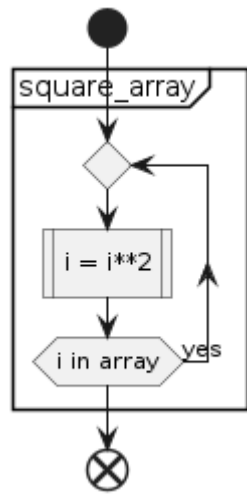
Se crean las funciones `square_array`, la cual eleva los elementos al cuadrado, `ascending_order_array`, la cual se encarga de organizar el arreglo de forma ascendente y `modify_list_in_place` la cual modifica la lista según las especificaciones del reto cuando el elemento supere el límite.

1

2

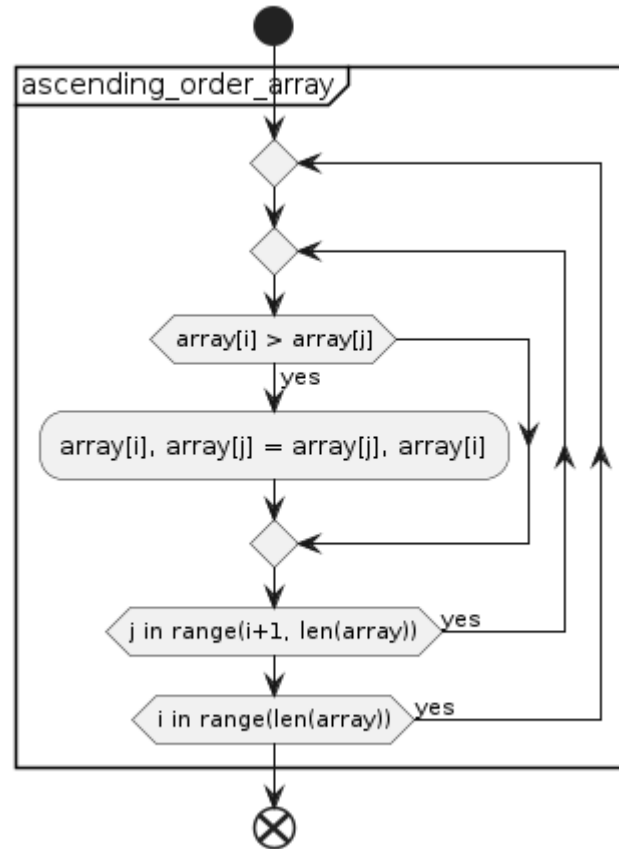
3





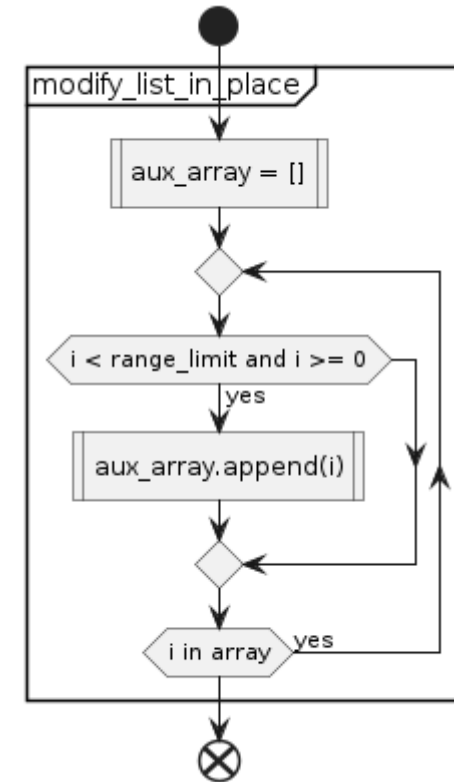
Flujograma función cuadrado

*Autoría propia*



Flujograma función organizar orden  
ascendente

*Autoría propia*



Flujograma función modificar lista con  
base en un rango

*Autoría propia*

1

2

3



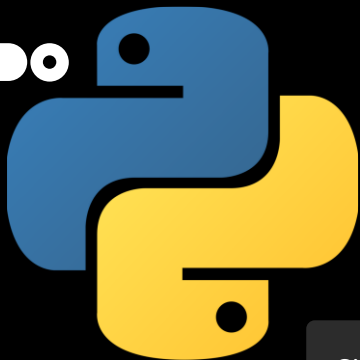


# Implementación

The screenshot shows a GitHub Actions workflow run for the repository 'Juferoga / ptbdb'. The workflow is named 'docs: improve icon #7'. The left sidebar shows the 'Jobs' section with three jobs: 'test-1', 'test-2', and 'test-3'. 'test-2' is selected. The main area shows the details of 'test-2', which succeeded 2 days ago in 5s. The workflow steps are: 'Set up job' (1s), 'Run actions/checkout@v3' (1s), and 'Ejecutar script 2' (0s). The 'Ejecutar script 2' step is expanded, showing a Python script that runs 'main.py' and prints the results of three tests. The output shows that all three tests passed.

```
1 ▶ Run python main.py
4 ]-----[Test 0]-----[
