

## RAMIFICACIÓN Y PODA

### Practica 5: Ejercicio 8 de la hoja

El código propio genera 3 ficheros de datos aleatorios para el **input**. Estos se representan con el nombre RyP\_in\_**x**.txt, siendo **x** la variable que distingue los 3 ficheros. En el cpp entregado, el programa genera 100 cuadros por cada fichero (funciona más ágilmente) pero, si se queda corto al gusto del profesor y/o de la corrección, es fácilmente modificable en el código como los ratios de valores de los cuadros y la longitud de la pared para la demostración en vivo.

Para el **output** se lee los 3 ficheros input secuencialmente y se generan tantos ficheros de salida con la nomenclatura RyP\_out\_**x**\_cota\_**y**, donde **x** representa el input utilizado e **y** la cota. Hay que tener en cuenta que el programa genera distintos ficheros en función de la cota, es decir, en total (como hay 2 cotas) serian 6 ficheros de output distintos. Cada uno de estos ficheros contiene el prestigio mejor obtenido, la **L** de la pared y los cuadros que componen esta. Y cada uno de los cuadros es descrito con su prestigio, longitud, proporción *prestigio/longitud* y si está colgado en horizontal o vertical. Si el profesor se pregunta dónde están los demás datos exigidos por el enunciado estos aparecen por consola (descrito más abajo en la memoria).

Al leer el input se **ordena de mayor a menor** todos los cuadros en función de proporción  $P_i/L_i$ , donde  $P_i$  indica el prestigio del cuadro  $i$  y  $L_i$  indica la longitud menor del cuadro  $i$ .

**Es solución** toda aquella combinación de cuadros que cumpla que la suma de sus lados elegidos sume exactamente **L** (la longitud de la pared).

Una combinación **es factible** si al sumar la longitud de los cuadros que se lleva y el lado mas grande de los cuadros restantes, la suma es mayor o igual que **L**. En este ejercicio en particular, utilizamos la factibilidad al principio de la exploración porque puede que la combinación de cuadros nunca pueda satisfacer el enunciado.

En cada **nivel del árbol de exploración** decidimos si colgamos el cuadro de mayor proporción en vertical, en horizontal o directamente no colgarlo.

En mi solución he utilizado **2 cotas optimistas**. La primera menos costosa y precisa, consiste en simplemente ir sumando los cuadros en orden de la proporción con su lado menor hasta que llegamos a un cuadro donde su **L** sea mayor o igual que la **L** restante de la pared. Si es mayor, partimos la **L** y el prestigio del cuadro en función de la **L** necesitada para completar la pared. La segunda cota es más costosa y precisa: operamos igual que en la anterior, pero al llegar a un cuadro que sobrepasa la longitud restante se le ignora y se explora toda la lista de cuadros hasta el final o hasta que la

