

Programación

Clase 07

U4

Arreglos



Información Importante

Tienen derecho a revisar su prueba en un horario definido por cada profesor.

La fecha, hora y lugar será entregado en un anuncio en Campus Virtual.

Si no puede asistir, debe coordinar con su profesor una alternativa. Ésta debe ser antes de la próxima evaluación (3 de junio).

Información Importante

Prueba Recuperativa: **3 de junio.**

Obligatoria para quienes **JUSTIFICARON** inasistencia a la **P1.**

Optativa para el resto.

En Hawaii pueden ver cuál es su situación respecto a la **PR.**

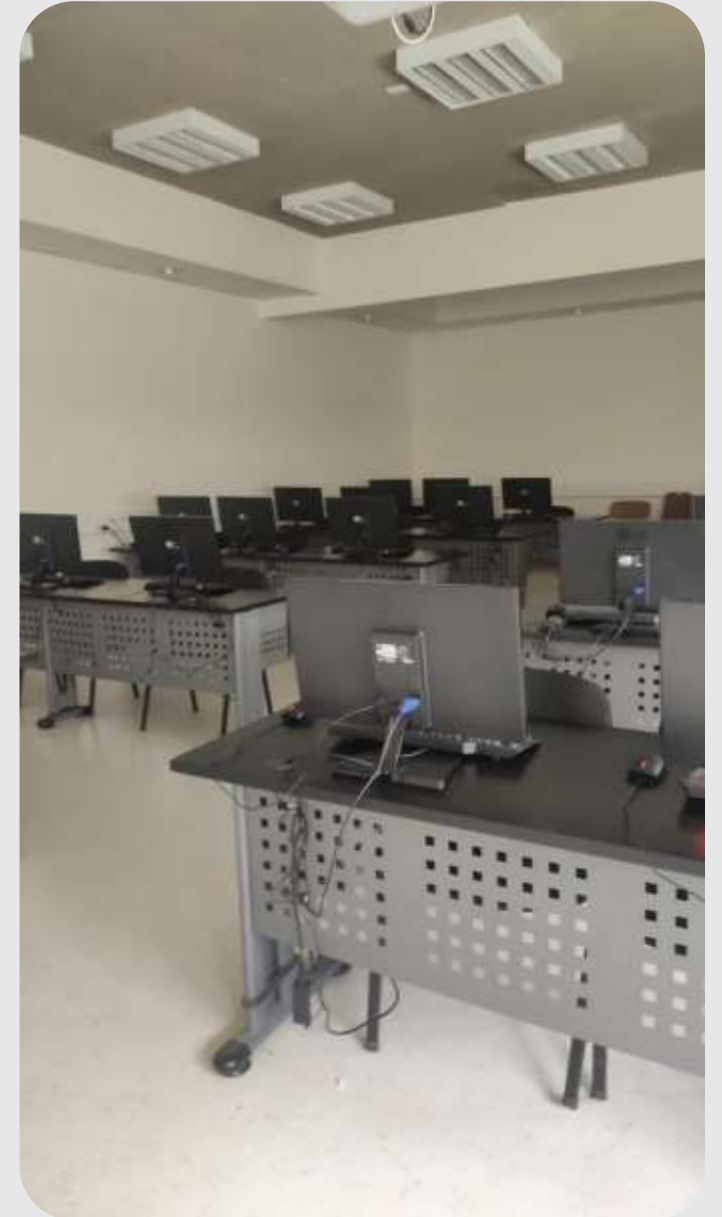
- Se promedia P1 y PR. (si $P1 \geq 4$)
- El mínimo entre PR y 4,0. (si $P1 < 4$)
- Nota máxima 4,0. (asistencia no justificada)
- Reemplaza P1. (asistencia justificada)

Información Importante

Utilice los horarios designador para las **ayudantías**. Es una de las oportunidades para resolver dudas.

