Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencias de la Computación
IIC2143 - Ingeniería de Software 2020-2

Tarea Ruby

Indicaciones Generales

Esta tarea es de caracter **opcional**. Las personas que realicen esta tarea obtendrán hasta **5 décimas** extra en su primera evaluación escrita. **Hacerla está muy recomendado**, ya que es una gran oportunidad para aprender Ruby (lenguaje sobre el que se construye el framework Ruby on Rails utilizado durante el curso). **El nombre del archivo principal debe ser main.rb.**

Introducción

El equipo docente del ramo te ha pedido que programes, utilizando ruby, un programa que le permita gestionar el curso, en el cual se pueda ingresar la información de los ayudantes, junto con sus grupos y alumnos.

Modelación

Deberás modelar las siguientes clases: Persona, Alumno, Ayudante y Grupo. Tanto la clase Alumno como Ayudante deben heredar de la clase Persona.

Clase Persona

- **Atributos**: id, nombre
- **Subclases**:
  - **Clase Alumno**
    - **Atributos**: nota_i1, nota_i2, nota_ex, grupo
  - **Clase Ayudante**
    - **Atributos**: grupos

Clase Grupo

- **Atributos**: id, nota_e0, nota_e1, nota_e2, nota_e3, nota_ef, nota_presentacion

Aclaraciones y Consideraciones Importantes

- En la clase **Alumno**, el atributo **grupo** debe ser igual al **id del grupo al que pertenece**.
- En la clase **Ayudante**, el atributo **grupos** debe ser igual a **un array con los ids de los grupos que tiene a su cargo**
- Pueden haber quedado alumnos con su éxamen pendiente, por lo que nota_ex en ese caso quedaría igual a "P".
- Todos los alumnos forman parte de un solo grupo.
- No todos los ayudantes tienen la misma cantidad de grupos, pero todos tienen, al menos, uno.
- Todos los grupos tienen tres alumnos.

Métodos

Para demostrar que implementaron de manera correcta cada método, deberán crear un archivo de output (más sobre este archivo en las secciones finales). En este deberán escribir datos según el respectivo método a modo de logs. Más detalles a continuación.

aprobados(): Método que escribe en el archivo de *output* un listado de todos los alumnos que aprobaron el curso (más abajo se explican los criterios de aprobación y reprobación), ordenados de menor a mayor según su id. El *output* debe tener el siguiente formato:

COMIENZA APROBADOS
{id alumno 1} {nombre alumno 1} {nota final alumno 1}
{id alumno 2} {nombre alumno 2} {nota final alumno 2}
⋮
{id alumno n} {nombre alumno n} {nota final alumno n}
TERMINA APROBADOS

reprobados(): Método que escribe en el archivo de *output* un listado de todos los alumnos que reprobaron el curso (más abajo se explican los criterios de aprobación y reprobación), ordenados de menor a mayor según su id. El *output* debe tener el siguiente formato:

COMIENZA REPROBADOS
{id alumno 1} {nombre alumno 1} {nota final alumno 1}
{id alumno 2} {nombre alumno 2} {nota final alumno 2}
⋮
{id alumno n} {nombre alumno n} {nota final alumno n}
TERMINA REPROBADOS

proyecto(): Método que escribe en el archivo de *output* un listado de toda la información relevante del proyecto (ayudante, junto a sus grupos y sus integrantes). Tanto los ayudantes, como los grupos y sus integrantes deben aparecer en orden de menor a mayor según su id. El *output* debe tener el siguiente formato:

COMIENZA PROYECTO
ayudante:
{id ayudante 1} {nombre ayudante 1}

grupo:
{id grupo 1}
integrantes:
{id alumno 1} {nombre alumno 1} {id alumno 2} {nombre alumno 2} {id alumno 3} {nombre alumno 3}
grupo:
{id grupo 2}
integrantes:
{id alumno 4} {nombre alumno 4} {id alumno 5} {nombre alumno 5} {id alumno 6} {nombre alumno 6}
ayudante:
{id ayudante 2} {nombre ayudante 2}
grupo:
{id grupo 3}
integrantes:
{id alumno 7} {nombre alumno 7} {id alumno 8} {nombre alumno 8} {id alumno 9} {nombre alumno 9}
grupo:
{id grupo 4}
integrantes:
{id alumno 10} {nombre alumno 10} {id alumno 11} {nombre alumno 11} {id alumno 12} {nombre alumno 12}
⋮
TERMINA PROYECTO

top_grupos(): Método que escribe en el archivo de *output* un listado de los 3 proyectos con las mejores notas, ordenados según su id. El *output* debe tener el siguiente formato:

COMIENZA TOP GRUPOS
{id grupo 1} {nota proyecto grupo 1 }
{id grupo 2} {nota proyecto grupo 2 }
{id grupo 3} {nota proyecto grupo 3 }
TERMINA TOP GRUPOS

top_ayudante(): Método que escribe en el archivo de *output* la información del ayudante cuyos grupos lograron obtener el mejor promedio de notas, es decir, el mejor promedio entre las notas finales de proyecto de los grupos de dicho ayudante. El *output* debe tener el siguiente formato:

COMIENZA TOP AYUDANTE
{id ayudante 1} {nombre ayudante 1} {promedio proyecto ayudante 1}
TERMINA TOP AYUDANTE

