Problema1. Organizar sesiones de doblaje

- Para la realización de la practica final de la asignatura de Algoritmos de Optimización he escogido el "Problema 1. Organizar sesiones de doblaje" entre los problemas propuestos.

Algoritmos de optimización - Seminario

- Nombre y Apellidos: Álvaro Alfonso Moya Sanz
- Url: https://github.com/coke869/Algoritmos-Optimizaci-n
- Problema: 1. Sesiones de doblaje

- Descripción del problema:

Se precisa coordinar el doblaje de una película. Los actores del doblaje deben coincidir en las tomas en las que sus personajes aparecen juntos en las diferentes tomas. Los actores de doblaje cobran todos la misma cantidad por cada día que deben desplazarse hasta el estudio de grabación independientemente del número de tomas que se graben. No es posible grabar más de 6 tomas por día. El objetivo es planificar las sesiones por día de manera que el gasto por los servicios de los actores de doblaje sea el menor posible.

- Número de actores: 10
- Número de tomas: 30
- Actores/Tomas: https://bit.ly/36D8IuK
- 1 indica que el actor participa en la toma
- 0 en caso contrario

(*) La respuesta es obligatoria

Carga de datos

In [1]:
```
#Importación de librerías
import pandas as pd
```

```
from pandas import Series, DataFrame
import numpy as np
```

In [2]:
```
#Carga del csv. El csv usado se está disponible en la URL de github. Deberá estar metido en la misma carpeta del noteb
import os
ruta = os.path.join("res" ,"datos_problema_doblaje.csv")
loaded = pd.read_csv(ruta, sep=',', index_col = 0)
loaded.head(3)
```

Out[2]:

	Actor	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9	Unnamed: 10	Unnamed: 11	Unnamed: 12
Toma	1	2	3	4	5	6	7	8	9	10	NaN	Total
1	1	1	1	1	1	0	0	0	0	0	NaN	5
2	0	0	1	1	1	0	0	0	0	0	NaN	3

Transformación de datos.

- Eliminamos la columna 'Toma', ya que no nos sirve.

In [3]:
```
#Con la función drop eliminamos la fila que indiquemos. inplace guarda el cambio en nuestro dataframe.
loaded.drop('Toma', axis=0, inplace = True)
```

- Nombramos las columnas

In [4]:
```
loaded.columns = ['Actor1','Actor2','Actor3','Actor4','Actor5','Actor6','Actor7','Actor8','Actor9','Actor10','Borrar',
loaded.head(3)
```

Out[4]:

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10	Borrar	Total
1	1	1	1	1	1	0	0	0	0	0	NaN	5
2	0	0	1	1	1	0	0	0	0	0	NaN	3
3	0	1	0	0	1	0	1	0	0	0	NaN	3

- Eliminamos la columna 'Borrar'

In [5]:
```
#eliminar columna Borrar
del loaded['Borrar']
display(loaded)
```

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10	Total
1	1	1	1	1	1	0	0	0	0	0	5
2	0	0	1	1	1	0	0	0	0	0	3
3	0	1	0	0	1	0	1	0	0	0	3
4	1	1	0	0	0	0	1	1	0	0	4
5	0	1	0	1	0	0	0	1	0	0	3
6	1	1	0	1	1	0	0	0	0	0	4
7	1	1	0	1	1	0	0	0	0	0	4
8	1	1	0	0	0	1	0	0	0	0	3
9	1	1	0	1	0	0	0	0	0	0	3
10	1	1	0	0	0	1	0	0	1	0	4
11	1	1	1	0	1	0	0	1	0	0	5
12	1	1	1	1	0	1	0	0	0	0	5
13	1	0	0	1	1	0	0	0	0	0	3
14	1	0	1	0	0	1	0	0	0	0	3
15	1	1	0	0	0	0	1	0	0	0	3
16	0	0	0	1	0	0	0	0	0	1	2
17	1	0	1	0	0	0	0	0	0	0	2
18	0	0	1	0	0	1	0	0	0	0	2
19	1	0	1	0	0	0	0	0	0	0	2
20	1	0	1	1	1	0	0	0	0	0	4
21	0	0	0	0	0	1	0	1	0	0	2
22	1	1	1	1	0	0	0	0	0	0	4
23	1	0	1	0	0	0	0	0	0	0	2
24	0	0	1	0	0	1	0	0	0	0	2
25	1	1	0	1	0	0	0	0	0	1	4

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10	Total
26	1	0	1	0	1	0	0	0	1	0	4
27	0	0	0	1	1	0	0	0	0	0	2
28	1	0	0	1	0	0	0	0	0	0	2
29	1	0	0	0	1	1	0	0	0	0	3
30	1	0	0	1	0	0	0	0	0	0	2
TOTAL	22	14	13	15	11	8	3	4	2	2	NaN

Antes de eliminar la fila "Total", que no queremos ya que nos puede distorsionar los cálculos que hagamos en la transformación de los datos, tenemos que analizar lo siguiente:

- Observamos que actores participan en más de 6 tomas y cuales 6 o menos
  - 6 tomas o más: Actores 1,2,3,4,5 y 6
  - Menos de 6 tomas: Actores 7,8,9 y 10

Esta observación nos servirá más adelante para la definición de nuestro algoritmo.

- Elimino la fila 'Total'

In [6]:
```
loaded.drop('TOTAL', axis=0, inplace = True)
display(loaded)
```

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10	Total
1	1	1	1	1	1	0	0	0	0	0	5
2	0	0	1	1	1	0	0	0	0	0	3
3	0	1	0	0	1	0	1	0	0	0	3
4	1	1	0	0	0	0	1	1	0	0	4
5	0	1	0	1	0	0	0	1	0	0	3
6	1	1	0	1	1	0	0	0	0	0	4
7	1	1	0	1	1	0	0	0	0	0	4
8	1	1	0	0	0	1	0	0	0	0	3
9	1	1	0	1	0	0	0	0	0	0	3
10	1	1	0	0	0	1	0	0	1	0	4
11	1	1	1	0	1	0	0	1	0	0	5
12	1	1	1	1	0	1	0	0	0	0	5
13	1	0	0	1	1	0	0	0	0	0	3
14	1	0	1	0	0	1	0	0	0	0	3
15	1	1	0	0	0	0	1	0	0	0	3
16	0	0	0	1	0	0	0	0	0	1	2
17	1	0	1	0	0	0	0	0	0	0	2
18	0	0	1	0	0	1	0	0	0	0	2
19	1	0	1	0	0	0	0	0	0	0	2
20	1	0	1	1	1	0	0	0	0	0	4
21	0	0	0	0	0	1	0	1	0	0	2
22	1	1	1	1	0	0	0	0	0	0	4
23	1	0	1	0	0	0	0	0	0	0	2
24	0	0	1	0	0	1	0	0	0	0	2
25	1	1	0	1	0	0	0	0	0	1	4

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10	Total
26	1	0	1	0	1	0	0	0	1	0	4
27	0	0	0	1	1	0	0	0	0	0	2
28	1	0	0	1	0	0	0	0	0	0	2
29	1	0	0	0	1	1	0	0	0	0	3
30	1	0	0	1	0	0	0	0	0	0	2

Análisis inicial del problema.

Como veremos más adelante en el cálculo del número de posibilidades o combinaciones posibles, nos damos cuenta que computacionalmente es imposible probar todas y cada una de ellas para encontrar la óptima. Esto solo nos deja como opción usar métodos heurísticos para poder encontrar una buena solución en un tiempo razonable.

Como hemos visto en clase no sabemos que técnica heurística será mejor aplicar en este tipo de problema de optimización, por lo que deberemos ir probando hasta encontrar una buena solución.

Pregunta 1

(*)¿Cuantas posibilidades hay sin tener en cuenta las restricciones?

Para calcular cuantas posibilidades hay sin restricciones hay que tener en cuenta que tenemos un problema de permutación sin repetición (n!).

En este problema tenemos que ordenar las 30 tomas de nuestro dataframe, es decir n = 30. Por lo tanto, el total de posibilidades que se pueden dar aplicando la fórmula de permutación sin repetición será:

```
30! = 265,252,859,812,191,058,636,308,480,000
```

Observamos que con un método exacto es imposible computacionalmente recorrer todas las posibilidades para obtener el resultado óptimo.

Pregunta 2

¿Cuantas posibilidades hay teniendo en cuenta todas las restricciones.

Las restricciones de nuestro problema son:

- Los actores deben coincidir en las tomas.
- No es posible grabar más de 6 tomas al día.
- Coste del actor (todos igual) por día que va al estudio a grabar.

La función objetivo será minimizar el gasto total por los servicios de los actores.

- La restricción que nos afecta para el cálculo de todas las posibilidades es que haya un máximo de 6 tomas diarias. Esto hace que tengamos 5 días en los que se graban el máximo permitido de 6 tomas y haya que calcular las posibilidades de cada día y multiplicarlas todas para que nos de el total de posibilidades. Habrá que tener en cuenta que según avancen los días, se podrán elegir solo tomas que no se hayan grabado en ninguno de los días anteriores.

- Para calcular las posibles posibilidades de cada día tenemos la fórmula de combinaciones de "n" elementos tomados en grupos de "r":

  C(n, r) = n! / (r! * (n - r)!)

Por lo tanto, tendremos las siguientes combinaciones posibles para cada día:

- Posibilidades día 1: C(30, 6) = 30! / (6! * (24)!)
- Posibilidades día 2: C(24, 6) = 24! / (6! * (18)!)
- Posibilidades día 3: C(18, 6) = 18! / (6! * (12)!)
- Posibilidades día 4: C(12, 6) = 12! / (6! * (6)!)
- Posibilidades día 5: Quedan ya solo 6 posibles tomas para escoger.

Hay que multiplicar las posibilidades calculadas justo arriba pertenecientes a cada día para calcular cuantas posibilidades hay en total con las restricciones. Igualmente, nos podemos percatar que sigue siendo un numero gigantesco el que nos dará como resultado.

$$Total\ posibilidades = C(30, 6) * C(24, 6) * C(18, 6) * C(12, 6) * 6$$

Pregunta 3

Modelo para el espacio de soluciones
(*) ¿Cual es la estructura de datos que mejor se adapta al problema? Argumentalo.(Es posible que hayas elegido una al principio y veas la necesidad de cambiar, arguentalo)

Para resolver este problema, lo primero que he hecho es cargar la matriz con los datos de las tomas en que participa cada actor. Esta información la he guardado en un dataframe de la librería de pandas. He usado esta estructura de datos ya que se le puede aplicar mediante la librería de numpy funciones muy rápidas y operar fácilmente con las filas y columnas. Considero que esta característica, que sean muy rápidas y eficientes, es muy importante de cara a los cálculos que necesitemos realizar ya que se trata de un volumen grande de operaciones.