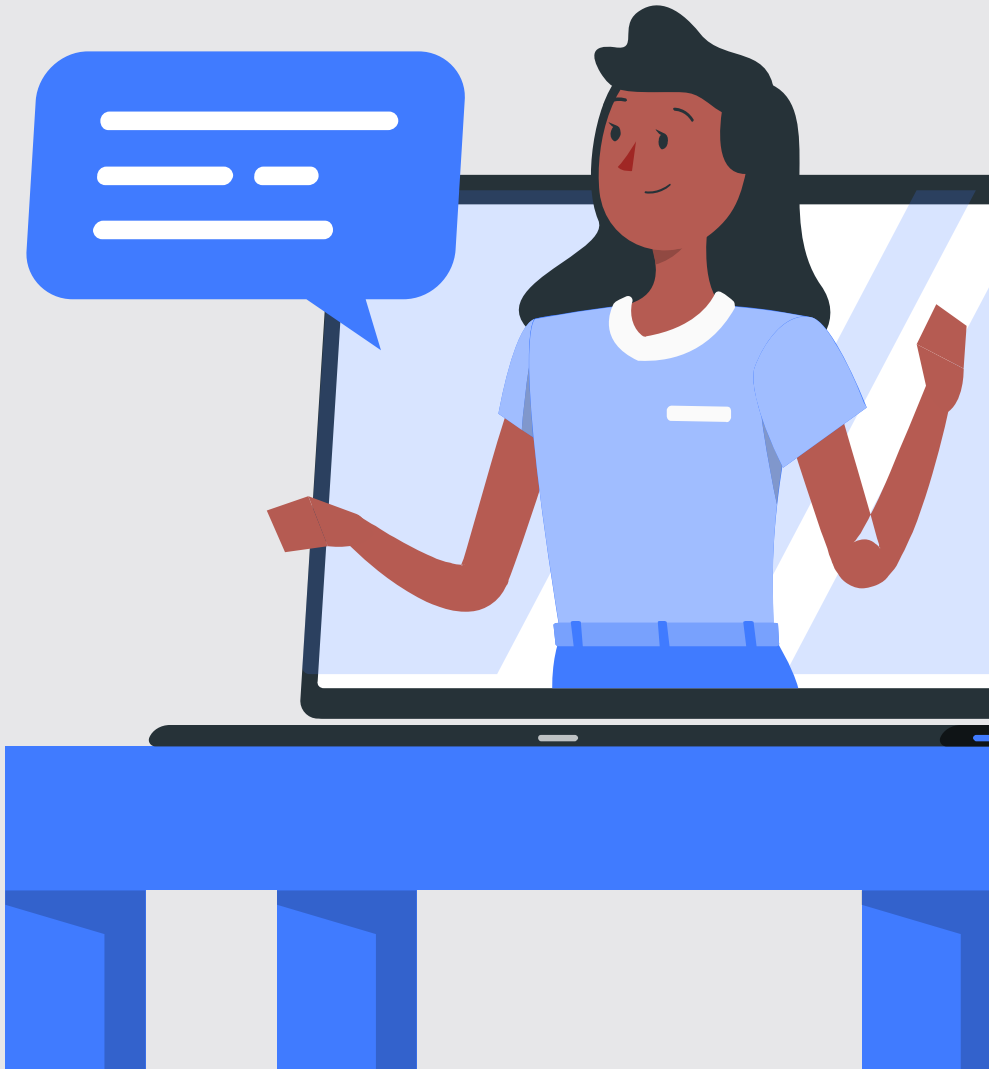
Adicionalmente existen Tutorías y otros apoyos en caso de que lo considere necesario. **Consulte a su profesor al respecto.**

El servidor de Discord es otro medio para resolver dudas, puede ingresar desde el widget del Campus Virtual.



Clase 07: Arreglos

Reflexión Encuesta



Introducción

Hasta ahora hemos aprendido a solucionar problemas que tienen que ver con contar elementos, calcular y determinar cosas como promedios, mayores, etc.