5 array original
6 [1, 2, 3, 5, 6, 8, 9]
7 Función square_array y resultado
8 [1, 4, 9, 25, 36, 64, 81]
9 Función ascending_order_array y resultado
10 [1, 4, 9, 25, 36, 64, 81]
11 Función modify_list_in_place y resultado
12 [1, 4, 9, 25, 36, 64]
13 ]-----[Test 1]-----[
14 array original
15 [-2, -1]
16 Función square_array y resultado
17 [4, 1]
18 Función ascending_order_array y resultado
19 [1, 4]
20 Función modify_list_in_place y resultado
21 [1, 4]
22 ]-----[Test 2]-----[
23 array original
24 [-6, -5, 0, 5, 6]
25 Función square_array y resultado
26 [36, 25, 0, 25, 36]
27 Función ascending_order_array y resultado
```

Repo.



GitHub





# Análisis

Given an array of positive integers representing the values of coins in your possession, write a function that returns the minimum amount of change (the minimum sum of money) that you CANNOT give change. The given coins can have any positive integer value and aren't necessarily unique (i.e., you can have multiple coins of the same value). You can use built-in sort of your language.

## RETO 3

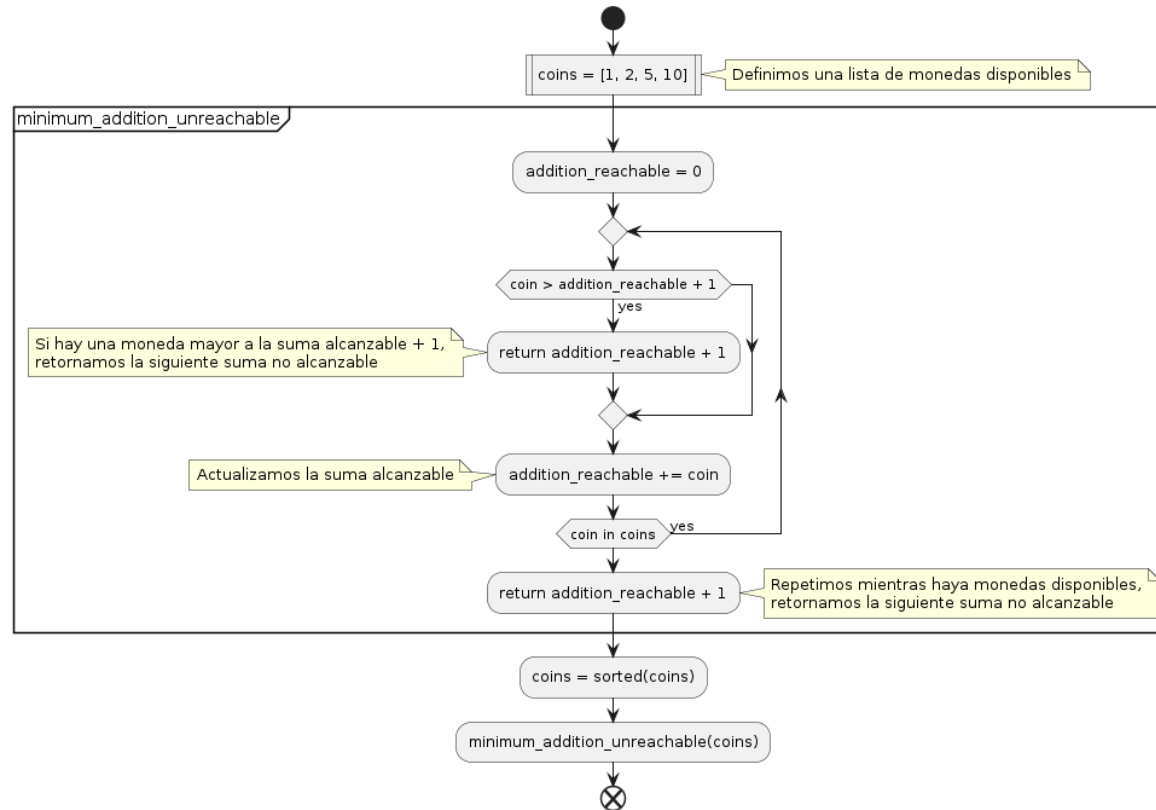
- Se debe recibir un arreglo de números enteros los cuales representan monedas.
- Se debe de organizar el arreglo para calcular la mínima suma alcanzable.
- Siempre son positivas y no son únicas, es decir, se puede tener varias monedas de la misma denominación.