Además, necesitare estructuras intermedias/auxiliares para ir calculando el coste y guardándolo. Para esto en principio usare arrays de numpy, diccionarios y variables.

Pregunta 4

Según el modelo para el espacio de soluciones
(*)¿Cual es la función objetivo?

Se quiere minimizar el gasto total por los servicios de los actores de doblaje. Tenemos la restricción de que no se puede grabar más de 6 tomas al día. Por lo tanto, la función objetivo a minimizar seria:

```
minimizar el gasto total = Sumatorio de los costes de los 5 días

minimizar el gasto total = (Cada día de grabación * coste de ese día)
```

Donde el coste de cada día va a depender del nº actores que participen en las tomas que se graban ese día.

```
coste día = actor1+actor2+actor3+actor4+actor5+actor6+actor7+actor8+actor9+actor10

- Cada actor podrá tomar valor un valor entero entre (0,1)
- 0 si no participa y 1 si participa en alguna de las tomas que se graban ese día.
```

La función objetivo se podría expresar como un sumatorio de 1 a 5 del coste de cada día. $$\sum_{día=1}^{5} costeDia = \sum_{día=1}^{5} =actor1+actor2+actor3+actor4+actor5+actor6+actor7+actor8+actor9+actor10 $$

Pregunta 5

(*)¿Es un problema de maximización o minimización?

Se trata de un problema de minimización, donde lo que se pretende es calcular que seis tomas se deben grabar cada día para minimizar el coste total de servicios de los actores de doblaje al finalizar la grabación de las 30 tomas. Hay que tener en cuenta que lo que se quiere es minimizar el gasto total, no solo el gasto de un día en concreto. Es importante observar que la solución optima no tiene por qué pasar por ir calculando la mejor opción de cada paso o de cada día por separado. Debemos ser conscientes que quizás una solución peor de alguno de los días, puede llevarnos a explorar mejores soluciones finales porque en los siguientes días encontraremos otras soluciones que nos optimicen más el resultado final.

Pregunta 6

Diseña un algoritmo para resolver el problema por fuerza bruta

En este apartado voy a desarrollar un algoritmo que resuelve el problema por fuerza bruta. Como hemos visto en la pregunta 1 es inviable computacionalmente explorar todas las soluciones por fuerza bruta. Por esto mismo voy a hacer el algoritmo reduciendo y acotando las entradas para que pueda ser ejecutado y probado.

Como he dicho, aunque el algoritmo se ha definido así para probar su correcto funcionamiento, si se quisiese con las especificaciones del enunciado seria fácilmente extrapolable para ejecutarlo con las 30 tomas y demás restricciones de nuestro problema (tardaría muchísimo tiempo en ejecutarse).

In [7]:
```
#Función para cargar nuestro array limpiando la columna indeseada.
def carga_DF(carga):
    o = carga
    print('')
    print(f' EL tamaño de la matriz es: {len(o)}')
    print('')
    #eliminar columna Borrar
    del o['Total']
```

```
    return o
    #display(ordenada)
```

In [8]:
```
#Usamos esta función para acotar los requisitos y poder tener una ejecución en un tiempo prudencial.
def mat_inicio_prueba(df):
    return df[:10]
```

In [9]:
```
#Esta función sirve para contar cuantos actores participan en nuestras tomas.
def contar_indices_distintos_de_cero(array):
    contador = 0
    for i in range(len(array)):
        if array[i] != 0:
            contador += 1
    return contador
```

In [10]:
```
#Definimos nuestra matriz inicial "i"
def fuerza_bruta(df):
    i = mat_inicio_prueba(df)
```

In [11]:
```
#Esta función la vamos a usar para ir calculando cuantas tomas lleva cada actor ese día.
def suma_columnas_dataframe(df):
    # Calcula la suma de cada columna utilizando el método 'sum()' de pandas
    suma_columnas = df.sum()
    # Convierte los valores de la suma en un array utilizando 'to_numpy()' de pandas
    array_suma = suma_columnas.to_numpy()
    return array_suma
```

In [12]:
```
#Asignamos a otra variable para no modificar la carga inicial.
mat = carga_DF(loaded)
```

```
 EL tamaño de la matriz es: 30
```

In [13]:
```
#Preparamos nuestro dataframe de prueba para el algoritmo de fuerza bruta.
prueba_fuerza_bruta= mat_inicio_prueba(mat)
display(prueba_fuerza_bruta)
```

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10
1	1	1	1	1	1	0	0	0	0	0
2	0	0	1	1	1	0	0	0	0	0
3	0	1	0	0	1	0	1	0	0	0
4	1	1	0	0	0	0	1	1	0	0
5	0	1	0	1	0	0	0	1	0	0
6	1	1	0	1	1	0	0	0	0	0
7	1	1	0	1	1	0	0	0	0	0
8	1	1	0	0	0	1	0	0	0	0
9	1	1	0	1	0	0	0	0	0	0
10	1	1	0	0	0	1	0	0	1	0

In [15]:
```
#Algoritmo de fuerza bruta: Recorre nuestra array para probar todas las posibilidades.
import time

#Inicia el contador de tiempo
inicio = time.time()

#Inicialización de variables.
cont = 0
lista_aux = []
coste_optimo = 99999999 # Le doy un coste muy alto para inicializarlo.
combinacion_optima = []

#Bucle para recorrer todas las posibles soluciones.
for index1, _ in prueba_fuerza_bruta.iterrows():
    #añadir a la lista el indice
    lista_aux.append(index1)
    for index2, _ in prueba_fuerza_bruta.iterrows():
        #añadir a la lista el indice
        lista_aux.append(index2)
        for index3, _ in prueba_fuerza_bruta.iterrows():
            #añadir a la lista el indice
            lista_aux.append(index3)
            for index4, _ in prueba_fuerza_bruta.iterrows():
                #añadir a la lista el indice
```

```
                lista_aux.append(index4)
                for index5, _ in prueba_fuerza_bruta.iterrows():
                    #añadir a la lista el indice
                    lista_aux.append(index5)
                    for index6, _ in prueba_fuerza_bruta.iterrows():
                        #añadir al set el indice
                        lista_aux.append(index6)
                        if len(set(lista_aux))== 6:
                            # Definir los índices concretos de mi secuencia de tomas.
                            indices_concretos = [index1,index2,index3,index4,index5,index6]

                            # Seleccionar las filas con los índices específicos utilizando loc[]
                            nuevo_dataframe = prueba_fuerza_bruta.loc[indices_concretos]

                            #Calculo cuantas tomas graba cada actor en mi secuencia de tomas elegidas.
                            resultado = suma_columnas_dataframe(nuevo_dataframe)

                            #Calculo el coste. Numero de actores que han grabado alguna toma ese dia.
                            coste_dia = contar_indices_distintos_de_cero(resultado)

                            #Si mejora mi solución, guardo la nueva como la óptima.
                            if coste_dia < coste_optimo:
                                coste_optimo = coste_dia
                                combinacion_optima = lista_aux

                            cont = cont +1

                        #Con estas instrucciones, voy eliminando de mi lista los valores para explorar el resto de las
                        lista_aux = lista_aux[:-1]
                    lista_aux = lista_aux[:-1]
                lista_aux = lista_aux[:-1]
            lista_aux = lista_aux[:-1]
        lista_aux = lista_aux[:-1]
    lista_aux = lista_aux[:-1]
print(f'Total de posibilidades calculadas = {cont}')
print(f'La combinacion optima es {combinacion_optima} con un coste total de: {coste_optimo}')

#Detiene el contador de tiempo
fin = time.time()

#Calcula el tiempo de ejecución del ciclo for
tiempo_ejecucion = fin - inicio
```

```
#Imprime el tiempo de ejecución en segundos
print("Tiempo de ejecución:", tiempo_ejecucion, "segundos")
```

```
Total de posibilidades calculadas = 151200
La combinacion optima es ['1', '2', '3', '6', '7', '9'] con un coste total de: 6
Tiempo de ejecución: 258.8416693210602 segundos
```

Después de varias ejecuciones, tarda prácticamente en mismo tiempo y por supuesto al ser un algoritmo exacto da siempre el mismo resultado.

En este apartado he ejecutado el algoritmo de fuerza bruta de un día, reduciendo la matriz de tomas de 30 a 10 tomas. Para el cálculo por fuerza bruta total, seria fácilmente modificable el código del algoritmo que he presentado arriba. Simplemente se tendría que llamar a este mismo algoritmo recursivamente hasta completar los 5 días. Como entrada a la función recursiva, se le tendría que pasar el dataframe, pero eliminando las filas elejidas de las tomas que hemos escogido en el día/días anteriores. Se tendrá que guardar el coste del día 1 para cuando se pase al día 2 con esa secuencia de tomas, se sigan evaluando las soluciones parciales como coste total = coste_sec_dia_1 + coste_sec_dia_2. Esta misma operativa se tendrá que realizar hasta completar los 5 días. Esto se tendría que hacer para todas las combinaciones, no solo para cada resultado óptimo que nos dé en ese día en concreto, ya que lo que buscamos es el óptimo total, por esta razón su ejecución es imposible de realizar computacionalmente. Cuando terminase el algoritmo, tendríamos que evaluar cuál de todas las combinaciones es la óptima. Pero como he comentado anteriormente, computacionalmente es inviable. Pongo un ejemplo de cómo seria:

- Entrada día 1: dataframe de las 30 tomas --> Ejecuto Algoritmo --> Salida: Todas las soluciones parciales. Para cada una de estas:
- Entrada día 2: dataframe de las 24 tomas restantes --> Ejecuto Algoritmo --> Salida: Soluciones parciales 1+2. Para cada una de estas:
- Entrada día 3: dataframe de las 18 tomas restantes --> Ejecuto Algoritmo --> Salida: Soluciones parciales 1+2+3. Para cada una de estas:
- Entrada día 4: dataframe de las 12 tomas restantes --> Ejecuto Algoritmo --> Salida: Soluciones parciales 1+2+3+4. Para cada una de estas:
- Entrada día 5: dataframe de las 6 tomas restantes --> Ejecuto Algoritmo --> Salida: Soluciones parciales 1+2+3+4+5

Pregunta 7

Calcula la complejidad del algoritmo por fuerza bruta

Para calcular la complejidad de un algoritmo, debemos analizar el número de operaciones realizadas en función del tamaño de la entrada (en este caso, el número de tomas del DataFrame).

