CodeWars 2007

Problem #7 -- RAID Sizer

7 Points

JAVA programmers: your program name must be: prob07.class
C programmers: your program name must be: prob07.exe

Task Description

Servers use a number of techniques to increase hard disk performance, while also adding duplicate disks (known as redundancy) to prevent a loss of data if one fails. These techniques are known as a *Redundant Array of Inexpensive Disks* or "RAID". There are three commonly used RAID levels:

- RAID 0: Two or more identical disks are made to appear as one large disk, with all data "striped" across the disks. This increases performance, but doesn't offer any protection in case one disk fails. Example: Three 250GB disks arranged in a RAID 0 array would appear to the user as a single, 750GB disk.
- RAID 1: For each disk, a duplicate disk is added to the system, with all data "mirrored" on both disks. While this does not increase capacity, it allows for one of the disks to fail without losing data. Mirrored disks are always added in identical pairs. Example: A 500GB disk is mirrored to another 500GB disk. The user sees one 500GB disk, but can sleep better knowing either disk can crash without her data being lost.
- RAID 5: This technique stripes data across 2 or more identical disks (up to 7), and produces "parity data" on an additional identical disk for redundancy. This is like using RAID 0 and adding a single mirror disk to handle the whole group. Example: Four 400GB disks are arranged in a RAID 5 array – three of them striped, giving a capacity of 1200GB. The fourth disk is used only for fault tolerance.

Write a program to determine the cheapest array configuration. Your array can use up to 8 disks of identical capacity. Your array can use the following capacities (prices): 250GB ($75), 400GB ($110), 500GB ($140), 750GB ($250).

Program Input

Your program must prompt for the array's capacity (not including capacity used for redundancy), and the desired RAID level: 0 (zero), 1 (one) or 5. Note that the array capacity may be larger than requested, based on the disk sizes.

Program Output

The program must output to the screen the least expensive combination of disks required to build the array, including the size and quantity of disks needed, and the total user capacity and cost of the array.

Sample Input/Output

```
Enter array capacity (in GB):   450
Enter RAID level (0, 1, 5):  1

Qty: 2  Disk: 500GB  Price:  $140
Total price of this 500GB array: $280

Enter array capacity (in GB):   1200
Enter RAID level (0, 1, 5):  0

Qty: 3  Disk: 400GB  Price:  $110
Total price of this 1200GB array: $330

Enter array capacity (in GB):   600
Enter RAID level (0, 1, 5):  5

Qty: 4  Disk: 250GB  Price:  $75
Total price of this 750GB array: $300
```

Sample Input/Output

```
Enter array capacity (in GB):   1300
Enter RAID level (0, 1, 5):  1

Qty: 6  Disk: 500GB  Price:  $140
Total price of this 1500GB array: $840

Enter array capacity (in GB):   3000
Enter RAID level (0, 1, 5):  5

Qty: 7  Disk: 500GB  Price:  $140
Total price of this 3000GB array: $980

Enter array capacity (in GB):   4500
Enter RAID level (0, 1, 5):  5

Qty: 7  Disk: 750GB  Price:  $250
Total price of this 4500GB array: $1750
```