

605.201 Module 10 Java I/O Assignment

Introduction:

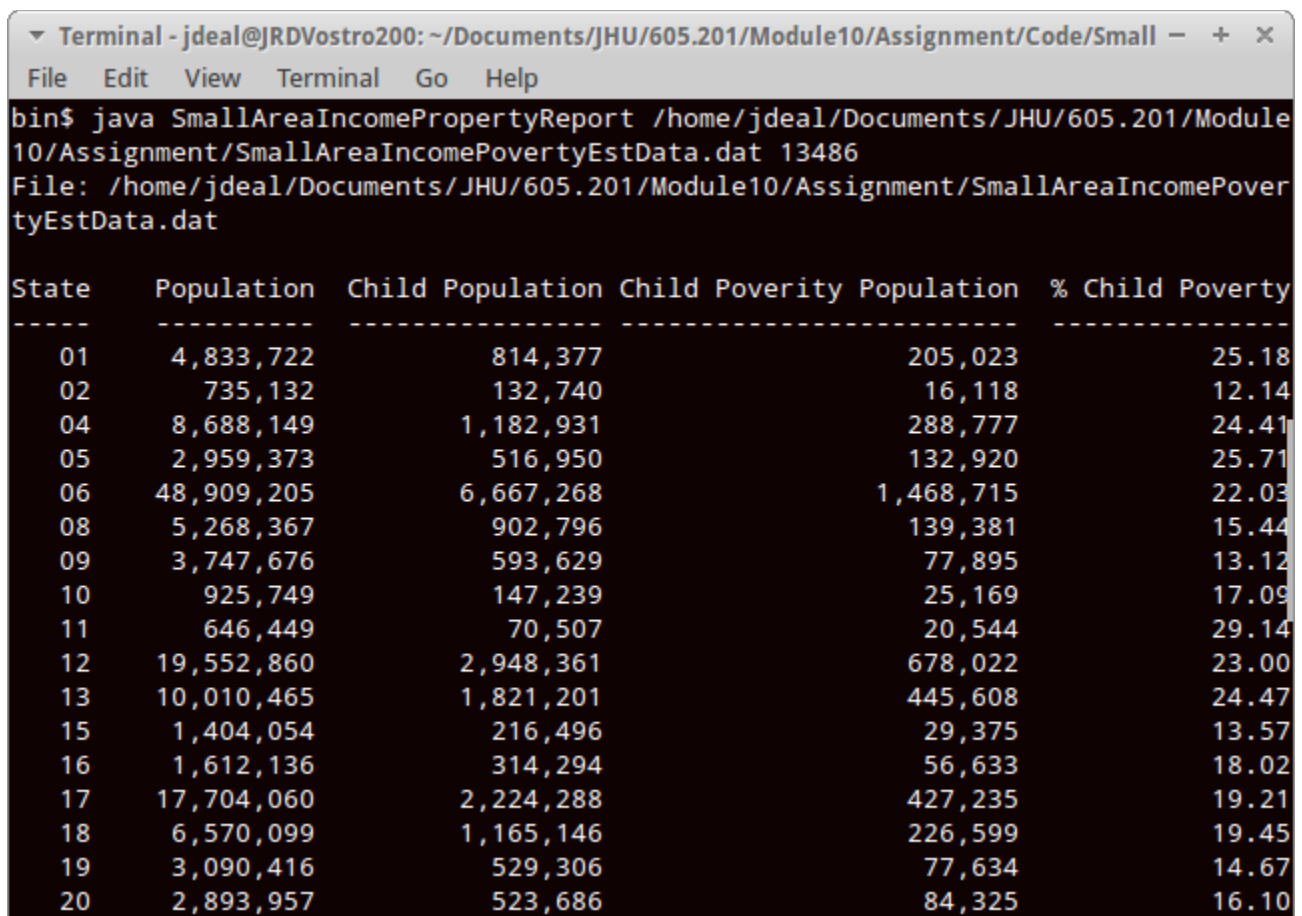
This assignment is to provide practice in using the Java I/O techniques discussed in the Module 10 video lectures and readings. Although the main focus of this assignment is Java I/O techniques, Java design and implementation techniques discussed in earlier modules should be incorporated in to this assignment.

Problem:

Supplied is a data file from the US Census which contains data from US school districts and reports statistics related to child poverty (<http://www.census.gov/did/www/saipe/data/highlights/2013.html>). It is desired to have a summary report which calculate basic statistics at the state level.

Desired Implementation:

Java 8 implementation to read the supplied text data and produce a report similar to the below:



The screenshot shows a terminal window titled "Terminal - jdeal@JRDVostro200: ~/Documents/JHU/605.201/Module10/Assignment/Code/Small". The command executed is `bin$ java SmallAreaIncomePropertyReport /home/jdeal/Documents/JHU/605.201/Module10/Assignment/SmallAreaIncomePovertyEstData.dat 13486`. The output is a table with 5 columns: State, Population, Child Population, Child Poverty Population, and % Child Poverty. The data is sorted by state number (01 to 20).

