Joule Thief
Programming Project 2

The story is not here.

Job Details. Your task is to implement a *dynamic programming algorithm* to solve the 0-1 Knapsack problem in `C++`. You may work in groups of two (2) or three (3) if you wish.

Phase 1 – Dynamic Program Write the basic dynamic program to solve the 0-1 Knapsack problem that returns the total value of the best solution. The algorithm to do the project is found on page 192 of the textbook (Section 4.5.3) and the terminology for the variables is found on page 190. The dynamic programming solution SHOULD find the optimal solution but it may take a long time.

Phase 2 – Report the Items Augment your solution to report which items were part of the final solution. This will need an auxiliary two-dimensional array similar to what we discussed in class about chain matrix multiplication.

Phase 3 – Refinement Perform the refinement that is described in Section 4.5.4 of the book. Your program should now, in addition to reporting the correct items and overall value, must also report how many entries of the matrix were actually calculated.

The input is a text file with the following form:

```
7 25
orange 50 5
banana 60 10
kitchensink 140 20
strawberry 100 14
tangerine 10 5
puppy 25 12
grape 30 7
```

The first line of input gives the *number of items* in the file, followed by the *knapsack capacity*. Each subsequent line lists a type of item (such as an orange), with the value per item (in pesos) and a weight per item (in pounds). The goal is to choose the items that maximizes the value stored in the knapsack. The optimal solution to the above example is to use the $(50, 5)$ and $(140, 20)$ items for a total weight of 25 pounds and a value of 190 pesos.

Your program should accept as input the name of a text file of the form above, and it should output a list of items to select, together with the total weight and value.

Program Output Required. Run your program that builds text files for n (the number of items) being 10, 20, 100, 1000, and 10000 that follow the format above. Make sure your program works for these cases. I will create my own input files to test on your code to make sure it works. This is the output format that I need for your programs (without the < and > symbols):

```
<number-of-items-in-solution>
<total-weight-of-items-in-solution>
<total-profit-of-items-in-solution>

Items in the Solution:
<item-1-name> <item-1-pesos> <item-1-weight>
<item-2-name> <item-2-pesos> <item-2-weight>
....
```

What To Turn In. Turn in all of the following items on Canvas.

- The code for the last phase of the project.
- Instructions for your project—format, how to run, etc.—if needed. (digital)
- Sample runs for 10 and 20.