CS 326 – Project #7

Purpose: Become familiar with basic Ruby syntax, data types, and control structures.
Points: 125

Assignment:
Write a simple Ruby program to create an HTML scores file for posting class scores.

The program should read a comma separated text file (exported form a spreadsheet) containing class scores and create an HTML file for displaying the scores. The program should read command line parameters to:

- determine if just the ID's or the Names and ID's should be displayed
  - command line qualifier: ***-ids*** or ***-names***
- input file name

The program should have a main that calls various procedures. You may use global variables. Additionally, you can use I/O redirection to capture the HTML output.

An example execution might be:

```
% ruby proj6.pl -ids scores1.csv > scores.html
% ruby proj6.pl -names scores1.csv > names.html
```

The source file will always be the same format. This includes:

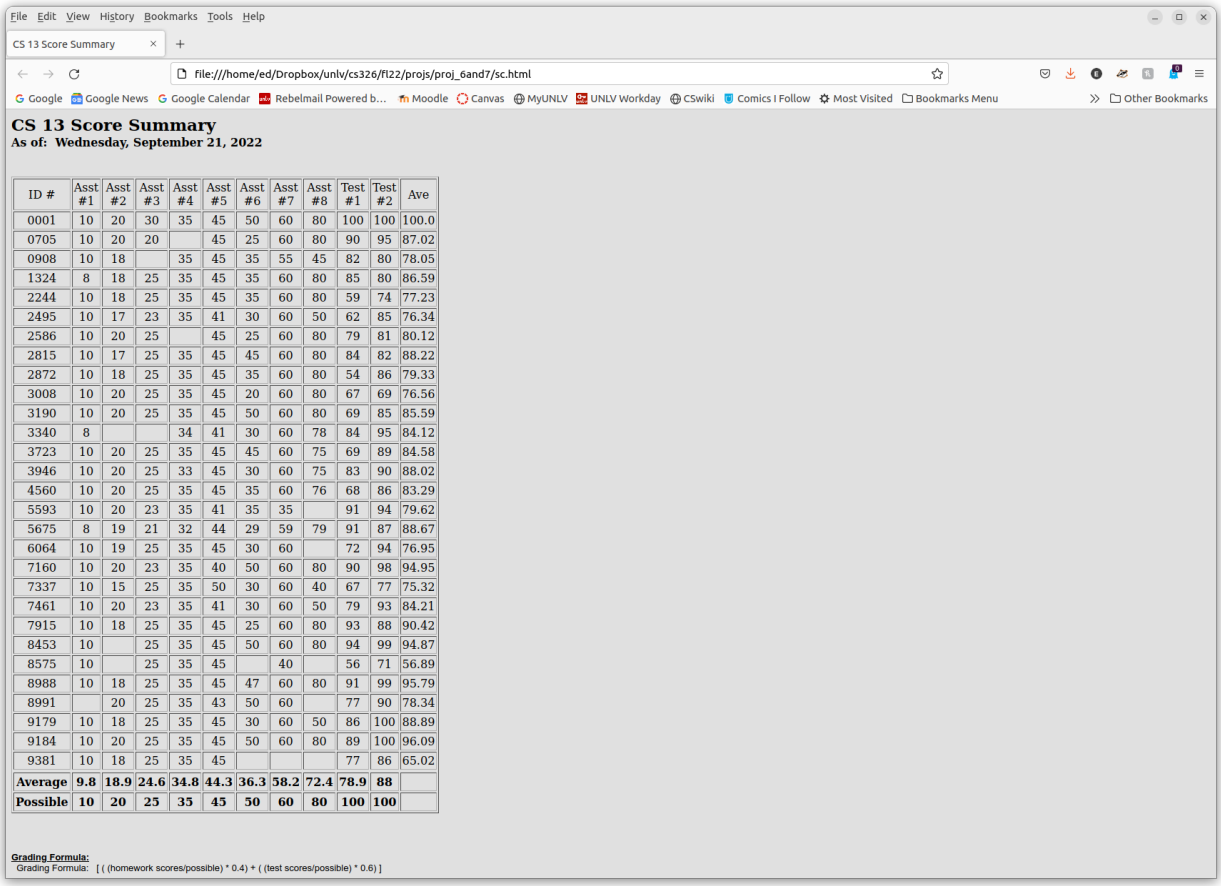
```
,<class title>
,Assignments, <assignment count>
,Tests,<tests count>
,Assignments weight, <assignments weight>
,Tests Weight, <tests weight>
<titles>
,Possible,<... possible scores ...>
<id>, <name>, <... scores ...>
<id>, <name>, <... scores ...>
<id>, <name>, <... scores ...>
...
...
```