State	Population	Child Population	Child Poverty Population	% Child Poverty
01	4,833,722	814,377	205,023	25.18
02	735,132	132,740	16,118	12.14
04	8,688,149	1,182,931	288,777	24.41
05	2,959,373	516,950	132,920	25.71
06	48,909,205	6,667,268	1,468,715	22.03
08	5,268,367	902,796	139,381	15.44
09	3,747,676	593,629	77,895	13.12
10	925,749	147,239	25,169	17.09
11	646,449	70,507	20,544	29.14
12	19,552,860	2,948,361	678,022	23.00
13	10,010,465	1,821,201	445,608	24.47
15	1,404,054	216,496	29,375	13.57
16	1,612,136	314,294	56,633	18.02
17	17,704,060	2,224,288	427,235	19.21
18	6,570,099	1,165,146	226,599	19.45
19	3,090,416	529,306	77,634	14.67
20	2,893,957	523,686	84,325	16.10

Terminal - jdeal@JRDVostro200: ~/Documents/JHU/605.201/Module10/Assignment/Code/Small - + x					
File	Edit	View	Terminal	Go	Help
32	2,790,136		483,411	99,599	20.60
33	1,472,055		205,461	19,714	9.60
34	10,552,547		1,488,882	222,992	14.98
35	2,085,287		368,816	103,790	28.14
36	19,901,043		3,066,336	666,553	21.74
37	9,848,060		1,673,310	386,419	23.09
38	723,393		113,921	12,685	11.13
39	11,570,743		1,958,998	398,688	20.35
40	3,851,487		682,548	144,867	21.22
41	3,931,430		627,584	118,023	18.81
42	12,773,801		1,999,741	342,181	17.11
44	1,065,907		159,355	31,368	19.68
45	4,790,785		787,482	194,639	24.72
46	844,877		148,002	24,675	16.67
47	6,778,703		1,091,900	260,103	23.82
48	26,452,422		5,101,161	1,198,322	23.49
49	2,900,872		642,722	85,745	13.34
50	940,840		92,223	11,990	13.00
51	8,260,405		1,352,420	190,734	14.10
53	6,971,406		1,151,175	197,126	17.12
54	1,854,304		279,484	64,539	23.09
55	5,956,920		963,445	157,356	16.33
56	582,360		99,290	11,701	11.78
bin\$					

There should be two separate “programs” (main()), one to read the text data file and write a reformatted file to be read by the second program which will create the report to standard out. Note before the report is displayed, a single line with “File: “ then the path of the input file for the report is displayed.

The first program is to create a data file (not the report) which provides a pre-processed view of the data supplied to it either by stripping off the unneeded fields or by stripping off the unneeded fields and summing the data by state code. The numbers should not be formatted and no additional records should be produced.

The second program should read the file produced by the first program and produce the report in the format shown in the above image. This program should format the numbers and produce the file path information and column headings.

The first program will have 3 run-time parameters, the data source file path, the destination file path, and the number of records in the data file (13486) .

The second program will have 2 run-time parameters, the input file path and the number of records.

Features and Restrictions:

This assignment is an individual effort. Collaboration with other students on design approaches, implementation techniques, etc. as well as using the course's Discussion Board and other course

resources are encouraged but the design, implementation, and submitted files *must* be your own creation.

A good reference for the Java 8 API is at: <http://docs.oracle.com/javase/8/docs/api/>

The programs should use standard (SE) Java 8 code and compile without errors or warnings. It should also run without errors or warnings when given valid input.

The programs should provide reasonable parameter validation (correct number of parameters, reasonable values, etc.).

The programs should not use any Java collections (ArrayList, Map, Vector, etc.) except standard Java arrays. Collections are introduced in a later module.

The file produced by the first program should not be deleted after running the report program.

The program's code should be reasonable formatted and commented as demonstrated so far in the course.

Resources:

File: SmallAreaIncomePovertyEstData.txt – contains the small area poverty data. It is a standard 8-bit readable text file.

File: SmallAreaIncomePovertyEstLayout.txt – contains information about the field layout of the SmallAreaIncomePovertyEstData.txt file.

Submitted Files:

Please submit the following files in a single compressed zip file (.zip) using the following naming convention: Assignment10_<JHUID><section number>. An example is: Assignment10_jdealjr182

- Source files needed to recreate both of your programs
- The compiled .class files for your programs and any needed custom classes
- The supplied text files and any other additional files needed to run your programs
- Short word processed file (.odt, .doc, .docx) explaining how to run your program.