En general, leyendo desde el teclado o un archivo, podemos saber cosas de los elementos que vamos “viendo”.

Pero aun así, hay problemas que todavía no podemos solucionar.

Un ejemplo

Leyendo desde un archivo, en que cada línea contiene un nombre y una edad, determinar el nombre de la persona mayor.

```
arch = open('personas.txt', 'r')

mayor_edad = -1
mayor_nombre = ''

li = arch.readline().strip()
while li != '':
    partes = li.split()
    nombre = partes[0]
    edad = int(partes[1])
    if edad > mayor_edad:
        mayor_edad = edad
        mayor_nombre = nombre
    li = arch.readline().strip()

print(mayor_nombre, mayor_edad)
```

Un ejemplo

Leyendo desde un archivo, en que cada línea contiene un nombre y una edad, obtener un listado de los **nombres sin duplicados**.



En conclusión

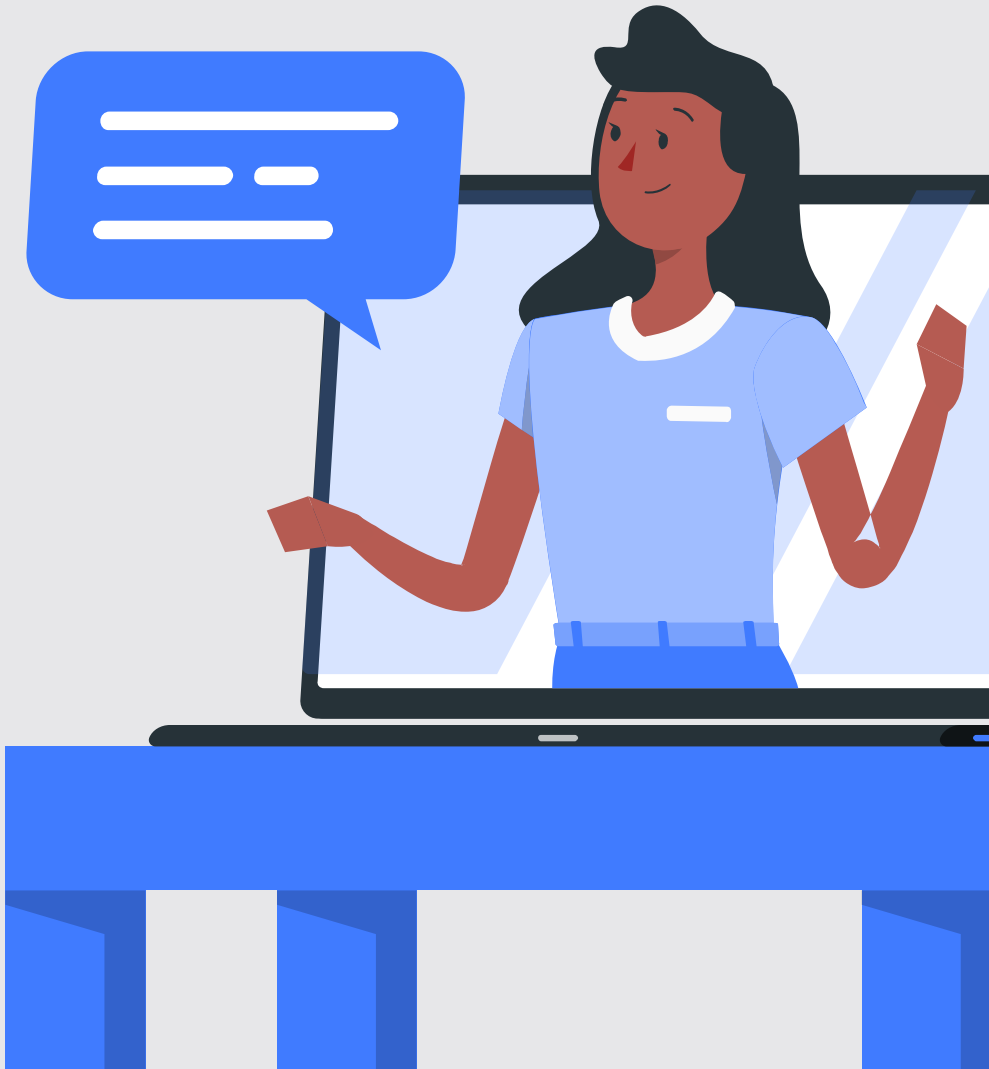
Para este tipo de problemas necesitamos saber manipular los datos de una **forma nueva**.

