Práctica 3: Las cuatro operaciones
Algoritmos Voraces y Programación Dinámica.

Dpto. Ciencias de la Computación e Inteligencia Artificial
E.T.S. de Ingenierías Informática y de Telecomunicación
Universidad de Granada

DECSAI

Algorítmica

Grado en Ingeniería Informática D

Índice de contenido

1. Introducción

Los objetivos de este guión de prácticas son los siguientes:

1. Entender la técnica de diseño de algoritmos Voraces y Programación Dinámica.
2. Entender que siempre no podemos llegar a la solución óptima con un algoritmo voraz.
3. Entender que siempre que se pueda aplicar Programación Dinámica la solución obtenida será la óptima

2. Suma hasta un número

2.1. Descripción del Problema

Dado un conjunto de números **S** y un número **M**, el objetivo es obtener el subconjunto de **S** que sume **M**. En caso de que no exista dar la suma inferior más cercana.

Por ejemplo:

S={2,3,4,1} y M = 8

Una solución es 3+4+1

2.2. Técnica Voraz

Obtener una solución de este problema usando la técnica Voraz. Para ello el alumno además de dar la implementación C++ debe aportar en el fichero documentacion.pdf los siguientes puntos:

1. Datos del Problema
2. Representación de la Solución
3. Formulación matemática

Con respecto al diseño de la solución, deberá indicar:

- Candidatos
- Función Solución
- Función de Selección
- Función Factible
- Añadir un objeto a la solución
- Función Objetivo

Establecer si el criterio de selección obtiene siempre la solución óptima si existe. En caso de que no sea así buscar un contraejemplo donde no se cumple.

2.2.1. Programa Voraz

El programa Voraz se ejecuta desde la linea de órdenes de la siguiente forma:

prompt%> voraz_maximo datos/conjunto.txt 10

Los parámetros de este programa son:

- Nombre del fichero con el conjunto de números **S**
- Numero **M** que queremos obtener usando la suma de elementos de **S**.

2.2.2. Eficiencia

El alumno hará un estudio de la eficiencia teórica. Este estudio se detallará en el fichero documentacion.pdf.

2.2.3. Dificultad

Añadir en documentacion.pdf el nivel de dificultad de este ejercicio en un rango [0,10], siendo 0: muy fácil y 10 muy difícil.

2.3. Programación Dinámica

Una vez que hemos asegurado que se cumple el principio de optimalidad de Bellman, podemos buscar una solución con la técnica de Programación Dinámica. Para poder obtener una solución con la técnica de diseño de Programación Dinámica debemos :

1. Obtener una ecuación recurrente del problema.
2. Definir la estrategia de aplicación de la fórmula:
    - Tablas utilizadas por el algoritmos. Cuál va a ser el numero de filas y columnas de nuestra tabla de subproblemas
    - Orden y forma de rellenarla.
3. Especificar como se reconstruye la solución final a partir de los valores de las tablas.

2.3.1. Programa PD

El programa PD se ejecuta desde la linea de órdenes de la siguiente forma:

prompt%> PD_maximo datos/conjunto.txt 10

Los parámetros de este programa son:

- Nombre del fichero con el conjunto de números **S**
- Numero **M** que queremos obtener usando la suma de elementos de **S**.

2.3.2. Eficiencia

Hacer un estudio de la eficiencia teórica. Este apartado se detallará el fichero documentacion.pdf.

2.3.3. Dificultad

Añadir en documentacion.pdf el nivel de dificultad de este ejercicio en un rango [0,10], siendo 0: muy fácil y 10 muy difícil.

3. Operaciones Básicas

Este problema consiste en dado un conjunto de números **S** y un número **M** el objetivo es

obtener el valor **M** con un subconjunto de **S** en el que las operaciones que se pueden realizar son: +,-,* y /. La división si no tiene un resto 0 no se puede aplicar.

