Trabajo práctico 5 - Algoritmos greedy

Programación 3 - TUDAI

Ejercicio 1
Cambio de monedas. Dado un conjunto *C* de *N* tipos de monedas con un número ilimitado de ejemplares de cada tipo, se requiere formar, si se puede, una cantidad *M* empleando el mínimo número de ellas. Por ejemplo, un cajero automático dispone de billetes de distintos valores: 100$, 25$, 10$, 5$ y 1$, si se tiene que pagar 289$, la mejor solución consiste en dar 10 billetes: 2 de 100$, 3 de 25$, 1 de 10$ y 4 de 1$.

Ejercicio 2
Problema de la Mochila. Se tienen n objetos y una mochila. Para *i* = 1,2,..*n*, el objeto *i* tiene un peso positivo pi y un valor positivo vi. La mochila puede llevar un peso que no sobrepase P. El objetivo es llenar la mochila de tal manera que se maximice el valor de los objetos transportados, respetando la limitación de capacidad impuesta. Los objetos pueden ser fraccionados, si una fracción xi ($0 \leq xi \leq 1$) del objeto i es ubicada en la mochila contribuye en xipi al peso total de la mochila y en xivi al valor de la carga. Formalmente, el problema puede ser establecido como.

Maximizar $i = \sum_{1}^{n} xivi$, con la restricción $i = \sum_{1}^{n} xipi \leq P$ donde vi>0, pi>0 y $0 \leq xi \leq 1$ para $1 \leq i \leq n$.

Por ejemplo, para la instancia en n = 3 y P = 20
(v1, v2, v3) = (25, 24, 15)
(p1, p2, p3) = (18, 15, 10)

Algunas soluciones posibles son:

(x1, x2, x3)	**xipi**	**xivi**
(½, ⅓, ¼)	16.5	24.25
(1, 2/15, 0)	20	28.2
(0, ⅔, 1)	20	31
(0, 1, ½)	20	31.5

Puede observarse que (0, 1, ½) produce el mayor beneficio.

Ejercicio 3
Maximizar el número de actividades compatibles. Se tienen n actividades que necesitan utilizar un recurso, tal como una sala de conferencias, en exclusión mutua. Cada actividad i tiene asociado un tiempo de comienzo ci y un tiempo de finalización fi de utilización del recurso, con ci < fi. Si la actividad i es seleccionada se llevará a cabo durante el intervalo [ci, fi). Las actividades i y j son compatibles si los intervalos [ci, fi) y [cj, fj) no se superponen (es decir, ci > fj o cj > fi). El problema consiste en encontrar la cantidad máxima de actividades compatibles entre sí.

Ejercicio 4
Algoritmo de Dijkstra. Dado un grafo con pesos no negativos, implemente el algoritmo de Dijkstra para determinar el array de distancias y de predecesores en el camino más corto, desde un vértice orígen dado como parámetro hacia el resto de los vértices. Una vez realizado, imprima el camino más corto que se debe seguir desde el orígen hacia cada vértice.

Ejercicio 5
Desde un cierto conjunto grande de ciudades del interior de una provincia, se desean transportar cereales hasta alguno de los 3 puertos pertenecientes al litoral de la provincia. Se pretende efectuar el transporte total con mínimo costo sabiendo que el flete es más caro cuanto más distancia tiene que recorrer. Dé un algoritmo que resuelva este problema, devolviendo para cada ciudad el camino que debería recorrer hacia el puerto de menor costo.

Ejercicio 6
Problema del viajante. Dado un grafo ponderado de ciudades conectadas todas con todas, implemente un algoritmo que brinde una aproximación al problema del viajante.

Ejercicio 7
Atrapando leones. Dado un arreglo donde en cada posición se encuentra un cazador o un león, queremos capturar la mayor cantidad de leones sabiendo que:

- Un cazador solo puede atrapar un león,
- Los cazadores sólo pueden capturar leones que estén a menos de K pasos de su posición.

Ejercicio 8
Armando CDs. Dado un conjunto de archivos de canciones, donde cada uno tiene la información de nombre, género, duración del tema, y tamaño en kilobytes, se desea grabar un disco CD (que tiene una capacidad máxima de M kilobytes) de modo tal de:

- Variante A: Maximizar la capacidad ocupada del disco CD.
- Variante B: Maximizar la cantidad de canciones que se pueden grabar en el CD.

Para ambas variantes se quiere, además, que el CD no contenga más de 3 canciones de un mismo género.

Ejercicio 9
Coloreo de un grafo. Dado un grafo se desea colorear cada uno de sus vértices utilizando la menor cantidad posible de colores totales, sabiendo que dos vértices adyacentes no podrán utilizar el mismo color.