1

2

3





# Diseño Algoritmo General

Se crea la función `minimum_addition_unreachable`, la cual se encargará de hallar la suma mínima alcanzable con un conjunto de monedas.

1

2

3

Flujograma algoritmo general

*Autoría propia*







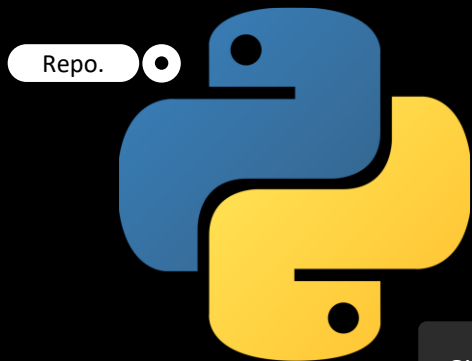
# Implementación

The screenshot shows a GitHub Actions workflow run for the repository 'Juferoga / ptbdb'. The workflow is named 'docs: improve icon #7'. The run is titled 'test-3' and succeeded 2 days ago in 4s. The workflow steps are:

- Set up job (0s)
- Run actions/checkout@v3 (0s)
- Ejecutar script 3 (0s)
- Post Run actions/checkout@v3 (0s)
- Complete job (0s)

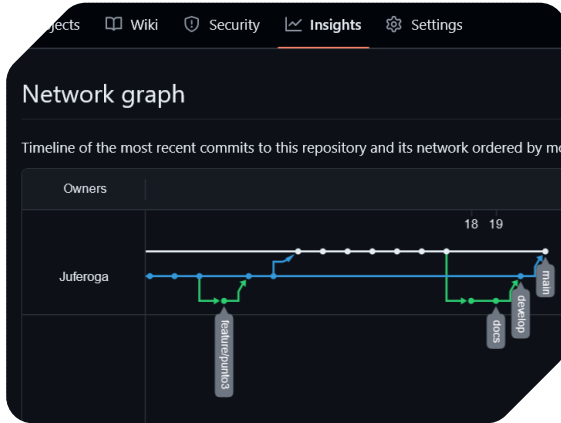
The 'Ejecutar script 3' step is expanded, showing the following code:

```
1 ▶ Run python main.py
4 ]-----[Test 0]-----[
5 Original
6 [5, 7, 1, 1, 2, 3, 22]
7 Organized [1, 1, 2, 3, 5, 7, 22]
8 Minimum addition to be unreachable
9 20
10 ]-----[Test 1]-----[
11 Original
12 [1, 1, 1, 1, 1]
13 Organized [1, 1, 1, 1, 1]
14 Minimum addition to be unreachable
15 6
16 ]-----[Test 2]-----[
17 Original
18 [1, 5, 1, 1, 1, 10, 15, 20, 100]
19 Organized [1, 1, 1, 1, 1, 5, 10, 15, 20, 100]
20 Minimum addition to be unreachable
21 55
```



