Nesting Boxes (Due 16 Oct 2020)

Imagine a room full of boxes. Each box has a length, width, and height. Since the boxes can be rotated those terms are inter- changeable. The dimensions are integral values in a consistent system of units. The boxes have rectangular surfaces and can be nested inside each other. A box can nest inside another box if all its dimensions are strictly less than the corresponding dimensions of the other. You may only nest a box such that the corresponding surfaces are parallel to each other. A box may not be nested along the diagonal. You cannot also put two or more boxes side by side inside another box.

You will read your input from standard input. The list of boxes will be in a file called boxes.in. The first line gives the number of boxes *n*. The next *n* lines gives a set of three integers separated by one or more spaces. These integers represent the 3 dimensions of a box. Since you can rotate the boxes, the order of the dimensions does not matter. It may be to your advantage to sort the dimensions in ascending order.

There will be just two lines in your output. The first line will be an integer giving the largest number of boxes that can fit inside each other. The second line will give the number of such sets of boxes that do fit. For the input values given above here is the solution set. For this solution set your output will be

```
4
6
```

Where *4* represents the largest number of boxes that fit inside each other and *6* represents the number of sets of such nesting boxes.

What if there are no nesting boxes? That is, you cannot find a pair of boxes that fit. Then for the above input file (if this case was true) your output will be

```
1
20
```

The *1* represents that there are no boxes that fit and the *20* is the total number of boxes.

For this assignment you may work with a partner. Both of you must read the paper on Pair Programming and abide by the ground rules as stated in that paper.

Here is the template of the file called Boxes.py that you will be turning in. We are looking for clean and structured design using the standard coding conventions in Python. You may not change the function signatures but you may add as many helper functions as needed. The file will have a header of the following form:

```
#  File: Boxes.py

#  Description:

#  Student Name:

#  Student UT EID:

#  Partner Name:

#  Partner UT EID:

#  Course Name: CS 313E

#  Unique Number:

#  Date Created:

#  Date Last Modified:
```

To run this code on the command line on the Mac you will do

```
python3 Boxes.py
```

To run the code on the command line on a Windows machine you will do

```
python Boxes.py
```

If you are working with a partner you will be submitting only one copy of your program but make sure that you have your partner's name and eid in your program. If you are working alone, then remove the two lines that has the partner's name and eid in the header.

Use the Canvas system to submit your **Boxes.py** file. We should receive your work by 11 PM on Friday, 16 Oct 2020. There will be substantial penalties if you do not adhere to the guidelines. Remember Python is case sensitive. The name of your file must match exactly what we have specified.

- Your Python program should have the proper header.
- Your code must run before submission.
- You should be submitting your file through the web based *Canvas* program. We will not accept files e-mailed to us.