Dado que tenemos 6 bucles for anidados que recorren el DataFrame de 30 filas, inicialmente, la complejidad sería O(30^6). Sin embargo, debido a que en cada iteración de los bucles se elimina una fila del DataFrame, la complejidad disminuye en cada llamada recursiva.

En total, se realizarían 4 llamadas recursivas, y en cada una de ellas, el DataFrame se reduce en 6 filas que no tenemos que explorar, ya que en cada llamada recursiva se agregan 6 índices a la lista de tomas que vamos eligiendo para nuestra solución.

Entonces, la complejidad se puede calcular como sigue:

```
30^6 + 24^6 + 18^6 + 12^6 + 6^6 --> Se puede aproximar al orden de O(n^6)
```

Sin embargo, es importante destacar que esta es solo una estimación del número de operaciones, y la complejidad exacta puede depender de otros factores, como las operaciones realizadas en cada iteración de los bucles y en las llamadas recursivas.

En términos generales, la complejidad de este algoritmo se puede expresar como una combinación de varios términos, pero se espera que sea bastante alta debido a los 6 bucles anidados y las llamadas recursivas adicionales. El algoritmo tendrá una complejidad exponencial, lo que lo hace muy ineficiente para tamaños grandes de entrada. En general, este tipo de algoritmos con múltiples bucles anidados y llamadas recursivas deben ser evitados si se requiere un buen rendimiento para tamaños de entrada significativos.

Pregunta 8

(*)Diseña un algoritmo que mejore la complejidad del algortimo por fuerza bruta. Argumenta porque crees que mejora el algoritmo por fuerza bruta

En este algoritmo, para encontrar una buena solucion vamos a seguir los siguientes pasos:

- El primer paso es crear una regla voraz como punto de partida. Esta regla voraz consiste en comprobar cuantos actores participan en menos de 6 tomas. De esta forma sabemos que podemos intentar incluirlos en el mismo dia de grabacion, y no nos provocaran costes adicionales otros dias. Si no encontramos ningun actor con menos de 6 tomas, Vamos a coger los n con mayor coincidencias.
- Una vez Tenemos las tomas iniciales con las que vamos a trabajar, vamos a ir añadiendo a ese dia tomas hasta completar el menor coste posible.

In [17]:
```
#Definimos esta función para calcular los actores que les quedan menos de 6 tomas por grabar.
def indices_columnas_suma_menor_seis(df):
    indices_suma_menor_seis = []
```

```
    for columna in df.columns:
        suma_columna = df[columna].sum()
        if suma_columna < 6:
            indices_suma_menor_seis.append(columna)

    return indices_suma_menor_seis
```

In [18]:
```
#Esta función sirve para contar cuantos actores participan en nuestras tomas.
def contar_indices_distintos_de_cero(array):
    contador = 0
    for i in range(len(array)):
        if array[i] != 0:
            contador += 1
    return contador
```

In [19]:
```
#Función que nos va a devolver los actores que podrían grabar todas sus tomas en un único día
def calculo_coste_dia_algMejorado(df):
    #Calculo que actores tienen menos de 6 tomas.
    indices_columnas_suma_menor_seis(df)
    #Llama a la función "indices_columnas_suma_menor_seis" pasando el DataFrame 'df'
    actores_1dia = indices_columnas_suma_menor_seis(df)
    print(actores_1dia)
```

In [20]:
```
def encontrar_valor_minimo_dicc(diccionario):
    # Obtener el valor mínimo del diccionario
    valor_minimo = min(diccionario.values())

    # Lista para almacenar los índices del valor mínimo
    indices_minimos = [clave for clave, valor in diccionario.items() if valor == valor_minimo]

    return indices_minimos
```

In [21]:
```
carga1 = mat
```

Función matriz_inicial_voraz(df,lista_tabu_actores):

In [22]:
```
#Función que me va a devolver mi matriz inicial aplicando una regla.
def matriz_inicial_voraz(df,lista_tabu_actores):
    #Esta funcion me calcula cuales son los actores que podrian ir a grabar todas sus tomas en un dia.
```

```
    actores_1dia = indices_columnas_suma_menor_seis(df)

    # Filtrar los elementos de los actores que no están en la lista tabu (es decir, ya los hemos cogido y no tendran y
    actores_1dia = [elemento for elemento in actores_1dia if elemento not in lista_tabu_actores]

    # Imprimir la lista resultante
    print("")
    print("Actores que pueden grabar todas sus tomas en un dia:",actores_1dia)

    #Selecciona aleatoriamente una actor del DataFrame
    columna_aleatoria = np.random.choice(actores_1dia)

    print("----> Actor seleccionado aleatoriamente al azar:", columna_aleatoria)

    #Añado a mi lista tabú, para que en los siguientes días no se pueda elegir.
    lista_tabu_actores.append(columna_aleatoria)

    #Encuentra los índices de las filas distintos de cero del actor seleccionada
    indices_distintos_de_cero = df.index[df[columna_aleatoria] != 0].tolist()
    print("Tomas en las que participa del actor seleccionado:", indices_distintos_de_cero)
    print("")

    # Creo mi matriz inicial con solo las filas en las que ese actor participa.
    i = df.loc[indices_distintos_de_cero]

    # Eliminar las filas de la matriz de tomas restantes.
    r = df.drop(indices_distintos_de_cero)

    return i,r,lista_tabu_actores
```

Función calcular_minimo_coste_añadir_una_toma():

In [23]:
```
#Esta función calcula cual es el coste mínimo de recorrer todas las tomas restantes.
def calcular_minimo_coste_añadir_una_toma(i,r):
    indice_coste_dict= {}
    for recorrer, _ in r.iterrows():
        #Calculamos si nuestra toma no está ya dentro de las que hemos elegido para ese día.
        if recorrer not in i.index:
            # Utiliza el método 'concat()' para concatenar
            i = pd.concat([i, r.loc[[str(recorrer)]]], axis=0)

            # Llama a la función 'suma_columnas_dataframe' pasando el DataFrame 'df' para saber cuántos actores van es
```

```
            resultado = suma_columnas_dataframe(i)

            #Calculamos el coste que tenemos asociado a ese indice.
            coste_dia = contar_indices_distintos_de_cero(resultado)

            # Guardar el índice de la toma y el coste en el diccionario
            indice_coste_dict[recorrer] = coste_dia

            #Elimino la toma una vez realizo el cálculo, ya que aún no se cual añadiré finalmente.
            i = i.drop(str(recorrer))
    return indice_coste_dict
```

Función añadir_toma(df_i,df_r,dia):

In [24]:

```
#Función que añade tomas a mi matriz inicial calculada anteriormente por una regla.
def añadir_toma(df_i,df_r,dia):

    #Defino mi matriz inicial
    i = df_i

    #Defino mi matriz restante
    r = df_r

    coste_total_dia = 0

    # inicializo un diccionario para almacenar el índice y el coste
    indice_coste_dict = {}

    #Bucle para que hasta que no complete mi día con 6 tomas, siga buscando añadir una toma.
    while len(i.index) < 6:

        #calculo del coste de añadir cada toma.
        indice_coste_dict = calcular_minimo_coste_añadir_una_toma(i,r)

        #Recorro mi diccionario línea por línea para mostrar como afecta al resultado añadir cada toma.
        for clave, valor in indice_coste_dict.items():
            print(f" Si añadimos la toma {clave}: tendria un coste (parcial) de: {valor}")

        # Llamar a la función para encontrar los índices del valor mínimo(puede haber varias tomas)
        indices_minimos = encontrar_valor_minimo_dicc(indice_coste_dict)

        # Imprimir los índices del valor mínimo(puede haber varias tomas
        print("Tomas que si añadimo minimizamos el coste:", indices_minimos)
```

```
        # Comprobar si todos los elementos de indices_minimos están en i
        estan_todos = all(elemento in i.index for elemento in indices_minimos)

        if not estan_todos:
            # Coger aleatoriamente 1 de estos
            toma_aleatoria = np.random.choice(indices_minimos)
            print("Toma aleatoria seleccionada al azar entre las que minimizan el coste:", toma_aleatoria)

            #Hago una validacion antes de incluirlo
            if toma_aleatoria not in i.index:
                #Creo mi nueva matriz con tomas elegidas.
                i = pd.concat([i, r.loc[[str(toma_aleatoria)]]], axis=0)
                #print(f' Tamaño I con: {len(i)}')
                #Elimino de mi matriz restante la toma que he añadido.
                r = r.drop(str(toma_aleatoria))
                #print(f' Tamaño R con : {len(r)}')
        else:
            indice_coste_dict = calcular_minimo_coste_añadir_una_toma(i,r)


    print(f'Se han escogido para el dia {dia} las siguientes tomas:')
    display(i)

    #Llama a la función 'suma_columnas_dataframe' pasando el DataFrame 'df' para saber cuantos actores van ese dia.
    resultado = suma_columnas_dataframe(i)
    print(f'Resultado final del dia:{resultado}')

    #Calculamos el coste que tenemos asociado a ese indice.
    coste_total_dia = contar_indices_distintos_de_cero(resultado)
    print(f'Coste final del dia: {coste_total_dia}')

    return i,r,coste_total_dia
```

Ejecución del algoritmo:

In [27]:
```
#Inicialización de variables.

ordenada = carga1
lista_tabu_actores=[]
coste_total_alg = 0

#Bucle for del día 1 al día 5 incluido.
```

```
for dia in range(1, 6):
    print("************************************:")
    print(f'*************  Día:" {dia} *************')
    print("************************************:")

    #Esta funcion nos va a devolver las matrices i (tomas iniciales), r (tomas restantes) y lista tabu de actores
    i,r,lista_tabu_actores = matriz_inicial_voraz(ordenada,lista_tabu_actores)

    #calcula coste de 1 dia.
    i,r,coste_dia_i = añadir_toma(i,r,dia)
    print(f'El coste del dia {dia} es {coste_dia_i}')

    #Calculo del coste total.
    coste_total_alg = coste_total_alg + coste_dia_i

    print("-----------------------------------------------------------------")
    print("-----------------------------------------------------------------")
    print("")
    print("")

    #Preparamos las matrices para el siguiente dia.
    i=0
    ordenada = r

print(f'El coste total obtenido con este algoritmo es de {coste_total_alg} en los 5 dias de rodaje.')
```

```
**********************************:
*************  Día:" 1 *************
**********************************:

Actores que pueden grabar todas sus tomas en un dia: ['Actor7', 'Actor8', 'Actor9', 'Actor10']
----> Actor seleccionado aleatoriamente al azar: Actor8
Tomas en las que participa del actor seleccionado: ['4', '5', '11', '21']

 Si añadimos la toma 1: tendria un coste (parcial) de: 8
 Si añadimos la toma 2: tendria un coste (parcial) de: 8
 Si añadimos la toma 3: tendria un coste (parcial) de: 8
 Si añadimos la toma 6: tendria un coste (parcial) de: 8
 Si añadimos la toma 7: tendria un coste (parcial) de: 8
 Si añadimos la toma 8: tendria un coste (parcial) de: 8
 Si añadimos la toma 9: tendria un coste (parcial) de: 8
 Si añadimos la toma 10: tendria un coste (parcial) de: 9
 Si añadimos la toma 12: tendria un coste (parcial) de: 8
 Si añadimos la toma 13: tendria un coste (parcial) de: 8
 Si añadimos la toma 14: tendria un coste (parcial) de: 8
 Si añadimos la toma 15: tendria un coste (parcial) de: 8
 Si añadimos la toma 16: tendria un coste (parcial) de: 9
 Si añadimos la toma 17: tendria un coste (parcial) de: 8
 Si añadimos la toma 18: tendria un coste (parcial) de: 8
 Si añadimos la toma 19: tendria un coste (parcial) de: 8
 Si añadimos la toma 20: tendria un coste (parcial) de: 8
 Si añadimos la toma 22: tendria un coste (parcial) de: 8
 Si añadimos la toma 23: tendria un coste (parcial) de: 8
 Si añadimos la toma 24: tendria un coste (parcial) de: 8
 Si añadimos la toma 25: tendria un coste (parcial) de: 9
 Si añadimos la toma 26: tendria un coste (parcial) de: 9
 Si añadimos la toma 27: tendria un coste (parcial) de: 8
 Si añadimos la toma 28: tendria un coste (parcial) de: 8
 Si añadimos la toma 29: tendria un coste (parcial) de: 8
 Si añadimos la toma 30: tendria un coste (parcial) de: 8
Tomas que si añadimo minimizamos el coste: ['1', '2', '3', '6', '7', '8', '9', '12', '13', '14', '15', '17', '18', '1
9', '20', '22', '23', '24', '27', '28', '29', '30']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 6
 Si añadimos la toma 1: tendria un coste (parcial) de: 8
 Si añadimos la toma 2: tendria un coste (parcial) de: 8
 Si añadimos la toma 3: tendria un coste (parcial) de: 8
 Si añadimos la toma 7: tendria un coste (parcial) de: 8
 Si añadimos la toma 8: tendria un coste (parcial) de: 8
 Si añadimos la toma 9: tendria un coste (parcial) de: 8
 Si añadimos la toma 10: tendria un coste (parcial) de: 9
```

```
 Si añadimos la toma 12: tendria un coste (parcial) de: 8
 Si añadimos la toma 13: tendria un coste (parcial) de: 8
 Si añadimos la toma 14: tendria un coste (parcial) de: 8
 Si añadimos la toma 15: tendria un coste (parcial) de: 8
 Si añadimos la toma 16: tendria un coste (parcial) de: 9
 Si añadimos la toma 17: tendria un coste (parcial) de: 8
 Si añadimos la toma 18: tendria un coste (parcial) de: 8
 Si añadimos la toma 19: tendria un coste (parcial) de: 8
 Si añadimos la toma 20: tendria un coste (parcial) de: 8
 Si añadimos la toma 22: tendria un coste (parcial) de: 8
 Si añadimos la toma 23: tendria un coste (parcial) de: 8
 Si añadimos la toma 24: tendria un coste (parcial) de: 8
 Si añadimos la toma 25: tendria un coste (parcial) de: 9
 Si añadimos la toma 26: tendria un coste (parcial) de: 9
 Si añadimos la toma 27: tendria un coste (parcial) de: 8
 Si añadimos la toma 28: tendria un coste (parcial) de: 8
 Si añadimos la toma 29: tendria un coste (parcial) de: 8
 Si añadimos la toma 30: tendria un coste (parcial) de: 8
Tomas que si añadimo minimizamos el coste: ['1', '2', '3', '7', '8', '9', '12', '13', '14', '15', '17', '18', '19', '2
0', '22', '23', '24', '27', '28', '29', '30']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 3
Se han escogido para el dia 1 las siguientes tomas:
```

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10
4	1	1	0	0	0	0	1	1	0	0
5	0	1	0	1	0	0	0	1	0	0
11	1	1	1	0	1	0	0	1	0	0
21	0	0	0	0	0	1	0	1	0	0
6	1	1	0	1	1	0	0	0	0	0
3	0	1	0	0	1	0	1	0	0	0

```
Resultado final del dia:[3 5 1 2 3 1 2 4 0 0]
Coste final del dia: 8
El coste del dia 1 es 8
-----------------------------------------------------------------
-----------------------------------------------------------------


**********************************:
*************  Día:" 2 *************
**********************************:

Actores que pueden grabar todas sus tomas en un dia: ['Actor7', 'Actor9', 'Actor10']
----> Actor seleccionado aleatoriamente al azar: Actor7
Tomas en las que participa del actor seleccionado: ['15']

 Si añadimos la toma 1: tendria un coste (parcial) de: 6
 Si añadimos la toma 2: tendria un coste (parcial) de: 6
 Si añadimos la toma 7: tendria un coste (parcial) de: 5
 Si añadimos la toma 8: tendria un coste (parcial) de: 4
 Si añadimos la toma 9: tendria un coste (parcial) de: 4
 Si añadimos la toma 10: tendria un coste (parcial) de: 5
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
 Si añadimos la toma 13: tendria un coste (parcial) de: 5
 Si añadimos la toma 14: tendria un coste (parcial) de: 5
 Si añadimos la toma 16: tendria un coste (parcial) de: 5
 Si añadimos la toma 17: tendria un coste (parcial) de: 4
 Si añadimos la toma 18: tendria un coste (parcial) de: 5
 Si añadimos la toma 19: tendria un coste (parcial) de: 4
 Si añadimos la toma 20: tendria un coste (parcial) de: 6
 Si añadimos la toma 22: tendria un coste (parcial) de: 5
 Si añadimos la toma 23: tendria un coste (parcial) de: 4
 Si añadimos la toma 24: tendria un coste (parcial) de: 5
 Si añadimos la toma 25: tendria un coste (parcial) de: 5
 Si añadimos la toma 26: tendria un coste (parcial) de: 6
 Si añadimos la toma 27: tendria un coste (parcial) de: 5
 Si añadimos la toma 28: tendria un coste (parcial) de: 4
 Si añadimos la toma 29: tendria un coste (parcial) de: 5
 Si añadimos la toma 30: tendria un coste (parcial) de: 4
Tomas que si añadimo minimizamos el coste: ['8', '9', '17', '19', '23', '28', '30']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 30
 Si añadimos la toma 1: tendria un coste (parcial) de: 6
 Si añadimos la toma 2: tendria un coste (parcial) de: 6
 Si añadimos la toma 7: tendria un coste (parcial) de: 5
 Si añadimos la toma 8: tendria un coste (parcial) de: 5
```

```
 Si añadimos la toma 9: tendria un coste (parcial) de: 4
 Si añadimos la toma 10: tendria un coste (parcial) de: 6
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
 Si añadimos la toma 13: tendria un coste (parcial) de: 5
 Si añadimos la toma 14: tendria un coste (parcial) de: 6
 Si añadimos la toma 16: tendria un coste (parcial) de: 5
 Si añadimos la toma 17: tendria un coste (parcial) de: 5
 Si añadimos la toma 18: tendria un coste (parcial) de: 6
 Si añadimos la toma 19: tendria un coste (parcial) de: 5
 Si añadimos la toma 20: tendria un coste (parcial) de: 6
 Si añadimos la toma 22: tendria un coste (parcial) de: 5
 Si añadimos la toma 23: tendria un coste (parcial) de: 5
 Si añadimos la toma 24: tendria un coste (parcial) de: 6
 Si añadimos la toma 25: tendria un coste (parcial) de: 5
 Si añadimos la toma 26: tendria un coste (parcial) de: 7
 Si añadimos la toma 27: tendria un coste (parcial) de: 5
 Si añadimos la toma 28: tendria un coste (parcial) de: 4
 Si añadimos la toma 29: tendria un coste (parcial) de: 6
Tomas que si añadimo minimizamos el coste: ['9', '28']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 9
 Si añadimos la toma 1: tendria un coste (parcial) de: 6
 Si añadimos la toma 2: tendria un coste (parcial) de: 6
 Si añadimos la toma 7: tendria un coste (parcial) de: 5
 Si añadimos la toma 8: tendria un coste (parcial) de: 5
 Si añadimos la toma 10: tendria un coste (parcial) de: 6
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
 Si añadimos la toma 13: tendria un coste (parcial) de: 5
 Si añadimos la toma 14: tendria un coste (parcial) de: 6
 Si añadimos la toma 16: tendria un coste (parcial) de: 5
 Si añadimos la toma 17: tendria un coste (parcial) de: 5
 Si añadimos la toma 18: tendria un coste (parcial) de: 6
 Si añadimos la toma 19: tendria un coste (parcial) de: 5
 Si añadimos la toma 20: tendria un coste (parcial) de: 6
 Si añadimos la toma 22: tendria un coste (parcial) de: 5
 Si añadimos la toma 23: tendria un coste (parcial) de: 5
 Si añadimos la toma 24: tendria un coste (parcial) de: 6
 Si añadimos la toma 25: tendria un coste (parcial) de: 5
 Si añadimos la toma 26: tendria un coste (parcial) de: 7
 Si añadimos la toma 27: tendria un coste (parcial) de: 5
 Si añadimos la toma 28: tendria un coste (parcial) de: 4
 Si añadimos la toma 29: tendria un coste (parcial) de: 6
Tomas que si añadimo minimizamos el coste: ['28']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 28
 Si añadimos la toma 1: tendria un coste (parcial) de: 6
```

```
 Si añadimos la toma 2: tendria un coste (parcial) de: 6
 Si añadimos la toma 7: tendria un coste (parcial) de: 5
 Si añadimos la toma 8: tendria un coste (parcial) de: 5
 Si añadimos la toma 10: tendria un coste (parcial) de: 6
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
 Si añadimos la toma 13: tendria un coste (parcial) de: 5
 Si añadimos la toma 14: tendria un coste (parcial) de: 6
 Si añadimos la toma 16: tendria un coste (parcial) de: 5
 Si añadimos la toma 17: tendria un coste (parcial) de: 5
 Si añadimos la toma 18: tendria un coste (parcial) de: 6
 Si añadimos la toma 19: tendria un coste (parcial) de: 5
 Si añadimos la toma 20: tendria un coste (parcial) de: 6
 Si añadimos la toma 22: tendria un coste (parcial) de: 5
 Si añadimos la toma 23: tendria un coste (parcial) de: 5
 Si añadimos la toma 24: tendria un coste (parcial) de: 6
 Si añadimos la toma 25: tendria un coste (parcial) de: 5
 Si añadimos la toma 26: tendria un coste (parcial) de: 7
 Si añadimos la toma 27: tendria un coste (parcial) de: 5
 Si añadimos la toma 29: tendria un coste (parcial) de: 6
Tomas que si añadimo minimizamos el coste: ['7', '8', '13', '16', '17', '19', '22', '23', '25', '27']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 16
 Si añadimos la toma 1: tendria un coste (parcial) de: 7
 Si añadimos la toma 2: tendria un coste (parcial) de: 7
 Si añadimos la toma 7: tendria un coste (parcial) de: 6
 Si añadimos la toma 8: tendria un coste (parcial) de: 6
 Si añadimos la toma 10: tendria un coste (parcial) de: 7
 Si añadimos la toma 12: tendria un coste (parcial) de: 7
 Si añadimos la toma 13: tendria un coste (parcial) de: 6
 Si añadimos la toma 14: tendria un coste (parcial) de: 7
 Si añadimos la toma 17: tendria un coste (parcial) de: 6
 Si añadimos la toma 18: tendria un coste (parcial) de: 7
 Si añadimos la toma 19: tendria un coste (parcial) de: 6
 Si añadimos la toma 20: tendria un coste (parcial) de: 7
 Si añadimos la toma 22: tendria un coste (parcial) de: 6
 Si añadimos la toma 23: tendria un coste (parcial) de: 6
 Si añadimos la toma 24: tendria un coste (parcial) de: 7
 Si añadimos la toma 25: tendria un coste (parcial) de: 5
 Si añadimos la toma 26: tendria un coste (parcial) de: 8
 Si añadimos la toma 27: tendria un coste (parcial) de: 6
 Si añadimos la toma 29: tendria un coste (parcial) de: 7
Tomas que si añadimo minimizamos el coste: ['25']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 25
Se han escogido para el dia 2 las siguientes tomas:
```