GitHub ↗



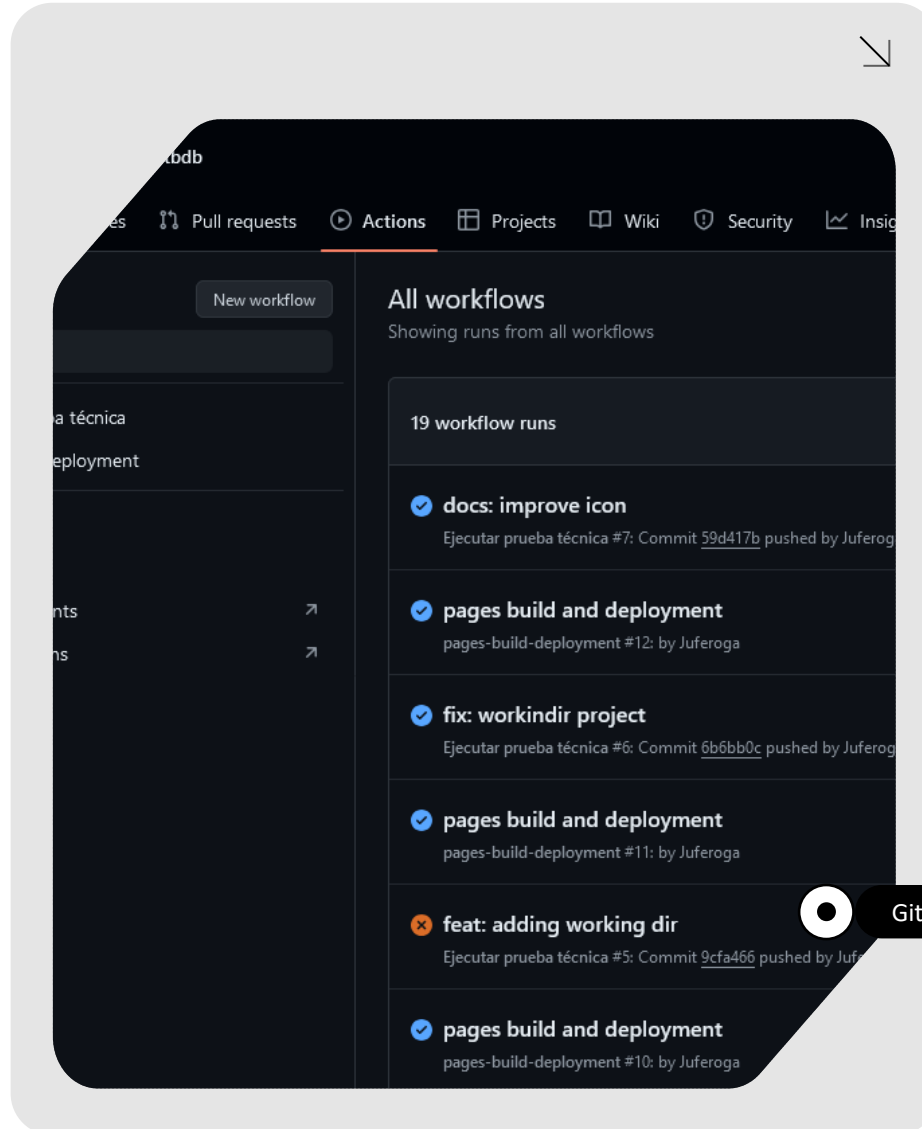


## Manejo de GIT

*Autoría propia*

### Estrategias GIT:

La implementación de Gitflow en la solución desarrollada, es fundamental para llevar el control y orden en el desarrollo.



GitHub

GitHub actions

*Autoría propia*

# Manejo de versiones

Se realizó el manejo de versiones por medio de GitHub, implementando workflows para realizar las pruebas sobre el código de los retos.





“La   creatividad es la  
Inteligencia divirtiendose”

Albert Einstein



(310) 628 6762 | [juferoga@gmail.com](mailto:juferoga@gmail.com)

[juferoga.github.io/ptbdb/](https://juferoga.github.io/ptbdb/)

Bogotá D.C.

Colombia

Contactame





Por su tiempo y  
consideración

# Gracias



Iniciar nuevamente

