Práctica 2.1: Divide y vencerás

Algoritmia y optimización

Curso 2024–25

1. Introducción

En esta práctica se debe plantear la resolución de problemas de algoritmia basándose en la estrategia *divide y vencerás*, de manera que los problemas se resuelvan a partir de soluciones a problemas más pequeños.

2. Objetivos

- Practicar el diseño de algoritmos basado en el esquema algorítmico *divide y vencerás*.
- Ganar destreza en la programación de soluciones recursivas.

3. Ejercicio 1

Implementa los siguientes ejemplos, estudiados en las sesiones de teoría:

- Juego del Nim
- Quicksort
- Mergesort
- Contar inversiones

Puedes encontrarlos en las transparencias del "Tema 3: Divide y vencerás".

4. Ejercicio 2

Razona e implementa soluciones recursivas para los siguientes problemas.

4.1. Director deportivo

Descripción: El equipo de tu ciudad acaba de ascender a la categoría reina de su deporte. Sin embargo, los patrocinadores siguen apoyando a los equipos mayoritarios de la liga y se olvidan de vosotros. No tienes dinero para fichar, así que decides crear un algoritmo para el club que permita cerrar la mejor plantilla posible para la próxima temporada. Dispones de un límite de gasto (G) y quieres confeccionar la plantilla mejor valorada según Transfermarkt sin sobrepasar el límite de gasto G (no dispones de límite de jugadores).

Instrucciones: Escribe la función *director_deportivo(valoraciones: arr, fichas: arr, G: int)* que reciba las valoraciones y fichas de los jugadores y calcule el valor máximo total posible de la plantilla sin sobrepasar el límite de gasto G. Por ejemplo, dados los siguientes jugadores y jugadoras, y con un límite de gasto G de 3000 eur, la mejor plantilla tendría una valoración total de 14.

Julia Valoración: 6 Ficha: 950 eur	Pedro Valoración: 1 Ficha: 2400 eur	Alejandro Valoración: 3 Ficha: 500 eur	Rosa Valoración: 8 Ficha: 2000 eur

4.2. Gira de conciertos

Descripción: Después de tus estudios musicales creas un grupo de jazz que empieza a tener cierto éxito. Un promotor te contacta para hacer una gira por España con algunas condiciones. Tienes un número límite de conciertos en la gira (C), además después de cada concierto debéis viajar, descansar y preparar nuevo repertorio por lo que se os indicará cuántas sesiones de un día entre actuaciones como mínimo debéis reservar (R). Para cada fecha disponible, tenéis la cantidad de entradas vendidas.

Instrucciones: Escribe la función *planear_gira(C: int, R: int, entradas_por_dia: arr)* que recibe la cantidad máxima de conciertos en la gira C, los días que reservar entre actuaciones para preparar nuevo repertorio y ensayar R y un array que representa las entradas vendidas por concierto. La función devuelve el número máximo de entradas. Por ejemplo, teniendo un límite de conciertos $C = 3$, y obligados a descansar $R = 3$. La cantidad de entradas vendidas serían 15000.

3000	6000	7000	8000	9000

4.3. Viaje a Marte

Descripción: Imagina que eres el jefe de logística de una agencia espacial que está planeando una misión a Marte. Necesitas cargar una nave espacial con suministros. Dispones de una variedad de artículos, cada uno con un volumen y un coste asociados. Tu objetivo

es llenar **completamente** la bodega de carga de la nave espacial con S metros cúbicos de suministros minimizando el coste total de los mismos.

Instrucciones: Diseña la función *viajar(costes: arr, volumenes: arr, S: int)* que recibe una lista con los costes de los suministros, otra lista con los volúmenes asociados y el espacio máximo de la bodega que debe completarse. La función debe devolver el coste de los suministros de la misión. Por ejemplo, dados los suministros de la tabla y un S máximo de 10 m^3, el coste mínimo de la misión serían 8 millones de euros (combustible, alimentos y herramientas).

Suministro	**Volumen (m^3)**	**Coste (millones eur)**
Alimentos	4	3
Agua	3	2
Oxígeno	2	5
Combustible	5	4
Herramientas	1	1

4.4. Reforestación

Descripción: En un esfuerzo por combatir la deforestación, una organización ecológica ha desarrollado un innovador dron de plantación de semillas. Este dron debe sobrevolar un terreno complejo el cual se representa como una mapa en el que:

- El 0 representa un espacio aéreo libre donde el dron puede volar.
- El 1 representa obstáculos como árboles, montañas o zonas restringidas que el dron debe evitar.

El dron empieza su recorrido en la parte superior-derecha del mapa y sólo puede moverse hacia el sur, el oeste y diagonal suroeste. Nuestro objetivo es llegar a la esquina inferior-izquierda de la zona de operación. Debemos ahorrar la máxima batería posible, teniendo en cuenta que cuantos más pasos demos hasta llegar a nuestro objetivo, más batería estamos gastando.

Instrucciones: Crea la función *plantar(mapa: matrix)* que recibe el mapa de la zona a sobrevolar y devuelve el número mínimo de pasos para llegar a plantar nuestra semilla. Por ejemplo, dado el siguiente mapa, el camino para llegar al objetivo y ahorrar el máximo de batería constaría en cambiar 4 veces de espacio aéreo.

0	0	0	0 (inicio)
0	1	1	0
0 (objetivo)	1	0	0

4.5. Salvar a la princesa

Descripción: Han capturado a la princesa (P) y se encuentra en la parte inferior-derecha de la mazmorra. La mazmorra tiene una distribución de M x N. Nuestro valiente caballero (K) ha sido envenenado y se encuentra en la parte superior-izquierda de la mazmorra, teniendo como objetivo salvar a la princesa.

El caballero tiene una vitalidad inicial dada por un valor entero. Si en algún momento desciende a 0... R.I.P.

Algunas habitaciones de la mazmorra se encuentran custodiadas por enemigos que nos harán perder vitalidad (representado por número negativos). También encontramos habitaciones en las que no perderemos vitalidad (representadas por 0) y otras en las que podremos descansar, recuperando vitalidad (números positivos). Para alcanzar más rápido a la princesa, el caballero decide moverse hacia la **derecha** y hacia **abajo**.

Instrucciones: Ayuda a rescatar a la princesa escribiendo la función *salvar_princesa(mazmorra: matrix[M][N])* que recibe el mapa de la mazmorra como una matriz de números enteros y devuelve la vida mínima con la que tiene que entrar el caballero K a la mazmorra para salvar a la princesa P. Por ejemplo, dada la siguiente mazmorra, la vida inicial del caballero debe ser de 7 (seguirá el camino: `DERECHA->DERECHA->ABAJO->ABAJO`).

-2 (K)	-3	3
-5	-10	1
10	30	-5 (P)

Notas:

- El caballero no tiene vida máxima
- Cualquier habitación puede contener peligros o zonas de descanso, inclusive la primera y última.