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10
15	1	1	0	0	0	0	1	0	0	0
30	1	0	0	1	0	0	0	0	0	0
9	1	1	0	1	0	0	0	0	0	0
28	1	0	0	1	0	0	0	0	0	0
16	0	0	0	1	0	0	0	0	0	1
25	1	1	0	1	0	0	0	0	0	1

```
Resultado final del dia:[5 3 0 5 0 0 1 0 0 2]
Coste final del dia: 5
El coste del dia 2 es 5
----------------------------------------------------------------
----------------------------------------------------------------


**********************************:
*************  Día:" 3 *************
**********************************:

Actores que pueden grabar todas sus tomas en un dia: ['Actor9', 'Actor10']
----> Actor seleccionado aleatoriamente al azar: Actor10
Tomas en las que participa del actor seleccionado: []

 Si añadimos la toma 1: tendria un coste (parcial) de: 5
 Si añadimos la toma 2: tendria un coste (parcial) de: 3
 Si añadimos la toma 7: tendria un coste (parcial) de: 4
 Si añadimos la toma 8: tendria un coste (parcial) de: 3
 Si añadimos la toma 10: tendria un coste (parcial) de: 4
 Si añadimos la toma 12: tendria un coste (parcial) de: 5
 Si añadimos la toma 13: tendria un coste (parcial) de: 3
 Si añadimos la toma 14: tendria un coste (parcial) de: 3
 Si añadimos la toma 17: tendria un coste (parcial) de: 2
 Si añadimos la toma 18: tendria un coste (parcial) de: 2
 Si añadimos la toma 19: tendria un coste (parcial) de: 2
 Si añadimos la toma 20: tendria un coste (parcial) de: 4
 Si añadimos la toma 22: tendria un coste (parcial) de: 4
 Si añadimos la toma 23: tendria un coste (parcial) de: 2
 Si añadimos la toma 24: tendria un coste (parcial) de: 2
 Si añadimos la toma 26: tendria un coste (parcial) de: 4
 Si añadimos la toma 27: tendria un coste (parcial) de: 2
 Si añadimos la toma 29: tendria un coste (parcial) de: 3
Tomas que si añadimo minimizamos el coste: ['17', '18', '19', '23', '24', '27']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 27
 Si añadimos la toma 1: tendria un coste (parcial) de: 5
 Si añadimos la toma 2: tendria un coste (parcial) de: 3
 Si añadimos la toma 7: tendria un coste (parcial) de: 4
 Si añadimos la toma 8: tendria un coste (parcial) de: 5
 Si añadimos la toma 10: tendria un coste (parcial) de: 6
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
 Si añadimos la toma 13: tendria un coste (parcial) de: 3
 Si añadimos la toma 14: tendria un coste (parcial) de: 5
 Si añadimos la toma 17: tendria un coste (parcial) de: 4
```

```
 Si añadimos la toma 18: tendria un coste (parcial) de: 4
 Si añadimos la toma 19: tendria un coste (parcial) de: 4
 Si añadimos la toma 20: tendria un coste (parcial) de: 4
 Si añadimos la toma 22: tendria un coste (parcial) de: 5
 Si añadimos la toma 23: tendria un coste (parcial) de: 4
 Si añadimos la toma 24: tendria un coste (parcial) de: 4
 Si añadimos la toma 26: tendria un coste (parcial) de: 5
 Si añadimos la toma 29: tendria un coste (parcial) de: 4
Tomas que si añadimo minimizamos el coste: ['2', '13']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 2
 Si añadimos la toma 1: tendria un coste (parcial) de: 5
 Si añadimos la toma 7: tendria un coste (parcial) de: 5
 Si añadimos la toma 8: tendria un coste (parcial) de: 6
 Si añadimos la toma 10: tendria un coste (parcial) de: 7
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
 Si añadimos la toma 13: tendria un coste (parcial) de: 4
 Si añadimos la toma 14: tendria un coste (parcial) de: 5
 Si añadimos la toma 17: tendria un coste (parcial) de: 4
 Si añadimos la toma 18: tendria un coste (parcial) de: 4
 Si añadimos la toma 19: tendria un coste (parcial) de: 4
 Si añadimos la toma 20: tendria un coste (parcial) de: 4
 Si añadimos la toma 22: tendria un coste (parcial) de: 5
 Si añadimos la toma 23: tendria un coste (parcial) de: 4
 Si añadimos la toma 24: tendria un coste (parcial) de: 4
 Si añadimos la toma 26: tendria un coste (parcial) de: 5
 Si añadimos la toma 29: tendria un coste (parcial) de: 5
Tomas que si añadimo minimizamos el coste: ['13', '17', '18', '19', '20', '23', '24']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 13
 Si añadimos la toma 1: tendria un coste (parcial) de: 5
 Si añadimos la toma 7: tendria un coste (parcial) de: 5
 Si añadimos la toma 8: tendria un coste (parcial) de: 6
 Si añadimos la toma 10: tendria un coste (parcial) de: 7
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
 Si añadimos la toma 14: tendria un coste (parcial) de: 5
 Si añadimos la toma 17: tendria un coste (parcial) de: 4
 Si añadimos la toma 18: tendria un coste (parcial) de: 5
 Si añadimos la toma 19: tendria un coste (parcial) de: 4
 Si añadimos la toma 20: tendria un coste (parcial) de: 4
 Si añadimos la toma 22: tendria un coste (parcial) de: 5
 Si añadimos la toma 23: tendria un coste (parcial) de: 4
 Si añadimos la toma 24: tendria un coste (parcial) de: 5
 Si añadimos la toma 26: tendria un coste (parcial) de: 5
 Si añadimos la toma 29: tendria un coste (parcial) de: 5
Tomas que si añadimo minimizamos el coste: ['17', '19', '20', '23']
```

```
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 23
 Si añadimos la toma 1: tendria un coste (parcial) de: 5
 Si añadimos la toma 7: tendria un coste (parcial) de: 5
 Si añadimos la toma 8: tendria un coste (parcial) de: 6
 Si añadimos la toma 10: tendria un coste (parcial) de: 7
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
 Si añadimos la toma 14: tendria un coste (parcial) de: 5
 Si añadimos la toma 17: tendria un coste (parcial) de: 4
 Si añadimos la toma 18: tendria un coste (parcial) de: 5
 Si añadimos la toma 19: tendria un coste (parcial) de: 4
 Si añadimos la toma 20: tendria un coste (parcial) de: 4
 Si añadimos la toma 22: tendria un coste (parcial) de: 5
 Si añadimos la toma 24: tendria un coste (parcial) de: 5
 Si añadimos la toma 26: tendria un coste (parcial) de: 5
 Si añadimos la toma 29: tendria un coste (parcial) de: 5
Tomas que si añadimo minimizamos el coste: ['17', '19', '20']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 20
 Si añadimos la toma 1: tendria un coste (parcial) de: 5
 Si añadimos la toma 7: tendria un coste (parcial) de: 5
 Si añadimos la toma 8: tendria un coste (parcial) de: 6
 Si añadimos la toma 10: tendria un coste (parcial) de: 7
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
 Si añadimos la toma 14: tendria un coste (parcial) de: 5
 Si añadimos la toma 17: tendria un coste (parcial) de: 4
 Si añadimos la toma 18: tendria un coste (parcial) de: 5
 Si añadimos la toma 19: tendria un coste (parcial) de: 4
 Si añadimos la toma 22: tendria un coste (parcial) de: 5
 Si añadimos la toma 24: tendria un coste (parcial) de: 5
 Si añadimos la toma 26: tendria un coste (parcial) de: 5
 Si añadimos la toma 29: tendria un coste (parcial) de: 5
Tomas que si añadimo minimizamos el coste: ['17', '19']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 17
Se han escogido para el dia 3 las siguientes tomas:
```