Los datos ahora van a tener una **estructura**, y muchas veces es necesario determinar cómo vamos a almacenar los datos, **antes de empezar a programar**.

Clase 07: Arreglos

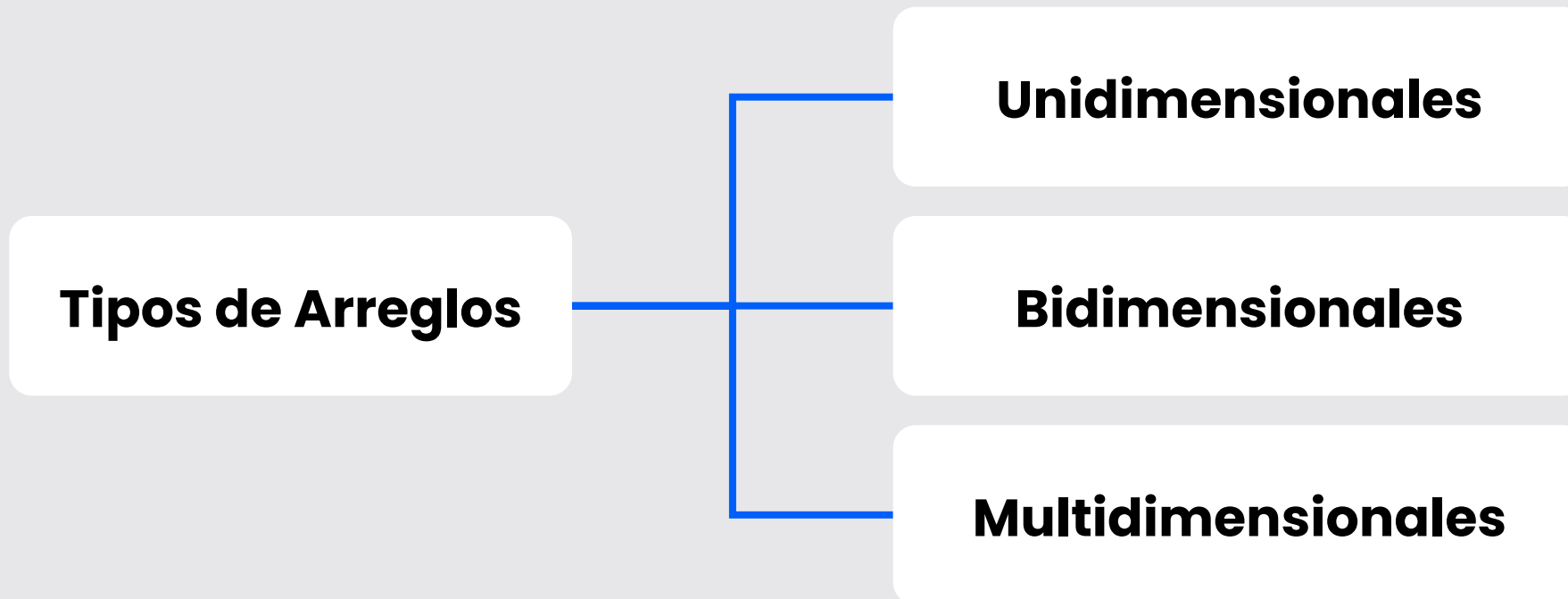
Arreglos

La primera estructura de datos



Arreglos

Un **arreglo** consiste en un número **finito** y **ordenado** de elementos, bajo un **nombre común** para todos ellos.