Por ejemplo:

Supongamos que **S**={10,11,2,100,90,80,30.40,21} y que **M**= 3

Una solución sería 10+11+2+100-90-30

3.1. Programación Dinámica

Una vez que hemos asegurado que se cumple el principio de optimalidad de Bellman, podemos buscar una solución con la técnica de Programación Dinámica. Para poder obtener una solución con la técnica de diseño de Programación Dinámica debemos :

1. Obtener una ecuación recurrente del problema.
2. Definir la estrategia de aplicación de la fórmula:
3. Tablas utilizadas por el algoritmos. Cuál va a ser el numero de filas y columnas de nuestra tabla de subproblemas
4. Orden y forma de rellenarla.
5. Especificar como se reconstruye la solución final a partir de los valores de las tablas.

3.1.1. Programa PD

El programa PD se ejecuta desde la linea de órdenes de la siguiente forma:

prompt%> PD_operaciones datos/numeros1_1.txt 3

Los parámetros de este programa son:

- Nombre del fichero con el conjunto de números **S**
- Numero **M** que queremos obtener, operando con +,-,*,/ un subconjunto de elementos de **S**

Un ejemplo de ejecución con PD sería:

```
***********
Numero a alcanzar :3

Números: 100    90      80      33      11      10      2
*********************
Todas las soluciones:
11-10+2
100+90-80/11/10+2
90-80/10+2
100-90/10+2
100+90-80+33/11-10
33/11
```

En este caso el programa nos está dando todas las soluciones posibles. Hay que interpretar los resultados leyendo, sin prioridad de operación, de izquierda a derecha. Por ejemplo:

100+90-80/11/10+2 es equivalente a haber priorizado como:

((((100+90)-80)/11)/10)+2

POSIBILIDADES:Se puede generar una versión de PD_cifras que :

1. Obtenga solamente una solución. Si has percibido muy difícil el ejercicio.
2. Obtenga todas las soluciones posibles. Si lo has percibido muy fácil. En este caso se

puede extender el ejercicio para que de las soluciones con los paréntesis para establecer la prioridad de las operaciones.

3.1.2. Eficiencia

Hacer un estudio de la eficiencia teórica. Este apartado se detallará el fichero documentacion.pdf. Para hacer el estudio la eficiencia vamos a a realizarla sobre la versión que obtiene una sola solución.

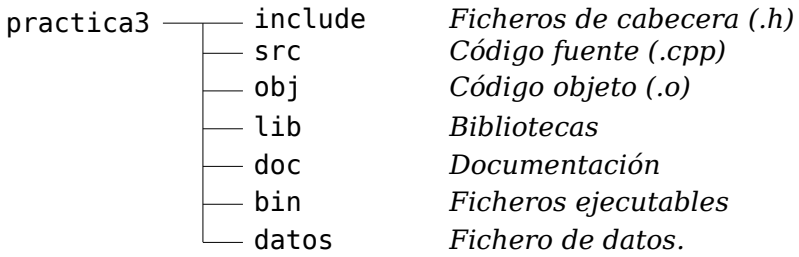
3.1.3. Dificultad

Añadir en documentacion.pdf el nivel de dificultad de este ejercicio en un rango [0,10], siendo 0: muy fácil y 10 muy difícil.

4. Práctica a entregar

El alumno deberá empaquetar todos los archivos relacionados en el proyecto en un archivo con nombre "practica3.*tgz*" y entregarlo antes de la fecha que se publicará en la página web de la asignatura. Tenga en cuenta que no se incluirán ficheros objeto, ni ejecutables, ni la carpeta datos. Es recomendable que haga una "limpieza" para eliminar los archivos temporales o que se puedan generar a partir de los fuentes.

El alumno debe incluir el archivo *Makefile* para realizar la compilación. Tenga en cuenta que los archivos deben estar distribuidos en directorios*:*



Para realizar la entrega, en primer lugar, realice la limpieza de archivos que no se incluirán en ella, y sitúese en la carpeta superior (en el mismo nivel de la carpeta "*practica3*") para ejecutar:

```
prompt% tar zcv practica3.tgz practica3
```

tras lo cual, dispondrá de un nuevo archivo practica3.tgz que contiene la carpeta practica3 así como todas las carpetas y archivos que cuelgan de ella.