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10
27	0	0	0	1	1	0	0	0	0	0
2	0	0	1	1	1	0	0	0	0	0
13	1	0	0	1	1	0	0	0	0	0
23	1	0	1	0	0	0	0	0	0	0
20	1	0	1	1	1	0	0	0	0	0
17	1	0	1	0	0	0	0	0	0	0

```
Resultado final del dia:[4 0 4 4 4 0 0 0 0 0]
Coste final del dia: 4
El coste del dia 3 es 4
------------------------------------------------------------------
------------------------------------------------------------------


**********************************:
*************  Día:" 4 *************
**********************************:

Actores que pueden grabar todas sus tomas en un dia: ['Actor4', 'Actor5', 'Actor9']
----> Actor seleccionado aleatoriamente al azar: Actor5
Tomas en las que participa del actor seleccionado: ['1', '7', '26', '29']

 Si añadimos la toma 8: tendria un coste (parcial) de: 7
 Si añadimos la toma 10: tendria un coste (parcial) de: 7
 Si añadimos la toma 12: tendria un coste (parcial) de: 7
 Si añadimos la toma 14: tendria un coste (parcial) de: 7
 Si añadimos la toma 18: tendria un coste (parcial) de: 7
 Si añadimos la toma 19: tendria un coste (parcial) de: 7
 Si añadimos la toma 22: tendria un coste (parcial) de: 7
 Si añadimos la toma 24: tendria un coste (parcial) de: 7
Tomas que si añadimo minimizamos el coste: ['8', '10', '12', '14', '18', '19', '22', '24']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 12
 Si añadimos la toma 8: tendria un coste (parcial) de: 7
 Si añadimos la toma 10: tendria un coste (parcial) de: 7
 Si añadimos la toma 14: tendria un coste (parcial) de: 7
 Si añadimos la toma 18: tendria un coste (parcial) de: 7
 Si añadimos la toma 19: tendria un coste (parcial) de: 7
 Si añadimos la toma 22: tendria un coste (parcial) de: 7
 Si añadimos la toma 24: tendria un coste (parcial) de: 7
Tomas que si añadimo minimizamos el coste: ['8', '10', '14', '18', '19', '22', '24']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 22
Se han escogido para el dia 4 las siguientes tomas:
```

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10
1	1	1	1	1	1	0	0	0	0	0
7	1	1	0	1	1	0	0	0	0	0
26	1	0	1	0	1	0	0	0	1	0
29	1	0	0	0	1	1	0	0	0	0
12	1	1	1	1	0	1	0	0	0	0
22	1	1	1	1	0	0	0	0	0	0

```
Resultado final del dia:[6 4 4 4 4 2 0 0 1 0]
Coste final del dia: 7
El coste del dia 4 es 7
----------------------------------------------------------------
----------------------------------------------------------------


**********************************:
*************  Día:" 5 *************
**********************************:

Actores que pueden grabar todas sus tomas en un dia: ['Actor1', 'Actor2', 'Actor3', 'Actor4', 'Actor6', 'Actor9']
----> Actor seleccionado aleatoriamente al azar: Actor1
Tomas en las que participa del actor seleccionado: ['8', '10', '14', '19']

 Si añadimos la toma 18: tendria un coste (parcial) de: 5
 Si añadimos la toma 24: tendria un coste (parcial) de: 5
Tomas que si añadimo minimizamos el coste: ['18', '24']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 24
 Si añadimos la toma 18: tendria un coste (parcial) de: 5
Tomas que si añadimo minimizamos el coste: ['18']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 18
Se han escogido para el dia 5 las siguientes tomas:
```

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10
8	1	1	0	0	0	1	0	0	0	0
10	1	1	0	0	0	1	0	0	1	0
14	1	0	1	0	0	1	0	0	0	0
19	1	0	1	0	0	0	0	0	0	0
24	0	0	1	0	0	1	0	0	0	0
18	0	0	1	0	0	1	0	0	0	0

```
Resultado final del dia:[4 2 4 0 0 5 0 0 1 0]
Coste final del dia: 5
El coste del dia 5 es 5
----------------------------------------------------------------
----------------------------------------------------------------


El coste total obtenido con este algoritmo es de 29 en los 5 dias de rodaje.
```

Pregunta 9

(*)Calcula la complejidad del algoritmo

Mi algoritmo consta de:

- un bucle inicial para recorrer los días. Dentro del bucle tiene dos llamadas a otros métodos principales:
  - matriz_inicial_voraz
  - añadir_toma

for dia in range(1, 6): i,r,lista_tabu_actores = matriz_inicial_voraz(ordenada,lista_tabu_actores) i,r,coste_dia_i = añadir_toma(i,r,dia)

El bucle for incrementa en multiplicar por 5, al estar limitado el número de elementos no tendríamos que subir el orden de complejidad, se trataría como una constante multiplicativa. La funcion matriz_inicial_voraz no tiene ningún bucle, su orden de complejidad es constante. La función añadir_toma tiene un bucle while (digamos que tiene m posibles iteraciones) y un bucle for (recorre n elementos de la lista) anidado dentro del while. Por lo tanto, este algoritmo tendría O(m*n). Considerando que n siempre va a ser mayor que m, el peor de los casos

tendríamos O(n^2). Con una constante multiplicativa del for de recorrer todos los días, pero la aproximación de la complejidad de mi algoritmo seria O(n^2).

Pregunta 10

Según el problema (y tenga sentido), diseña un juego de datos de entrada aleatorios

In [31]:
```python
#Función que me calcula una matriz inicial con n tomas que escoge aleatoriamente.
def matriz_aleatoria_inicial(df,n):

    #Verifica si el número n es mayor que el número total de tomas diarias permitidas.
    if n > 6:
        print("El número de tomas iniciales a seleccionar es mayor que el número total diarios permitido")
        return None

    # Seleccionar n tomas aleatorias del DataFrame utilizando la función sample()
    dataframe_aleatorio = df.sample(n=n, random_state=42)  # Puedes cambiar el valor de random_state para obtener sele

    #Asigno a mi matriz i
    i = dataframe_aleatorio

    #Eliminar las filas de la matriz de tomas restantes.
    r = df.drop(dataframe_aleatorio.index)

    return i,r
```

- Simplemente como tota informativa, notar que si usásemos este algoritmo con el parámetro n=6 sería un algoritmo que nos calcularía secuencias de resultados 100% aleatorias.

Pregunta 11

Aplica el algoritmo al juego de datos generado

In [32]:
```python
#Aplicamos nuestra funcion de datos aleatorios.
```

In [33]:
```
#Ejecucion de algoritmo con datos aleatorios de entrada.

#Inicializo variables.
ordenada = carga1
lista_tabu_actores=[]
coste_total_alg = 0

#Indico que me coja para cada dia 3 tomas aleatoriamente.
n=3

#Bucle for del día 1 al día 5 inclusive
for dia in range(1, 6):
    print("************************************:")
    print(f'*************  Día:" {dia} *************')
    print("************************************:")

    i,r = matriz_aleatoria_inicial(ordenada,n)

    #calcula coste de 1 dia.
    i,r,coste_dia_i = añadir_toma(i,r,dia)
    print(f'El coste del dia {dia} es {coste_dia_i}')

    coste_total_alg = coste_total_alg + coste_dia_i

    print("-----------------------------------------------------------------")
    print("-----------------------------------------------------------------")
    print("")
    print("")

    #Preparamos las matrices para el siguiente dia.
    i=0
    ordenada = r

print(f'El coste total obtenido con este algoritmo es: {coste_total_alg} en los 5 dias de rodaje.')
```

```
**********************************:
*************  Día:" 1 *************
**********************************:
 Si añadimos la toma 1: tendria un coste (parcial) de: 7
 Si añadimos la toma 2: tendria un coste (parcial) de: 6
 Si añadimos la toma 3: tendria un coste (parcial) de: 8
 Si añadimos la toma 4: tendria un coste (parcial) de: 8
 Si añadimos la toma 5: tendria un coste (parcial) de: 7
 Si añadimos la toma 6: tendria un coste (parcial) de: 7
 Si añadimos la toma 7: tendria un coste (parcial) de: 7
 Si añadimos la toma 8: tendria un coste (parcial) de: 6
 Si añadimos la toma 9: tendria un coste (parcial) de: 6
 Si añadimos la toma 10: tendria un coste (parcial) de: 7
 Si añadimos la toma 11: tendria un coste (parcial) de: 8
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
 Si añadimos la toma 13: tendria un coste (parcial) de: 6
 Si añadimos la toma 14: tendria un coste (parcial) de: 5
 Si añadimos la toma 15: tendria un coste (parcial) de: 7
 Si añadimos la toma 17: tendria un coste (parcial) de: 5
 Si añadimos la toma 18: tendria un coste (parcial) de: 5
 Si añadimos la toma 19: tendria un coste (parcial) de: 5
 Si añadimos la toma 20: tendria un coste (parcial) de: 6
 Si añadimos la toma 21: tendria un coste (parcial) de: 6
 Si añadimos la toma 22: tendria un coste (parcial) de: 6
 Si añadimos la toma 23: tendria un coste (parcial) de: 5
 Si añadimos la toma 25: tendria un coste (parcial) de: 6
 Si añadimos la toma 26: tendria un coste (parcial) de: 7
 Si añadimos la toma 27: tendria un coste (parcial) de: 6
 Si añadimos la toma 29: tendria un coste (parcial) de: 6
 Si añadimos la toma 30: tendria un coste (parcial) de: 5
Tomas que si añadimo minimizamos el coste: ['14', '17', '18', '19', '23', '30']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 30
 Si añadimos la toma 1: tendria un coste (parcial) de: 7
 Si añadimos la toma 2: tendria un coste (parcial) de: 6
 Si añadimos la toma 3: tendria un coste (parcial) de: 8
 Si añadimos la toma 4: tendria un coste (parcial) de: 8
 Si añadimos la toma 5: tendria un coste (parcial) de: 7
 Si añadimos la toma 6: tendria un coste (parcial) de: 7
 Si añadimos la toma 7: tendria un coste (parcial) de: 7
 Si añadimos la toma 8: tendria un coste (parcial) de: 6
 Si añadimos la toma 9: tendria un coste (parcial) de: 6
 Si añadimos la toma 10: tendria un coste (parcial) de: 7
 Si añadimos la toma 11: tendria un coste (parcial) de: 8
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
```

```
 Si añadimos la toma 13: tendria un coste (parcial) de: 6
 Si añadimos la toma 14: tendria un coste (parcial) de: 5
 Si añadimos la toma 15: tendria un coste (parcial) de: 7
 Si añadimos la toma 17: tendria un coste (parcial) de: 5
 Si añadimos la toma 18: tendria un coste (parcial) de: 5
 Si añadimos la toma 19: tendria un coste (parcial) de: 5
 Si añadimos la toma 20: tendria un coste (parcial) de: 6
 Si añadimos la toma 21: tendria un coste (parcial) de: 6
 Si añadimos la toma 22: tendria un coste (parcial) de: 6
 Si añadimos la toma 23: tendria un coste (parcial) de: 5
 Si añadimos la toma 25: tendria un coste (parcial) de: 6
 Si añadimos la toma 26: tendria un coste (parcial) de: 7
 Si añadimos la toma 27: tendria un coste (parcial) de: 6
 Si añadimos la toma 29: tendria un coste (parcial) de: 6
Tomas que si añadimo minimizamos el coste: ['14', '17', '18', '19', '23']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 17
 Si añadimos la toma 1: tendria un coste (parcial) de: 7
 Si añadimos la toma 2: tendria un coste (parcial) de: 6
 Si añadimos la toma 3: tendria un coste (parcial) de: 8
 Si añadimos la toma 4: tendria un coste (parcial) de: 8
 Si añadimos la toma 5: tendria un coste (parcial) de: 7
 Si añadimos la toma 6: tendria un coste (parcial) de: 7
 Si añadimos la toma 7: tendria un coste (parcial) de: 7
 Si añadimos la toma 8: tendria un coste (parcial) de: 6
 Si añadimos la toma 9: tendria un coste (parcial) de: 6
 Si añadimos la toma 10: tendria un coste (parcial) de: 7
 Si añadimos la toma 11: tendria un coste (parcial) de: 8
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
 Si añadimos la toma 13: tendria un coste (parcial) de: 6
 Si añadimos la toma 14: tendria un coste (parcial) de: 5
 Si añadimos la toma 15: tendria un coste (parcial) de: 7
 Si añadimos la toma 18: tendria un coste (parcial) de: 5
 Si añadimos la toma 19: tendria un coste (parcial) de: 5
 Si añadimos la toma 20: tendria un coste (parcial) de: 6
 Si añadimos la toma 21: tendria un coste (parcial) de: 6
 Si añadimos la toma 22: tendria un coste (parcial) de: 6
 Si añadimos la toma 23: tendria un coste (parcial) de: 5
 Si añadimos la toma 25: tendria un coste (parcial) de: 6
 Si añadimos la toma 26: tendria un coste (parcial) de: 7
 Si añadimos la toma 27: tendria un coste (parcial) de: 6
 Si añadimos la toma 29: tendria un coste (parcial) de: 6
Tomas que si añadimo minimizamos el coste: ['14', '18', '19', '23']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 23
Se han escogido para el dia 1 las siguientes tomas:
```