Cálculo de Nota y Criterios de Aprobación

Para efectos de la tarea, se consideran dos notas importantes al calcular la nota final: **NC** y **NP**:

NC representa la nota de cátedra y se calcula como

$$(I1 + I2 + Ex)/3$$

Siendo:

- I1 la nota obtenido en la interrogación 1
- I2 la nota obtenida en la interrogación 2

- Ex la nota obtenida en el examen

NP representa la nota de proyecto y se calcula como:

$$0.5 * (E0 + E1 + E2 + E3)/4 + 0.2 * EF + 0.3 * P$$

Siendo:

- E0 la nota obtenida en la entrega 0
- E1 la nota obtenida en la entrega 1
- E2 la nota obtenida en la entrega 2
- E3 la nota obtenida en la entrega 3
- EF la nota obtenida en la entrega final
- P la nota obtenida en la presentación

Para aprobar el curso, tanto **NC** como **NP** deben ser **mayores o iguales a 4.0**. En el caso de que esto se cumpla, la nota final en el curso (**NF**) se calcula como:

$$0.5 * NC + 0.5 * NP$$

En el caso de que **NC** o **NP sea menor a 4.0**, **NF** se calcula como:

$$MIN(3.9,\ 0.5 * NC + 0.5 * NP)$$

En el caso de que un alumno haya quedado con una "P" en el examen, no se considera como aprobado ni reprobado.

Archivos CSV + TXT

Para cargar los datos, se les entregarán los siguientes archivos en formato csv: *alumnos.csv*, *ayudantes.csv* y *grupos.csv*. Adicionalmente, habrá un cuarto archivo en formato txt que contendrá las instruciones que serán ejecutadas con el programa separadas por líneas. Este archivo se entregará en la línea de comandos como un **argv** (más sobre este archivo en las últimas secciones). Los archivos tienen la siguiente estructura:

alumnos.csv

id,nombre,nota_i1,nota_i2,nota_ex,grupo
0,Alexander Rovint,7.0,6.3,6.3,0
1,Elliot Alderson,5.4,6.0,4.1,7
2,Steven Spielberg,3.2,4.5,P,2

ayudantes.csv

id,nombre,grupos
0,Dwayne Johnson,0:1:2
1,John Hurt,3:4:5
2,Matt Smith,6:7

grupos.csv

id,nota_e0,nota_e1,nota_e2,nota_e3,nota_ef,nota_presentacion
0,7.0,6.5,7.0,6.8,6.7,6.0
1,7.0,7.0,7.0,6.8,7.0,6.5
2,6.0,5.5,6.5,7.0,6.5,5.8
3,6.5,4.3,5.5,6.0,5.8,5.2
4,7.0,7.0,7.0,7.0,7.0,7.0
5,5.5,5.5,7.0,6.5,6.5,6.0

El archivo de las instrucciones tendrá la siguiente forma:

aprobados
reprobados
top_ayudante
top_grupos
proyecto

En otras palabras, será una serie de líneas, cada una conteniendo una instrucción. **Pueden haber instrucciones repetidas. Pueden faltar instrucciones. El orden es totalmente arbitrario**. Por temas de espacio, **en los ejemplos entregados arriba puede que hayan referencias a ids de grupos que no existen, pero en la tarea eso no ocurrirá**.

Ejecución

Para corregir la tarea, los ayudantes ejecutarán el siguiente comando:

ruby main.rb {datasets_folder} {instructions_file} {output_file}

Los archivos .csv con los datos siempre se llamarán según lo especificado en la sección Archivos CSV + TXT, pero irán dentro de la carpeta {datasets folder}. El archivo de instrucciones tendrá el nombre especificado en {instructions file} y el archivo al cual escribir el output tendrá el nombre especificado en {output file}.

Para dejar esto más claro, si nuestro repositorio tuviera la siguiente forma:

repo/
− − | data/
− − − − −| alumnos.csv
− − − − −| ayudantes.csv
− − − − −| grupos.csv
− − | instr.txt
− − | main.rb

El comando a correr sería:

ruby main.rb data instr.txt output.txt

Esto debería generar un archivo output.txt con los outputs esperados, dejando el repositorio con la siguiente forma:

repo/
− − | data/
− − − − −| alumnos.csv

– – – – –| ayudantes.csv
– – – – –| grupos.csv
– –| instr.txt
– –| main.rb
– –| output.txt

Importante: la corrección se realizará de manera automatizada, por lo que se espera que el formato que utilicen para el archivo de *output* sea exactamente el que se pide en el enunciado. Para facilitar su desarrollo, se les entregará un archivo de prueba que pordrán utilizar para verificar que, efectivamente, están cumpliendo con esto.

Foro

Cualquier duda que tengan sobre la tarea pueden preguntarla en las https://github.com/IIC2143-2020-2/syllabus/issues del repo del curso escribiendo en el título "[Tarea] - Pregunta...".

Entrega

La entrega de esta tarea se realizará mediante un *assignment* en Canvas, donde deberán subir un archivo .zip **sin** archivos .csv ni txt, dentro del cual deben incluir el o los archivos que utilicen en su programa. Tienen hasta el viernes 28 de Agosto a las 23:59 para entregar su tarea.