Assignment 4

CPSC 217 – L02

Purpose

You will be writing a Python program to read data from a file and visualize this data using an external drawing tool. You will structure your program using modules and functions to demonstrate good design. Exceptions will be used where appropriate in order to handle errors in a user-friendly manner.

Important Note

- This is an individual assignment. What you submit must be your own work, although you may discuss the problem in general terms with other people. You should definitely not be showing other people your code, and generally speaking, it is not a good idea to talk about the assignment when you're sitting in front of the computer.
- Sources of algorithms and code, if any, must be properly cited. Remember that plagiarism regulations apply to code too. You can put citations into comments in your Python code.
- If you have any questions about what you can and can't safely do, feel free to talk to me.

Data visualization

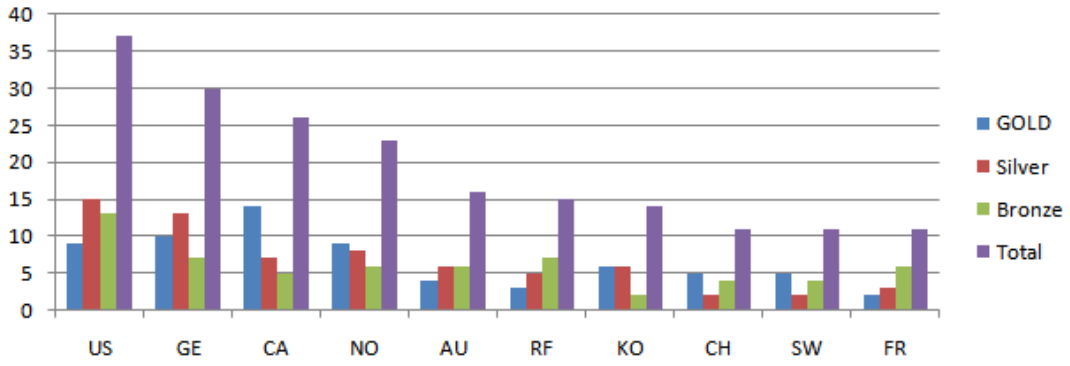
Data visualization is a graphical means to visualize sets of data so that the data becomes easier to read and analyze. A common form of data visualization is called bar charts.

For example, the file medal_count.txt contains the following data:

Country	Gold	Silver	Bronze	Total
US	9	15	13	37
GE	10	13	7	30
CA	14	7	5	26
NO	9	8	6	23
AU	4	6	6	16

RF	3	5	7	15
KO	6	6	2	14
CH	5	2	4	11
SW	5	2	4	11
FR	2	3	6	11

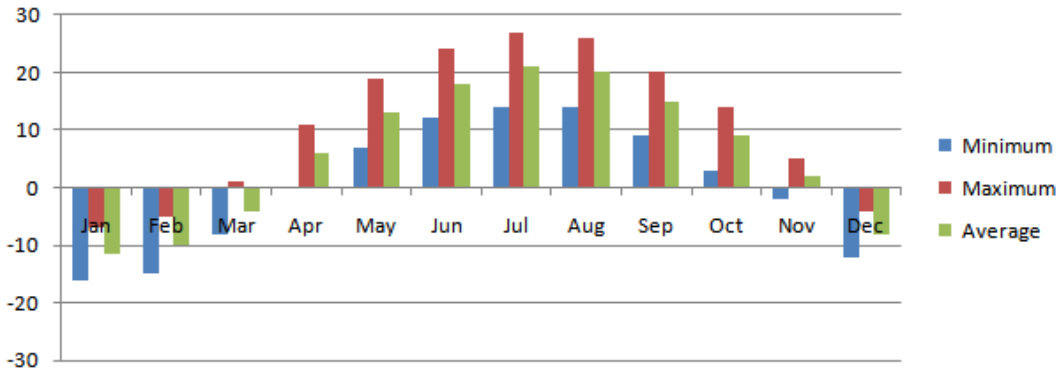
This data can be visualized using a bar chart as follows:



Another example is the file temperatures.txt with the following data:

Month	**Minimum**	**Maximum**	**Average**
Jan	-16	-7	-11.5
Feb	-15	-5	-10
Mar	-8	1	-4
Apr	0	11	6
May	7	19	13
Jun	12	24	18
Jul	14	27	21
Aug	14	26	20
Sep	9	20	15
Oct	3	14	9
Nov	-2	5	2
Dec	-12	-4	-8

This data can be visualized using a bar chart as follows:

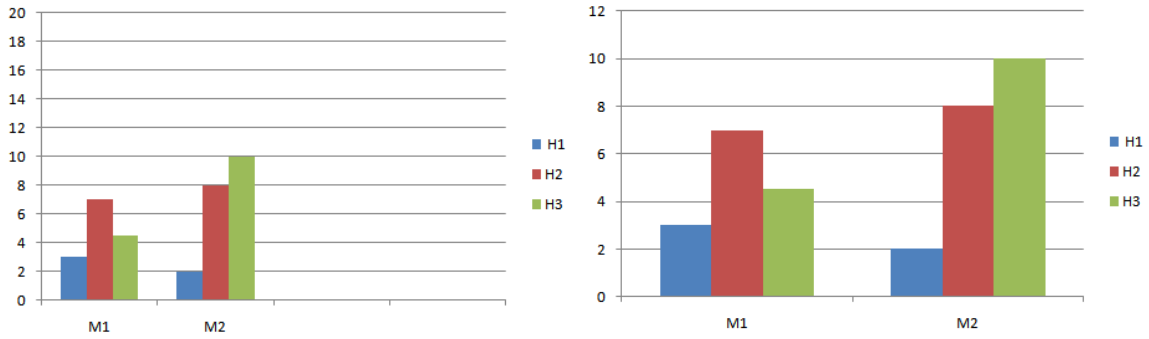


Your Task

Your task is to read some data from a text file, and plot this data using a bar chart. Keep in mind the following:

- Provide a legend to enable the user to map between colors and data. Make sure the colors you pick have enough contrast with the background and with each other.

- Your bar chart should scale properly (vertically and horizontally) to make use of the space on the screen. For example, if all the values are positive, the y-axis should not extend to the negative side. Also, your bar chart should not extend a lot beyond the maximum absolute value in the data file. If you only have a few rows of data, you should scale up the width of the bars. Look at the examples below. The visualization to the left is not ideal because there is a lot of unused white space in the graph. The visualization to the right is better because it scales properly.



- Fields in the data file are tab-separated.

- All the fields except the header row and the first column should contain numerical data. If this is not the case, you should ignore any row that contains invalid data, and report a user-friendly error message to the user.

- Your python program should take the data file name as a command line argument. For example, to draw the bar chart above, we should be able to type:

  ```
  python bar_chart.py temperatures.txt | java –jar quickdraw.jar
  ```

  To get the file name in your python program, you can say:

  ```
  import sys
  filename = sys.argv[1]
  ```

  If the user did not pass any argument, or the file does not exist, you should report a user-friendly error message to the user.

- When you write your python program, make sure you use modules and functions to produce well-structured code. Testing for error conditions (including invalid input) should be handled by exceptions where appropriate.

- Use the provided data files for testing purposes to make sure your solution meets the specifications. It is also advisable that you create your own data files to test further.

Bonus

For a bonus mark, provide a doctest for at least three of your functions.

General Hints

- There are plenty of resources online on how to draw a good bar chart using simple drawing tools. You can start there, but make sure you reference any resources you use.
- Make a directory to keep your files for this assignment in.
- Take notes as you go along, so you can remember what you've tried already and what did and didn't work.
- Break the task down into small pieces.
- Use an incremental approach, and try out things on the Python command line as you go, before putting them in your program.

- Work through everything one last time prior to your demo to help catch any last-minute problems.
- Make absolutely sure that your solution will run on the Computer Science machines!

Evaluation

You must do two things:

1. Hand in a printout of your solution to the assignment boxes by 4pm Friday, 9 April 2010. Your printout should include all the .py files corresponding to the classes and modules you defined. Make sure it has your name and/or student ID number, and double-check to ensure you're using the correct assignment box! The assignment boxes are on the second floor of the Math Science building. Note that you may turn your printout into the box early. If your TA is accepting solutions in any other form (e.g., email) then they will let you know - however, you must still observe the deadline.
2. Demonstrate your solution to your TA in tutorials during the week of April 12. To keep things fair, the solution you demonstrate **must** be the same as what you handed in on the printout, or you will not be given credit for the demo. Your TA may ask you questions about what the different parts of your solution do.

If you do not hand in a printout or if you do not demonstrate your assignment during tutorial time, or both, you cannot be given a grade above a zero on this assignment.

Your printout **must** show your Python program and all the .py files corresponding to the modules you defined. **Note that you must have a program in a .py file - you cannot turn in a solution that only uses the Python command line.**

Your solution must be demonstrated using your account on the CPSC machines.

Tutorials during the week of April 12 are allocated for demos. Your TA is not obliged to see demos outside this time; they have their own schoolwork to do!

The TA has the right to assign a mark of zero for the entire assignment if you fail the demo.

- 1/2 of the marks assigned are for the design and style of your code. This includes the effective use of classes, modules, functions, and the appropriate

data structures to solve the problem. It also includes code readability, appropriate variable names and comments.

- 1/2 of the marks assigned are for the correctness of your output. This includes the conformance of your output to the criteria listed in the specifications above, robust handling of data and user input, and appropriate error checking and reporting.