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10
28	1	0	0	1	0	0	0	0	0	0
16	0	0	0	1	0	0	0	0	0	1
24	0	0	1	0	0	1	0	0	0	0
30	1	0	0	1	0	0	0	0	0	0
17	1	0	1	0	0	0	0	0	0	0
23	1	0	1	0	0	0	0	0	0	0

```
Resultado final del dia:[4 0 3 3 0 1 0 0 0 1]
Coste final del dia: 5
El coste del dia 1 es 5
----------------------------------------------------------------
----------------------------------------------------------------


**********************************:
*************  Día:" 2 *************
**********************************:
 Si añadimos la toma 2: tendria un coste (parcial) de: 5
 Si añadimos la toma 3: tendria un coste (parcial) de: 6
 Si añadimos la toma 4: tendria un coste (parcial) de: 7
 Si añadimos la toma 5: tendria un coste (parcial) de: 6
 Si añadimos la toma 6: tendria un coste (parcial) de: 5
 Si añadimos la toma 7: tendria un coste (parcial) de: 5
 Si añadimos la toma 8: tendria un coste (parcial) de: 6
 Si añadimos la toma 10: tendria un coste (parcial) de: 7
 Si añadimos la toma 11: tendria un coste (parcial) de: 6
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
 Si añadimos la toma 13: tendria un coste (parcial) de: 5
 Si añadimos la toma 14: tendria un coste (parcial) de: 6
 Si añadimos la toma 15: tendria un coste (parcial) de: 6
 Si añadimos la toma 18: tendria un coste (parcial) de: 6
 Si añadimos la toma 20: tendria un coste (parcial) de: 5
 Si añadimos la toma 21: tendria un coste (parcial) de: 7
 Si añadimos la toma 22: tendria un coste (parcial) de: 5
 Si añadimos la toma 25: tendria un coste (parcial) de: 6
 Si añadimos la toma 26: tendria un coste (parcial) de: 6
 Si añadimos la toma 27: tendria un coste (parcial) de: 5
 Si añadimos la toma 29: tendria un coste (parcial) de: 6
Tomas que si añadimo minimizamos el coste: ['2', '6', '7', '13', '20', '22', '27']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 20
 Si añadimos la toma 2: tendria un coste (parcial) de: 5
 Si añadimos la toma 3: tendria un coste (parcial) de: 6
 Si añadimos la toma 4: tendria un coste (parcial) de: 7
 Si añadimos la toma 5: tendria un coste (parcial) de: 6
 Si añadimos la toma 6: tendria un coste (parcial) de: 5
 Si añadimos la toma 7: tendria un coste (parcial) de: 5
 Si añadimos la toma 8: tendria un coste (parcial) de: 6
 Si añadimos la toma 10: tendria un coste (parcial) de: 7
 Si añadimos la toma 11: tendria un coste (parcial) de: 6
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
 Si añadimos la toma 13: tendria un coste (parcial) de: 5
```

```
 Si añadimos la toma 14: tendria un coste (parcial) de: 6
 Si añadimos la toma 15: tendria un coste (parcial) de: 6
 Si añadimos la toma 18: tendria un coste (parcial) de: 6
 Si añadimos la toma 21: tendria un coste (parcial) de: 7
 Si añadimos la toma 22: tendria un coste (parcial) de: 5
 Si añadimos la toma 25: tendria un coste (parcial) de: 6
 Si añadimos la toma 26: tendria un coste (parcial) de: 6
 Si añadimos la toma 27: tendria un coste (parcial) de: 5
 Si añadimos la toma 29: tendria un coste (parcial) de: 6
Tomas que si añadimo minimizamos el coste: ['2', '6', '7', '13', '22', '27']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 13
 Si añadimos la toma 2: tendria un coste (parcial) de: 5
 Si añadimos la toma 3: tendria un coste (parcial) de: 6
 Si añadimos la toma 4: tendria un coste (parcial) de: 7
 Si añadimos la toma 5: tendria un coste (parcial) de: 6
 Si añadimos la toma 6: tendria un coste (parcial) de: 5
 Si añadimos la toma 7: tendria un coste (parcial) de: 5
 Si añadimos la toma 8: tendria un coste (parcial) de: 6
 Si añadimos la toma 10: tendria un coste (parcial) de: 7
 Si añadimos la toma 11: tendria un coste (parcial) de: 6
 Si añadimos la toma 12: tendria un coste (parcial) de: 6
 Si añadimos la toma 14: tendria un coste (parcial) de: 6
 Si añadimos la toma 15: tendria un coste (parcial) de: 6
 Si añadimos la toma 18: tendria un coste (parcial) de: 6
 Si añadimos la toma 21: tendria un coste (parcial) de: 7
 Si añadimos la toma 22: tendria un coste (parcial) de: 5
 Si añadimos la toma 25: tendria un coste (parcial) de: 6
 Si añadimos la toma 26: tendria un coste (parcial) de: 6
 Si añadimos la toma 27: tendria un coste (parcial) de: 5
 Si añadimos la toma 29: tendria un coste (parcial) de: 6
Tomas que si añadimo minimizamos el coste: ['2', '6', '7', '22', '27']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 6
Se han escogido para el dia 2 las siguientes tomas:
```