Arreglos Unidimensionales

También se denominan **vectores**, o **listas** en Python

Tiene **un** solo **índice**

Cada componente del vector se direcciona mediante un nombre seguido del número correspondiente al índice entre paréntesis cuadrados

En Python el primer elemento **siempre** está en la posición **0**

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
1	2	3	4	5	6	7	8	9	10

Trabajando con listas

Para trabajar con listas necesitamos:

- Crear la lista
 - De forma que exista la variable
- Ingresar elementos a la lista
 - Estos son los datos que vamos a procesar para solucionar el problema
- Obtener elementos de la lista
 - Dependiendo del problema, necesitaremos leer los datos para procesarlos, escribirlos, etc.

Un ejemplo

Crear una lista vacía:

```
puntajes = []
```

Crear una lista con elementos:

```
puntajes = [10, 34, 78, 91]
```

Agregando elementos a la lista

Agregando elementos al **final** de la lista:
`puntajes.append(39)`

Si la lista está inicialmente vacía, y agregamos un elemento al final, ¿dónde queda ubicado el elemento?

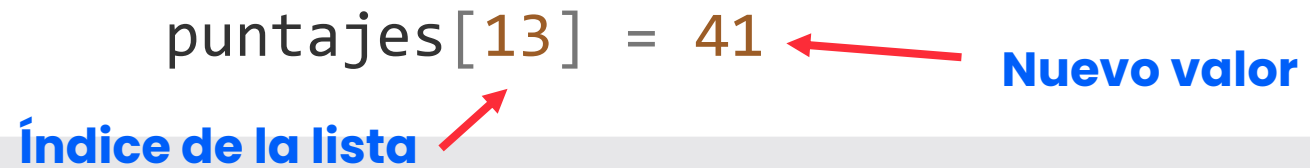
Sobrescribiendo elementos a la lista

Si sabemos que un elementos de la lista existe, podemos reemplazar su valor:

puntajes[13] = 41

Índice de la lista

Nuevo valor

A diagram illustrating list assignment. The code 'puntajes[13] = 41' is shown. A red arrow points from the text 'Índice de la lista' to the index '13'. Another red arrow points from the text 'Nuevo valor' to the value '41'.

Hay que tener cuidado con esto

Si tratamos de sobrescribir un elemento que no existe, vamos a obtener un error:

IndexError: list assignment index out of range

Obteniendo elementos de la lista

Esto se hace usando paréntesis cuadrados, especificando el índice que queremos examinar:

```
# Escribimos el elemento en la posición 3  
print(puntajes[2])
```

```
# Escribimos el elemento en la posición 4  
x = puntaje[3]
```

```
# Agregamos al final el elemento en la primera posición  
puntajes.append(puntajes[0])
```

Un ejemplo

```
puntajes = []  
  
print(puntajes)  
  
for i in range(1, 10, 2):  
    puntajes.append(i)  
    print(i, puntajes)  
  
puntajes[10] = 74
```

```
[]  
1 [1]  
3 [1, 3]  
5 [1, 3, 5]  
7 [1, 3, 5, 7]  
9 [1, 3, 5, 7, 9]
```

**IndexError: list assignment
index out of range**

Total de elementos de una lista

Para verificar la cantidad de elementos de una lista, utilizaremos la función **len()**

```
lista = [1,2,3,4,5,6,7]  
print(len(lista))
```

```
lista.append(8)  
print(len(lista))
```



7



8

Ciclo for en una lista

Para iterar en una lista tenemos dos posibilidades:

- La recorremos por cada elemento
- La recorremos por posición

Por elemento:

```
lista = [1,2,3,4,'hola',6,7]
for i in lista:
    print(i)
```

Por posición:

```
lista = [1,2,3,4,'hola',6,7]
for i in range(len(lista)):
    print(lista[i])
```

Salida:

1
2
3
4
hola
6
7

Quitar elementos de una lista

Para quitar elementos de una lista tenemos tres métodos a nuestra disposición.