An example input file is provided on the class web page.

Submission:
1) Submit a copy of the Ruby source file.
2) Submit a copy of the output for the provided input file.

Example Execution:

A sample output of your program might produce a web page that looks something like this (with ID's):



File Edit View History Bookmarks Tools Help

CS 13 Score Summary

file:///home/ed/Dropbox/unlv/cs326/fl22/projs/proj_6and7/sc.html

Google Google News Google Calendar Rebelmail Powered b... Moodle Canvas MyUNLV UNLV Workday CSwiki Comics I Follow Most Visited Bookmarks Menu Other Bookmarks

CS 13 Score Summary

As of: Wednesday, September 21, 2022

ID #	Asst #1	Asst #2	Asst #3	Asst #4	Asst #5	Asst #6	Asst #7	Asst #8	Test #1	Test #2	Ave
0001	10	20	30	35	45	50	60	80	100	100	100.0
0705	10	20	20		45	25	60	80	90	95	87.02
0908	10	18		35	45	35	55	45	82	80	78.05
1324	8	18	25	35	45	35	60	80	85	80	86.59
2244	10	18	25	35	45	35	60	80	59	74	77.23
2495	10	17	23	35	41	30	60	50	62	85	76.34
2586	10	20	25		45	25	60	80	79	81	80.12
2815	10	17	25	35	45	45	60	80	84	82	88.22
2872	10	18	25	35	45	35	60	80	54	86	79.33
3008	10	20	25	35	45	20	60	80	67	69	76.56
3190	10	20	25	35	45	50	60	80	69	85	85.59
3340	8			34	41	30	60	78	84	95	84.12
3723	10	20	25	35	45	45	60	75	69	89	84.58
3946	10	20	25	33	45	30	60	75	83	90	88.02
4560	10	20	25	35	45	35	60	76	68	86	83.29
5593	10	20	23	35	41	35	35		91	94	79.62
5675	8	19	21	32	44	29	59	79	91	87	88.67
6064	10	19	25	35	45	30	60		72	94	76.95
7160	10	20	23	35	40	50	60	80	90	98	94.95
7337	10	15	25	35	50	30	60	40	67	77	75.32
7461	10	20	23	35	41	30	60	50	79	93	84.21
7915	10	18	25	35	45	25	60	80	93	88	90.42
8453	10		25	35	45	50	60	80	94	99	94.87
8575	10		25	35	45		40		56	71	56.89
8988	10	18	25	35	45	47	60	80	91	99	95.79
8991		20	25	35	43	50	60		77	90	78.34
9179	10	18	25	35	45	30	60	50	86	100	88.89
9184	10	20	25	35	45	50	60	80	89	100	96.09
9381	10	18	25	35	45				77	86	65.02
Average	**9.8**	**18.9**	**24.6**	**34.8**	**44.3**	**36.3**	**58.2**	**72.4**	**78.9**	**88**	
Possible	**10**	**20**	**25**	**35**	**45**	**50**	**60**	**80**	**100**	**100**	

Grading Formula:

Grading Formula: [((homework scores/possible) * 0.4) + ((test scores/possible) * 0.6)]