

Data Proc 2024: Assignment 04 (Part 2)

- Due: 6 Dec 2024 (before class: 1315 PM)
- Complete the assignment on the server using your user account.
 - You must create the scripts in the correct location in your home directory.

Problems past problem4.py are extra credit

Assignment 04 Part 2: Goals

- 1) Simple python processing
 - read command-line arguments via `sys.argv`
 - open and read a short file
 - write a loop to process the file contents
 - process some file contents (parse to numbers and add)
 - write results to a different file

Example scripts in \$DP24

- See the examples in for python examples:
 - \$DP24/lectures/04/examples

```
riveale@dataproc2023vm: /usr/share/dataproc/2024/lectures/04/examples$ ls
00_sysargv.py          09_process_strings.py
01_python_formatting.py  exfile1.txt
02_python_opentextfile.py exfile2.txt
03_python_textbylines.py exfile2_wspace.txt
04_python_withas_open.py run_examples.sh
05_python_otherreadlines.py testout.bin
06_python_binaryread.py  testout.scratch
07_python_writingfile.py testout.scratch.bin
08_python_writebinfile.py
```

I strongly recommend you look at these and make sure you understand them.

Assignment 04: Part 2

- Create a directory in your home:
 - `$HOME/dataproc2024/assignments/04_2/`
- You will create/complete the following scripts:
 - `$HOME/dataproc2024/assignments/04/problem0.py`
 - `$HOME/dataproc2024/assignments/04/problem1.py`
 - `$HOME/dataproc2024/assignments/04/problem2.py`
 - `$HOME/dataproc2024/assignments/04_2/problem3.py`
 - `$HOME/dataproc2024/assignments/04_2/problem4.py`
- **Extra credit:**
 - `$HOME/dataproc2024/assignments/04_2/problem5.py`
 - `$HOME/dataproc2024/assignments/04_2/problem6.py`
 - `$HOME/dataproc2024/assignments/04_2/problem7.py`

Assignment 04: Part 2

- Create a directory in your home:
 - `$HOME/dataproc2024/assignments/04_2/`
- You do not need to do "Extra Credit"
(it is for people who want to test their skills doing more complex problems,
e.g. binary)

 - `$HOME/dataproc2024/assignments/04_2/problem3.py`
 - `$HOME/dataproc2024/assignments/04_2/problem4.py`
- **Extra credit:**
 - `$HOME/dataproc2024/assignments/04_2/problem5.py`
 - `$HOME/dataproc2024/assignments/04_2/problem6.py`
 - `$HOME/dataproc2024/assignments/04_2/problem7.py`

Assignment 04: Part 2

- Create a directory in your home:

Copy the boilerplate (skeletons) I created in:
\$DP24/assignments/04_2

```
riveale@dataproc2023vm: /usr/share/dataproc/2024/assignments/04_2$ ls  
exfile2.txt  problem4.py  problem6.py  
problem3.py  problem5.py  problem7.py  
riveale@dataproc2023vm: /usr/share/dataproc/2024/assignments/04_2$
```

- \$HOME/dataproc2024/assignments/04_2/problem5.py
- \$HOME/dataproc2024/assignments/04_2/problem4.py
- Extra credit:
 - \$HOME/dataproc2024/assignments/04_2/problem5.py
 - \$HOME/dataproc2024/assignments/04_2/problem6.py
 - \$HOME/dataproc2024/assignments/04_2/problem7.py

Task 4 (Problem 03)

Rearranging, sorting, filtering

- `$HOME/dataproc2024/assignments/04_2/problem3.py`
 - (see/copy: `$DP24/assignments/04_2/problem3.py`)
- Write a script that:
 - Opens/reads file specified in command line argument.
 - Example file: `exfile2.txt` (see the format)
 - Format:
 - 1 header row!
 - separated by commas (,)
 - line format: Name,Surname,AgeYrs,StudentID,University
- **Output: filename specified by second command line argument**
 - No header
 - One line containing "FamilyName GivenName" for each **student** (i.e. input records with a non-null university and ID)
 - Lines are in ALPHABETICAL ORDER (a-z) for by family name (ignore case of letters i.e. a=A z=Z)

Format of exfile2.txt

```
Name, Surname, AgeYrs, StudentID, University
Bob, Saget, 21, 29392002020, Kyoto University
Johnny, Depp, 60, ,
Nicholas, Cage, 65, ,
Miki, Yawata, 21, B03989020202, Kyoto University
Don, Draper, 35, ,
Arnold, Schwarzanegger, 75, ,
Harvey, Specter, 40, ,
Kris, Donalds, 24, 77777229, Tokyo University
Kagami, Matthews, 18, 20019020001920, Harvard University
```

Task 5 (problem 04)

Filtering

- `$HOME/dataproc2024/assignments/04_2/problem4.py`
 - (see/copy: `$DP24/assignments/04_2/problem4.py`)
- Write a script that:
 - Opens/reads file specified in command line argument.
 - Example file: `exfile2.txt` (see the format)
 - Format:
 - 1 header row!
 - separated by commas (,)
 - line format: Name,Surname,AgeYrs,StudentID,University
- **Output: print to stdout the number of people whose family names have 10 or more characters.**

Extra Credit

Task 6 (Extra Credit 1)

Binary, Efficient Storage

- `$HOME/dataproc2024/assignments/04_2/problem5.py`
 - (see/copy: `$DP24/assignments/04_2/problem5.py`)
- Write a script that:
 - Opens/reads file specified in command line argument.
 - Example file: `exfile2.txt` (see the format)
 - Format:
 - 1 header row!
 - separated by commas (,)
 - line format: Name,Surname,AgeYrs,StudentID,University
- **Output: Binary file named "ages.out":**
 - **First 4 bytes being 'a', 'g', 'e', 's'.**
 - **5th byte's first bit tells whether entries will be big- (1) or little- (0) endian. Yours will always be big- endian (1)**
 - **5th byte's other 7 bits represents as an unsigned integer the number of bytes *B* per entry that will follow (in your case, always set $B=1$)**
 - **6th byte until end of file: arbitrary number of entries, each *B* bytes long, which are unsigned integers representing the age of each person in the input file in years. These should be sorted from SMALLEST to LARGEST age.**

Task 7 (Extra Credit 2)

Optimizing data size

- Modify the script from task 6 so that B is not necessarily 1 (name it: problem6.py).
- The script creates an ages.out file which uses the minimum number of bytes per age (B) necessary to represent the ages in the passed input file.

Extra things to think about:

What other optimizations could you do?

E.g.: < 1 byte per age (people usually not 255 years)

Task 8 (Extra Credit 3)

Maximum represented data limits

- Write a python script (name it: problem7.py) that prints to standard output the largest age that could possibly be represented in an ages.out file from the previous problem (Task 7).
 - Hint: it will be constant (think about B and how (unsigned) integers are represented...how does number of bytes relate to the number of possible values that can be represented)