Computational Problem Solving CSCI-603
You Are a Medieval Trader Lab 5

1 Problem

You are a medieval trader living between two towns. There is a market in each town, and since this is medieval times, there is not much communication between the towns. Therefore, prices of the same goods may be quite different in the two towns. You have realized that you may be able to make a good living buying goods in one town and taking them to the other to sell. You have sent out your two children to find out the prices of the items for sale in both towns, along with how much of each item is for sale, and they have now returned with that information.

You are now about to head out - you can take only one trip, and can carry only ten items (in these towns, all items are the same weight, be it a gold bar, a donkey, or a basket of vegetables). You can assume that as a successful trader, you have enough money to purchase any items you would like. At your destination city you will set up your own stall and sell those items for the prevailing price in that city. You would like to know which city to go to, which items to buy, and how much profit you will make when selling them in the other city.

Problem Solving

For the problem-solving session, you will work in groups of approximately four students, as directed by your instructor. Put all of your groups' answers on a single sheet, with all group members' names.

1. One particular day your children go out and find the current available goods and their prices in two towns, Hilltown and Valleydale:

Hilltown	Valleydale
4 Swords @ £9 each	10 Donkeys @ £5 each
8 Donkeys @ £7 each	2 Swords @ £6 each
6 Cookpots @ £2 each	12 Cookpots @ £3 each

   Recall that if you buy items at the given price in one town, you can sell them for the listed price in the other town.
   (a) If you can carry a maximum of 5 items, what is the most profit you can make by buying items in Hilltown and selling them in Valleydale?
   (b) If you can carry a maximum of 5 items, what is the most profit you can make by buying items in Valleydale and selling them in Hilltown?
   (c) If you can carry as many items as you want, which town should you buy in, and how much profit will you make?

2. Now consider how we might represent this problem with a computer.
   (a) How would you represent the above market information in two files, one per town? Describe the representation in one or two sentences.
   (b) What would the file look like for the town of Hilltown? Carefully write the file out by hand.
   (c) What data structure could you use to represent the information in the files? Remember that we want to know which goods to buy from a particular town where we would make the maximum profit, so you will eventually need to compare the costs of the same item in the two towns. Your description should include include related terms and syntax from Python.

3. Give pseudocode for reading in two data files and building the data structure needed to determine the maximum possible profit. Do not give the pseudocode or code for actually computing the total profit — we are simply concerned with reading in the files and populating your data structures.

Implementation

For this assignment, you will write a piece of code that gets data from two towns and determines how to make the maximum profit.

Your code should prompt for a maximum number of items to be carried and two file names. The first line of each file will be the name of the city, and each following line will contain the name of an item, the number of those items for sale, and the unit price. For example:

```
Valleydale
Donkeys 10 5
Swords 2 6
Cookpots 12 3
```

It should then compute which town to purchase items in, and which items to purchase, in order to maximize your profit. The output of your code should look something like:

```
Go to Valleydale and buy:
2 Swords for a profit of 6.0
3 Donkeys for a profit of 6.0

income is 12.0
```

You may find it useful to sort some items in your solution. To do this, for this week, we would like you **not** to use Python's `sorted` function. Instead, please use and modify one of the sorts presented in lecture this week. Leave this in a separate file, making sure to include the proper authorship tags (but if you modify it, you should add yourselves as authors). You can then import this into the file with your main code.

Another Python feature that you may find used is the *namedtuple*. A named tuple is a simple type of class that Python provides for storing small data objects with a few components. A named tuple is in fact just a tuple, but in addition to accessing the items within the tuple by index, you can index them by name. For example (from the Python documentation):

```
Point = namedtuple('Point', ['x', 'y'])   # define the data type (class)
>>> p = Point(11, 22)      # instantiate with positional arguments
>>> p = Point(x=11, y=22)  # instantiate with keyword arguments
>>> p[0] + p[1]            # indexable like the plain tuple (11, 22)
33
>>> p.x + p.y              # fields also accessible by name
33
>>> x, y = p               # unpack like a regular tuple
>>> x, y
(11, 22)
```

To used `namedtuple` you will need to import it: `from collections import namedtuple`.

Submission

Submit your code via the `try` system — `try` will expect one file named `trader.py` but will also accept any other `.py` files that your solution uses:

```
try grd-603 lab5-1 trader.py
```

or

```
try grd-603 lab5-1 trader.py your-sorting-module.py
```

Grading

Your grade will be determined as follows:

- 20% : problem solving participation and solutions
- 10% : proper file handling
- 40% : computation of maximum possible profit
- 20% : output includes complete shopping lists
- 10% : proper documentation and coding style