Chp.6 – Olympics

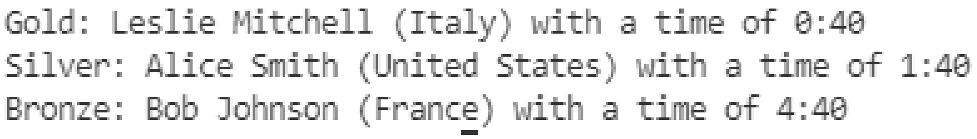
You are an official working at the 2024 Summer Olympics in Paris. As an official, your job is to take in the times for the track events and determine who won gold, silver, and bronze medals. The timing system at the Olympics outputs a comma-separated values (CSV) file. CSV files are a common way to store data. An example CSV file the timing system outputs is as follows:

Alice Smith,United States,1:40

Bob Johnson,France,4:40

Leslie Mitchell,Italy,0:40

Each line has the results of a different runner. The start of the line is the runner's name, followed by a comma, then the country, followed by another comma, and then the time of the runner. The time is always the minute, followed by a colon, and then the seconds. In this case, there will only be minutes and seconds. Do not worry about hours. In CSV files, each property is separated by a comma. In our case, there are commas after the runner's name and after the country to separate these properties. Your task is read in the CSV file and determine who wins the gold, silver, and bronze medals. An example output for the CSV file above is below:



```
Gold: Leslie Mitchell (Italy) with a time of 0:40
Silver: Alice Smith (United States) with a time of 1:40
Bronze: Bob Johnson (France) with a time of 4:40
```

Unparseable Lines

The timing system used at the Olympics was written over a decade ago and has many errors. Unfortunately, the person who developed the timing system has since retired and no one can understand its code (a problem in CS that is more common than one would think). Thus, your program must account for possible errors in the CSV file. Here are the errors your program should handle:

- One or more of the properties on a line is blank. Examples:

  ,United States,1:40

  ,,1:40

  Alice Smith,United States,

- The line has more than three properties. Example:

  Alice Smith,United States,1:40,Women's 400

- The minute or second is not an integer. Examples:

  Alice Smith,United States,MIN:40

  Alice Smith,United States,1:QWERTY

In the case that your program fails to parse a line due to one of the errors above, ignore the line. Once the program has determined who won gold, silver, and bronze, print out the number of lines that contained errors. For example take the following CSV file:

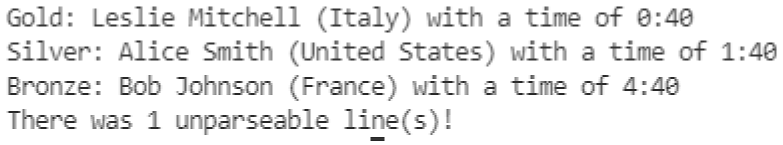
Alice Smith,United States,1:40

Bob Johnson,France,4:40

Leslie Mitchell,Italy,0:40

Bob Ross,,0:01

Since Bob Ross is missing his country, the output would be:



```
Gold: Leslie Mitchell (Italy) with a time of 0:40
Silver: Alice Smith (United States) with a time of 1:40
Bronze: Bob Johnson (France) with a time of 4:40
There was 1 unparseable line(s)!
```

Other Considerations:

- You ***MUST name your CSV file "results.txt" and place it in the same directory as your Python program***. ***Do not use absolute paths (ie C:/Users/Marquette/results.txt or /Users/Marquette/Documents/results.txt) when trying to open the file***. As long as you place the file and the Python program in the same directory, you can open it by using "results.txt" as the name. Using absolute paths or a file name other than "results.txt" will cause TA-Bot to fail all test cases. No one wants that.
- Assume there are no ties. No runners will have the same times.
- Your program should handle an unspecified number of runners.
- Some events will have less than three runners (due to invalid lines or less than 3 runners in the CSV file). In this case, do not print out the medals for which no runner won. For example, take a CSV file that contains these entries:

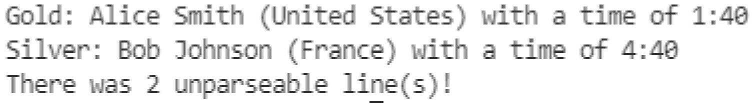
  Alice Smith,United States,1:40

  Bob Johnson,France,4:40

  ,,0:40

  Bob Ross,,0:01

  In this case, the last two lines are invalid and thus discarded. The output will be:

  ```
  Gold: Alice Smith (United States) with a time of 1:40
  Silver: Bob Johnson (France) with a time of 4:40
  There was 2 unparseable line(s)!
  ```

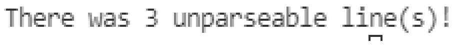
  This means there may be cases where no medals are awarded due to all lines being invalid. For example if the CSV file contains:

  ,,,

  ,,0:40

  Bob Ross,,0:01

  Then the output will be:



```
There was 3 unparseable line(s)!
```

No medals were awarded because all lines were invalid.

- Python does have native libraries to parse CSV files, however these will be of no use. TA-Bot does not install any libraries before it runs student code. Instead, use string manipulation to divide the parts of the line.