Remove: `lista.remove(elemento)`

Pop: `lista.pop(indice)`

Del: `del lista[indice]`

Quitar elementos: Remove

`remove(elemento)` elimina el primer elemento de la lista que se desea eliminar. Si no encuentra ningún elemento para eliminar nos devolverá un `ValueError`.

```
lista = [1,2,3,4,'hola',6,7]
lista.remove('hola')
print(lista)
```



```
[1,2,3,4,6,7]
```

```
lista = [1,2,3,4,'hola',6,7]
lista.remove('chao')
print(lista)
```



```
ValueError: list.remove(x): x
not in list
```

Quitar elementos: Remove

```
lista = [1,2,3,4,'hola',6,'hola']  
lista.remove('hola')  
print(lista)
```



```
[1,2,3,4,6,'hola']
```



Solo eliminamos
el primer hola, el
segundo sigue
en la lista

Quitar elementos: Pop

`pop(indice)` elimina el elemento de la lista en el **índice** (posición) definida. Si no encuentra el índice para eliminar nos devolverá un **IndexError**.

Si `pop()` no recibe ningún argumento, eliminará el último elemento de la lista.

```
lista = [1,2,3,4,'hola',6,'hola']  
lista.pop(1)  
print(lista)
```



```
[1,3,4,'hola',6,'hola']
```


Quitar elementos: Del

Del puede ser utilizado para eliminar un elemento en un índice determinado.

```
lista = [1,2,3,4,'hola',6,'hola']  
del lista[1]  
print(lista)
```