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10
9	1	1	0	1	0	0	0	0	0	0
19	1	0	1	0	0	0	0	0	0	0
1	1	1	1	1	1	0	0	0	0	0
20	1	0	1	1	1	0	0	0	0	0
13	1	0	0	1	1	0	0	0	0	0
6	1	1	0	1	1	0	0	0	0	0

```
Resultado final del dia:[6 3 3 5 4 0 0 0 0 0]
Coste final del dia: 5
El coste del dia 2 es 5
-----------------------------------------------------------------
-----------------------------------------------------------------


**********************************:
*************  Día:" 3 *************
**********************************:
 Si añadimos la toma 4: tendria un coste (parcial) de: 8
 Si añadimos la toma 5: tendria un coste (parcial) de: 8
 Si añadimos la toma 7: tendria un coste (parcial) de: 7
 Si añadimos la toma 8: tendria un coste (parcial) de: 7
 Si añadimos la toma 10: tendria un coste (parcial) de: 8
 Si añadimos la toma 11: tendria un coste (parcial) de: 8
 Si añadimos la toma 14: tendria un coste (parcial) de: 7
 Si añadimos la toma 15: tendria un coste (parcial) de: 7
 Si añadimos la toma 18: tendria un coste (parcial) de: 7
 Si añadimos la toma 21: tendria un coste (parcial) de: 8
 Si añadimos la toma 22: tendria un coste (parcial) de: 7
 Si añadimos la toma 25: tendria un coste (parcial) de: 8
 Si añadimos la toma 26: tendria un coste (parcial) de: 8
 Si añadimos la toma 27: tendria un coste (parcial) de: 7
 Si añadimos la toma 29: tendria un coste (parcial) de: 7
Tomas que si añadimo minimizamos el coste: ['7', '8', '14', '15', '18', '22', '27', '29']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 7
 Si añadimos la toma 4: tendria un coste (parcial) de: 8
 Si añadimos la toma 5: tendria un coste (parcial) de: 8
 Si añadimos la toma 8: tendria un coste (parcial) de: 7
 Si añadimos la toma 10: tendria un coste (parcial) de: 8
 Si añadimos la toma 11: tendria un coste (parcial) de: 8
 Si añadimos la toma 14: tendria un coste (parcial) de: 7
 Si añadimos la toma 15: tendria un coste (parcial) de: 7
 Si añadimos la toma 18: tendria un coste (parcial) de: 7
 Si añadimos la toma 21: tendria un coste (parcial) de: 8
 Si añadimos la toma 22: tendria un coste (parcial) de: 7
 Si añadimos la toma 25: tendria un coste (parcial) de: 8
 Si añadimos la toma 26: tendria un coste (parcial) de: 8
 Si añadimos la toma 27: tendria un coste (parcial) de: 7
 Si añadimos la toma 29: tendria un coste (parcial) de: 7
Tomas que si añadimo minimizamos el coste: ['8', '14', '15', '18', '22', '27', '29']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 29
 Si añadimos la toma 4: tendria un coste (parcial) de: 8
```

```
 Si añadimos la toma 5: tendria un coste (parcial) de: 8
 Si añadimos la toma 8: tendria un coste (parcial) de: 7
 Si añadimos la toma 10: tendria un coste (parcial) de: 8
 Si añadimos la toma 11: tendria un coste (parcial) de: 8
 Si añadimos la toma 14: tendria un coste (parcial) de: 7
 Si añadimos la toma 15: tendria un coste (parcial) de: 7
 Si añadimos la toma 18: tendria un coste (parcial) de: 7
 Si añadimos la toma 21: tendria un coste (parcial) de: 8
 Si añadimos la toma 22: tendria un coste (parcial) de: 7
 Si añadimos la toma 25: tendria un coste (parcial) de: 8
 Si añadimos la toma 26: tendria un coste (parcial) de: 8
 Si añadimos la toma 27: tendria un coste (parcial) de: 7
Tomas que si añadimo minimizamos el coste: ['8', '14', '15', '18', '22', '27']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 22
Se han escogido para el dia 3 las siguientes tomas:
```

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10
2	0	0	1	1	1	0	0	0	0	0
3	0	1	0	0	1	0	1	0	0	0
12	1	1	1	1	0	1	0	0	0	0
7	1	1	0	1	1	0	0	0	0	0
29	1	0	0	0	1	1	0	0	0	0
22	1	1	1	1	0	0	0	0	0	0

```
Resultado final del dia:[4 4 3 4 4 2 1 0 0 0]
Coste final del dia: 7
El coste del dia 3 es 7
----------------------------------------------------------------
----------------------------------------------------------------


**********************************:
*************  Día:" 4 *************
**********************************:
 Si añadimos la toma 5: tendria un coste (parcial) de: 9
 Si añadimos la toma 8: tendria un coste (parcial) de: 10
 Si añadimos la toma 10: tendria un coste (parcial) de: 10
 Si añadimos la toma 11: tendria un coste (parcial) de: 9
 Si añadimos la toma 14: tendria un coste (parcial) de: 10
 Si añadimos la toma 15: tendria un coste (parcial) de: 9
 Si añadimos la toma 18: tendria un coste (parcial) de: 10
 Si añadimos la toma 21: tendria un coste (parcial) de: 10
 Si añadimos la toma 27: tendria un coste (parcial) de: 9
Tomas que si añadimo minimizamos el coste: ['5', '11', '15', '27']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 11
 Si añadimos la toma 5: tendria un coste (parcial) de: 9
 Si añadimos la toma 8: tendria un coste (parcial) de: 10
 Si añadimos la toma 10: tendria un coste (parcial) de: 10
 Si añadimos la toma 14: tendria un coste (parcial) de: 10
 Si añadimos la toma 15: tendria un coste (parcial) de: 9
 Si añadimos la toma 18: tendria un coste (parcial) de: 10
 Si añadimos la toma 21: tendria un coste (parcial) de: 10
 Si añadimos la toma 27: tendria un coste (parcial) de: 9
Tomas que si añadimo minimizamos el coste: ['5', '15', '27']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 27
 Si añadimos la toma 5: tendria un coste (parcial) de: 9
 Si añadimos la toma 8: tendria un coste (parcial) de: 10
 Si añadimos la toma 10: tendria un coste (parcial) de: 10
 Si añadimos la toma 14: tendria un coste (parcial) de: 10
 Si añadimos la toma 15: tendria un coste (parcial) de: 9
 Si añadimos la toma 18: tendria un coste (parcial) de: 10
 Si añadimos la toma 21: tendria un coste (parcial) de: 10
Tomas que si añadimo minimizamos el coste: ['5', '15']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 15
Se han escogido para el dia 4 las siguientes tomas:
```

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10
26	1	0	1	0	1	0	0	0	1	0
25	1	1	0	1	0	0	0	0	0	1
4	1	1	0	0	0	0	1	1	0	0
11	1	1	1	0	1	0	0	1	0	0
27	0	0	0	1	1	0	0	0	0	0
15	1	1	0	0	0	0	1	0	0	0

```
Resultado final del dia:[5 4 2 2 3 0 2 2 1 1]
Coste final del dia: 9
El coste del dia 4 es 9
----------------------------------------------------------------
----------------------------------------------------------------


**********************************:
*************  Día:" 5 *************
**********************************:
 Si añadimos la toma 10: tendria un coste (parcial) de: 6
 Si añadimos la toma 14: tendria un coste (parcial) de: 6
 Si añadimos la toma 18: tendria un coste (parcial) de: 6
Tomas que si añadimo minimizamos el coste: ['10', '14', '18']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 10
 Si añadimos la toma 14: tendria un coste (parcial) de: 7
 Si añadimos la toma 18: tendria un coste (parcial) de: 7
Tomas que si añadimo minimizamos el coste: ['14', '18']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 18
 Si añadimos la toma 14: tendria un coste (parcial) de: 7
Tomas que si añadimo minimizamos el coste: ['14']
Toma aleatoria seleccionada al azar entre las que minimizan el coste: 14
Se han escogido para el dia 5 las siguientes tomas:
```

	Actor1	Actor2	Actor3	Actor4	Actor5	Actor6	Actor7	Actor8	Actor9	Actor10
5	0	1	0	1	0	0	0	1	0	0
8	1	1	0	0	0	1	0	0	0	0
21	0	0	0	0	0	1	0	1	0	0
10	1	1	0	0	0	1	0	0	1	0
18	0	0	1	0	0	1	0	0	0	0
14	1	0	1	0	0	1	0	0	0	0

```
Resultado final del dia:[3 3 2 1 0 5 0 2 1 0]
Coste final del dia: 7
El coste del dia 5 es 7
------------------------------------------------------------------
------------------------------------------------------------------
```

```
El coste total obtenido con este algoritmo es: 33 en los 5 dias de rodaje.
```

Si evaluamos los dos algoritmos, comprobamos que en promedio mi algoritmo optimizado con la regla inicial (el presentado en la pregunta 8) da muchos mejores resultados comparado con escoger aleatoriamente inicialmente varias tomas. Esto no quiere decir que en alguna ejecución del algoritmo con entradas aleatoria no de buenos resultados o incluso se pueda encontrar una solución mejor, pero es cierto que en media es mucho más robusto el implementado con regla voraz (pregunta 8).

Pregunta 12

Enumera las referencias que has utilizado(si ha sido necesario) para llevar a cabo el trabajo

En general me he apoyado en los apuntes de la asignatura de Python en IA que cursamos hace unos meses y cuando lo he necesitado he realizado búsquedas de funciones y como se realizan distintas funcionalidades en internet.

Pregunta 13

Describe brevemente las lineas de como crees que es posible avanzar en el estudio del problema. Ten en cuenta incluso posibles variaciones del problema y/o variaciones al alza del tamaño

En mi algoritmo se usa una mezcla varios de los métodos o técnicas estudiadas en clase:

- Método GRASP
  - Regla voraz donde se obliga a para escoger las tomas de alguno de los actores que pueden finalizar todas sus tomas en un día.
  - Aleatoriedad: Entre estos actores se escoge aleatoriamente.
- Búsqueda Tabú: El actor que se elige, se va a guardar en una lista tabú, para que no se pueda elegir en ninguno de los siguientes días.
- Búsqueda local: Finalmente para completar las tomas diarias, se usa un método de búsqueda local, donde para cada toma que se quiere añadir se busca cual es la mejor opción.
  - Aleatoriedad: En igualdad de condiciones, se escoge aleatoriamente entre una de las tomas que nos incrementan el coste parcial (de las tomas que llevamos de ese día + esta que añadimos) de forma similar.
  - Técnica de programación dinámica: Se puede decir que se usa también parte de la técnica de programación dinámica, donde se va guardando el resultado de soluciones parciales para elegir cual es la mejor dentro de la búsqueda local.

Para avanzar con el estudio de este problema, se podría seguir investigando si funcionan otras técnicas heurísticas con las que quizás se pueda obtener mejores soluciones. Por ejemplo, varias cosas que se me ocurren podrían ser las siguientes:

- Multiarranque + programación dinámica: Se podría coger en vez de una única solución, partir de varias, por ejemplo, de 3 el primer día. De estas 3 mejores soluciones, el día dos se explorarían de cada una de estas otras 3 y así sucesivamente. El día 5, tendríamos que tener guardadas todas las posibles soluciones que tenemos y con esto decidir cuál es la mejor de todas.
- Centrarnos en la intensificación del problema: Para buscar las mejores soluciones, en vez de ir escogiendo la mejor opción en cada toma que se añade, hacer el cálculo para la mejor opción de los siguientes 2 o 3 tomas (habría que estudiar computacionalmente mediante prueba-error cuanto tardaría la ejecución y si mejoraría o no la solución).
- Con Algoritmos genéticos: Otra opción de mejora que se me ocurre, que quizás sería algo más complicada de implementar, seria guardarnos cuales son las n tomas (puede ser un valor parametrizable) que nos incrementan más el coste de los resultados parciales en los distintos días, e intentar hacer una mutación de ese elemento por otro de los que más incrementan el coste de otro día y comprobar si mejora el resultado final. Para estas mutaciones se podrían realizar X iteraciones y comprobar si en alguna de estas se consigue una mejora en la minimización del coste final.