[redacted]

CMPT 120: Final

Coding Test
April 15, 2018

Instructions
You are not permitted any reference sheet for this test.

Log into http://coursys.sfu.ca and complete the following questions for **Final - Coding Q1**, **Final - Coding Q2** and **Final - Coding Q3**.

It must be submitted to **Coursys** before 4:45pm to be considered for marks.
You may not leave the exam room before 4:45pm.

REMEMBER to use **2 spaces instead of tab** for indentation in your code submission.

You MAY NOT have any other browser tabs or applications open other than Coursys.

Final - Coding Q1

Unique Baby Animal Names - 10 points

You are considering how to name your new pet turtle, and are tired of your usual "Annie", "Buster", "Clifford", etc. You decide you need to start looking for names from the end of alphabet, like "Xena".

You have a pet name dataset with 12,000 names, so you start by testing with a smaller list.

Consider the following test list, representing a partial list of names and their popularity scores.

```
names = [ ("Yoda", 0.455), ("Abbie", 0.66), ("Gigi", 0.5), ("Xena", 0.12), ("Champ", 0.3) ]
```

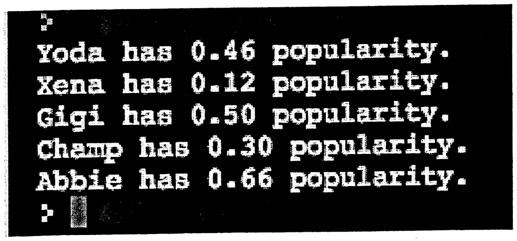
Next, consider that in Python, one can compare strings in the same way that numbers are compared. For instance, print("apple" < "banana") outputs **True**, and print("apple" > "banana") outputs **False**. In other words, alphabetical order is respected for comparison operators.

Question 1a

Define a function **sort_backwards(names)** that can sort the list above, based on the **string** in the tuple at **index 0**. It should return the sorted list of tuples in reverse-alphabetical order, i.e. from Z to A. You may not use any built-in Python sorting methods to accomplish this task.

Question 1b

Write a short program that uses the **sort_backwards(names)** function from 1a. It should use the test list **names** above as input, and print the first 5 elements in the list, formatted as below. Note that the number of decimal places in the formatting must be respected, and that the code should also work for when **names** is set to its full length of 12,000 names.



```
>
Yoda has 0.46 popularity.
Xena has 0.12 popularity.
Gigi has 0.50 popularity.
Champ has 0.30 popularity.
Abbie has 0.66 popularity.
>
```

Final - Coding Q2

Old MacDonald Had a Vowel - 10 points

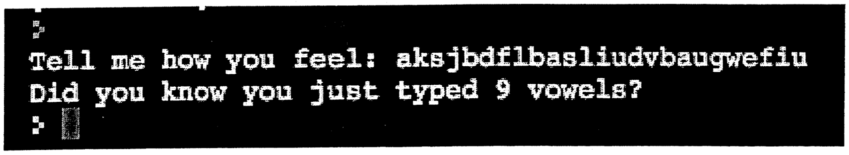
It's the end of the semester, and you have taken to banging your head on the keyboard.

Question 2a

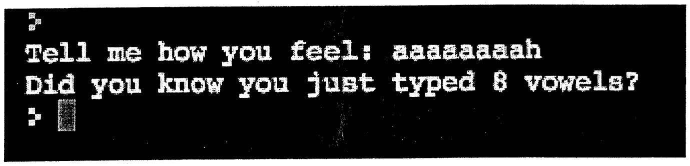
Write a program that counts how many vowels (a, e, i, o, u) you inadvertently type. See the sample runs below.

Your program must include the definition and use of a function **vowel_count(letters)** which returns the number of vowels in the string **letters**.

Two sample runs:



```
>
Tell me how you feel: aksjbdflbasliudvbaugwefiu
Did you know you just typed 9 vowels?
> 
```



```
>
Tell me how you feel: aaaaaaaah
Did you know you just typed 8 vowels?
> 
```

Question 2b

Now, write a **recursive** function **vowel_count_rec(letters)** that performs the same task as in Question 2a, but using recursion. Bonus point: Can you can make the function work perfectly in 5 lines or less?

Final - Coding Q3

Wine Snob - 10 points

You have been given a **wines.csv** file containing a list of 130,000 wines in the world today. The first 6 lines of the file are shown here:

wines.csv

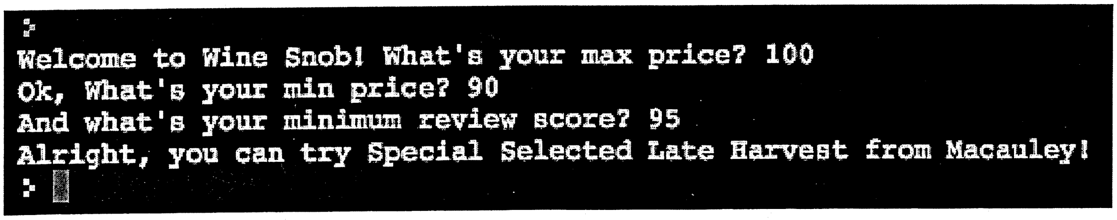
```
name,vineyard,type,price,reviewscore
Martha's Vineyard,Heitz,Cabernet Sauvignon,235,96
Carodorum Reserva,Bodega Carmen Rodriguez,Tinta de Toro,110,96
Special Selected Late Harvest,Macauley,Sauvignon Blanc,90,96
Reserve,Ponzi,Pinot Noir,65,96
La Brulade,Domaine de la Begude,Provence Red Blend,66,95
```

Question 3a

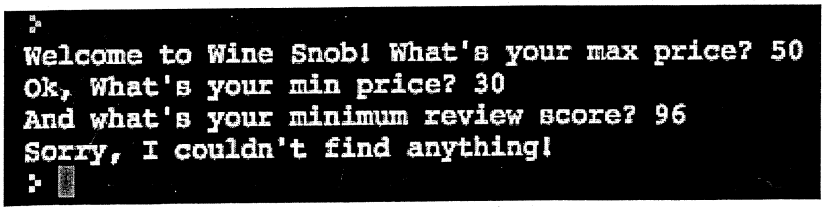
Write a **Wine Snob** program that reads the wines.csv file and searches for a wine that suits the user's preferences. Your program should ask the user a minimum price, a maximum price and a minimum review score. It should output the name of the wine and the vineyard, similar to the sample runs below.

The overall algorithm you choose must be the most efficient for this search task involving unsorted data. Assume you only need to find one wine that fits the criteria, and only need to perform this search once before terminating. You may not use any built-in Python search or sorting methods to accomplish this task.

Here are two example runs, using a test file containing the first 6 lines of wines.csv.



```
>
Welcome to Wine Snob! What's your max price? 100
Ok, What's your min price? 90
And what's your minimum review score? 95
Alright, you can try Special Selected Late Harvest from Macauley!
>
```



```
>
Welcome to Wine Snob! What's your max price? 50
Ok, What's your min price? 30
And what's your minimum review score? 96
Sorry, I couldn't find anything!
>
```

Question 3b

What is the worst-case complexity of your program in big O notation?