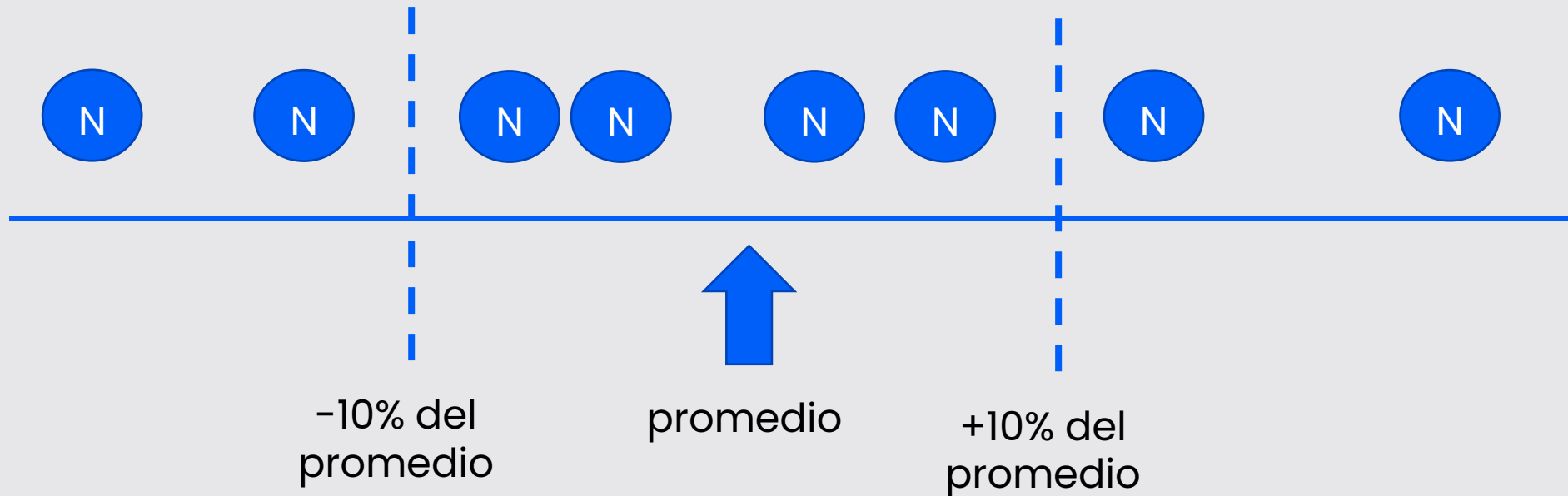


```
[1,3,4,'hola',6,'hola']
```

Otro ejemplo

Hay un archivo que contiene un número en cada línea

Queremos determinar la cantidad de dichos números que están entre $\pm 10\%$ del promedio de los números



```
arch = open("numeros.txt", "r")
# Primero, vamos a leer todos los
# números del archivo
numeros = []
linea = arch.readline().strip()
while linea != "":
    numero = int(linea)
    numeros.append(numero)
    linea = arch.readline().strip()
# Ahora, vamos a imprimir todos
# aquellos que estén dentro de los
# límites. Antes, necesitamos
# conocer esos límites:
prom = calcularPromedio(numeros)
limite1 = prom * 0.9
limite2 = prom * 1.1
```

```
# Ahora, recorremos la lista y si el
# elemento está dentro de los
# límites, lo contamos
cantidad = 0
for x in numeros:
    if x >= limite1 and x <= limite2:
        cantidad = cantidad + 1
print(cantidad)
```

```
# Esta función recibe una lista como  
# parámetro y retorna el valor  
# promedio de los elementos de la lista  
def calcularPromedio(lista):  
    suma = 0  
    # Retorna la cantidad de elementos  
    # presentes en la lista  
    num = len(lista)  
  
    for i in range(0, num):  
        suma = suma + lista[i]  
    promedio = suma / num  
    return promedio
```

Pero podemos hacerlo mejor

Fíjate que este problema tiene 3 etapas bien definidas:

- Obtener los datos
- Calcular
- Comunicar

¿Qué podemos hacer para que el código quede mejor?

Obtener datos:

```
arch = open("numeros.txt", "r")  
numeros = leerNumeros(arch)
```

Calculemos los límites:

```
prom = calcularPromedio(numeros)  
limite1 = prom * 0.9  
limite2 = prom * 1.1
```

Contar

```
cantidad = contarDentroLimites(numeros,  
limite1, limite2)
```

Comunicar

```
print(cantidad)
```

¿Y el resto del código? 1/3

```
# Esta función recibe una lista como  
# parámetro y retorna el valor  
# promedio de los elementos de la lista  
def calcularPromedio(lista):  
    suma = 0  
  
    # Retorna la cantidad de elementos  
    # presentes en la lista  
    num = len(lista)  
    for i in range(0, num):  
        suma = suma + lista[i]  
    promedio = suma / num  
    return promedio
```

¿Y el resto del código? 2/3

```
# Esta función recibe una lista, y retorna la  
# cantidad de elementos de la lista que están  
# dentro de los límites. 0 sea, que sean  
# >= a limite1 y <= a límite 2
```

```
def contarDentroLimites(lista, limite1, limite2):  
    n = 0  
    for x in lista:  
        if x >= limite1 and x <= limite2:  
            n = n + 1  
    return n
```


¿Y el resto del código? 3/3

```
# Esta función lee desde el arch, y  
# retorna los números como una lista  
  
def leerNumeros(arch):  
    # Primero, vamos a leer todos los  
    # números del archivo  
    numeros = []  
    linea = arch.readline().strip()  
    while linea != "":  
        numero = int(linea)  
        numeros.append(numero)  
        linea = arch.readline().strip()  
    return numeros
```

Recapitulando...

¿Qué es una lista?

Un arreglo de tipo **unidimensional** que nos permite almacenar datos de manera estructurada.

Para el ejemplo anterior ¿Por qué era necesario usar una lista?



Trabajo autónomo mínimo

Revisar capítulo 9 (sección 1) libro guía.

Resolver problemas del archivo 'Clase 07 – Ejercicios.pptx'

Resolver ejercicios 1, 2, 3.

Clase 07: Arreglos

¡A practicar!