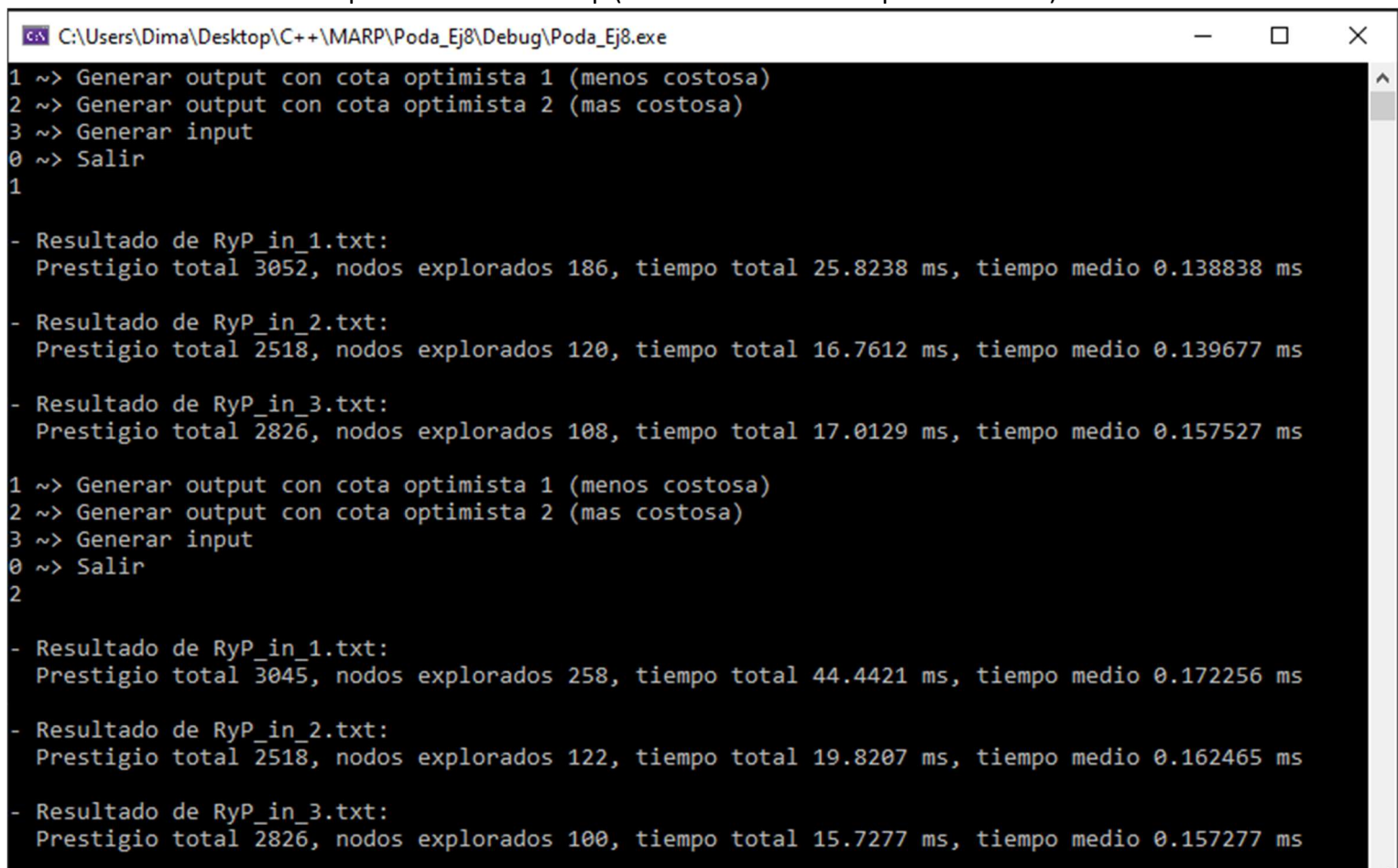
longitud restante sea 0. También me gustaría hablar de una 3ª cota que tenía en consideración, operaba igual que la segunda, pero si al llegar al final de la lista de cuadros nos encontrábamos que la L restante no es 0 partíamos el cuadro con la L mas pequeña de toda la lista y lo restábamos a la L restante. Lo acabe descartando porque la 2ª cota (aparte de ser más sencilla en muchos sentidos) daba mejores resultados que la otra. De todos modos, si el profesor esta interesado en ver el funcionamiento de dicha cota en la demostración en vivo, es fácilmente posible porque, aunque puesto como comentario, el código está escrito y perfectamente funcional.

En este ejercicio no tenemos cota pesimista ya que puede que no tengamos resultado para una combinación de cuadros (visto en la factibilidad).

En la consola de ejecución vemos 4 opciones de operación: generar ficheros output para la cota 1, generar ficheros para la cota 2, generar nuevos ficheros input (importante si se ha modificado alguna variable de generación de datos o si directamente estos ficheros no existen) y la salida de la aplicación. Para la salida de cada una de las cotas obtenemos, aparte de los ficheros output, datos extra en la consola. Estos datos extra son: prestigio total conseguido, nodos explorados, tiempo total y medio. Si una combinación de cuadros no es factible se nos notificará por consola y, obviamente, no se generará ningún fichero output.

En el enunciado de la practica también se nos pide conclusiones personales obtenidas, pero no se exactamente a que se refiere esto. Lo único que puedo decir es que programando este ejercicio por fin he podido entender mejor RyP y perder el miedo por esta.

En la siguiente captura de pantalla se enseña los resultados obtenidos en consola con los ficheros input enviados en el zip (también están los output obtenidos):



```
C:\Users\Dima\Desktop\C++\MARP\Poda_Ej8\Debug\Poda_Ej8.exe
1 ~> Generar output con cota optimista 1 (menos costosa)
2 ~> Generar output con cota optimista 2 (mas costosa)
3 ~> Generar input
0 ~> Salir
1
- Resultado de RyP_in_1.txt:
  Prestigio total 3052, nodos explorados 186, tiempo total 25.8238 ms, tiempo medio 0.138838 ms
- Resultado de RyP_in_2.txt:
  Prestigio total 2518, nodos explorados 120, tiempo total 16.7612 ms, tiempo medio 0.139677 ms
- Resultado de RyP_in_3.txt:
  Prestigio total 2826, nodos explorados 108, tiempo total 17.0129 ms, tiempo medio 0.157527 ms
1 ~> Generar output con cota optimista 1 (menos costosa)
2 ~> Generar output con cota optimista 2 (mas costosa)
3 ~> Generar input
0 ~> Salir
2
- Resultado de RyP_in_1.txt:
  Prestigio total 3045, nodos explorados 258, tiempo total 44.4421 ms, tiempo medio 0.172256 ms
- Resultado de RyP_in_2.txt:
  Prestigio total 2518, nodos explorados 122, tiempo total 19.8207 ms, tiempo medio 0.162465 ms
- Resultado de RyP_in_3.txt:
  Prestigio total 2826, nodos explorados 100, tiempo total 15.7277 ms, tiempo medio 0.157277 ms
```