Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación
IIC2143 — Ingeniería de Software 2022-1

Entrega: 24 de marzo a las 23:59:59 horas

Tarea Ruby

1. Entrega

- **Fecha y Hora: 24 de marzo a las 23:59:59 horas**
- **Lugar:** *assignment* de Canvas.

Indicaciones Generales

Las personas que realicen esta tarea obtendrán hasta **5 décimas** extra en su primera evaluación escrita. Si bien la tarea es de carácter **opcional**, **hacerla está muy recomendado**, ya que es una excelente oportunidad para aprender Ruby, lenguaje sobre el cual se construye el framework Ruby on Rails utilizado durante todo el curso. IMPORTANTE: El carácter de esta tarea es **100 % INDIVIDUAL**

2. Introducción

Al momento de aprender cosas nuevas, es muy útil tener de referencia algo que conozcas. Es por eso que en esta ocasión, vamos a aprender ruby basarnos en el famoso juego Minecraft, más especificamente en algo que no todos conocen (o aprovechan) al momento de jugar, los intercambios con los aldeanos! Obviamente, adaptado ligeramente para que calce con el curso, es decir, intercambiando *rubíes* ;)

Deberás modelar clases y crear métodos que te permitan simular algunas funcionalidades del juego **Ruby-Craft**.

3. Modelación

Deberás modelar las siguientes clases:

`Jugador, Aldeano, Granjero, Herrero, Bibliotecario, Item, Intercambio`.

Si bien la modelación presentada a continuación es sugerida, pero no obligatoria, **sí se debe tener todas las clases escritas en esta sección.**

`Clase Jugador`

Atributos

- `id`: int
- `nombre`: str
- `experiencia`: int
- `reputacion`: float
- `rubies`: int

`Clase Aldeano`

Atributos

- `id`: int
- `name`: string
- `experiencia`: int
- `nivel`: int

Subclases

- `Clase Granjero`
  - **Atributos**
    - `descuento`: float
- `Clase Herrero`
  - **Atributos**
    - `descuento`: float
- `Clase Bibliotecario`
  - **Atributos**
    - `descuento`: float

`Clase Item`

Atributos

- `id`: int

- `nombre`: str
- `categoria`: int
- `subcategoria`: int
- `preciobase`: int

Clase Intercambio

Atributos

- `id`: int
- `jugador`: int
- `objetos`: list

4. Aclaraciones y Consideraciones importantes

- En la clase **Jugador**, el atributo **reputacion**, es un float entre -0.5 y 0.5.
- En la clase **Jugador**, el atributo **rubies** nunca puede quedar en un valor negativo.
- En la clase **Aldeano**, el atributo **nivel**, es un int del 1 al 5, que depende de la experiencia segun el tipo de aldeano.
- En las subclases **Granjero, Herrero** y **Bibliotecario**, el atributo **descuento** tiene valor 0.6, 0.5 y 0.35.
- En la clase **Item**, el atributo **categoria** hace referencia a qué tipo de aldeano lo vende con descuento, tiene valor **1** si es el **Granjero**, **2** si es el **Herrero** y **3** si es el **Bibliotecario**.
- En la clase **Item**, el atributo **subcategoria** hace referencia al nivel mínimo que debe tener un aldeano para venderlo.
- En la clase **Item**, el atributo **preciobase** es el precio al que se le deben aplicar todos los descuentos y/o aumentos al momento de hacer el intercambio.
- En la clase **Intercambio**, el atributo **jugador** hace referencia al **id** del jugador que está haciendo los intercambios.
- En la clase **Intercambio**, el atributo **objetos** debe ser igual a **un array de arrays, donde cada array corresponde al id del objeto junto al id del aldeano al que se le está intentando comprar**.

5. Descripciones

Antes de ponernos a programar, hay ciertas cosas que debemos tener claro de los aldeanos y los jugadores.

5.1. Tipos de aldeanos

1. Granjero

   Este aldeano es de los más fáciles de encontrar, se especializa en los cultivos y en la comida que se obtiene de estos. Tiene un descuento del 60 % para este tipo de items. Sube rápidamente de nivel, alcanzando el nivel 2 con 10 xp, el nivel 3 con 40 xp, el nivel 4 con 70xp y el nivel 5 con 100 xp.

2. Herrero

Este aldeano es más duro, ya que sus items son los más codiciados, se especializa en las herramientas y los minerales. Tiene un descuento del 50 % para este tipo de items. Sube muy lentamente de nivel, alcanzando el nivel 2 con 20 xp, el nivel 3 con 60 xp, el nivel 4 con 100xp y el nivel 5 con 150 xp.

3. Bibliotecario

Este aldeano suele estar oculto la mayor parte del tiempo, se especializa en los libros y en los encantamientos. Tiene un descuento solamente del 35 % para este tipo de items. Sube de nivel más rápido que los herreros, pero no tanto como los granjeros. Alcanza el nivel 2 con 15 xp, el nivel 3 con 50 xp, el nivel 4 con 85xp y el nivel 5 con 125 xp.

5.2. Precios

Ya mencionamos que los aldeanos ofrecen descuentos al intercambiar items de su especialidad, pero esa no es la unica forma de variar el precio. Como verás, los aldeanos viven en una comunidad muy cercana, y son bastante copuchentos, por lo que el comportamiento de un jugador con un aldeano puede afectar sus futuros precios con todos los aldeanos.

La forma de calcular esto es usando la reputación de un jugador, como fue mencionado más arriba, este es un valor entre -0.5 y 0.5, siendo -0.5 la peor reputación, 0.5 la mejor, y 0 neutral. Este valor afecta al precio base ponderandose este precio por 1 - reputación. De esta forma, el precio final sería:

$$precio_f = precio_i * (1 - reputacion) * (1 - descuento)$$

en caso de que se venda con descuento, y:

$$precio_f = precio_i * (1 - reputacion)$$

en caso de que no.

IMPORTANTE: este precio debe ser redondeado al **entero más cercano**

5.3. Cambios de Reputación

Ahora probablemente te estes preguntando cómo cambia la reputación, debes saber que hay 3 formas.

1. Al realizar un intercambio con un aldeano exitosamente, la reputación del jugador aumentará en 0.1, ya que está demostrando que es confiable.
2. Al fallar un intercambio por falta de rubíes, el aldeano pensará que lo estan intentando de estafar, por lo que la reputación disminuira en 0.15
3. Al fallar un intercambio por falta de nivel del aldeano, la reputación del jugador disminuirá en 0.1 (lo hicieron admitir su falta de nivel y le dió vergüenza...)

5.4. Experiencia

Finalmente, al completar un intercambio, tanto el jugador como el aldeano ganan experiencia.

El jugador ganará 5xp, y el aldeano ganará 10xp, pudiendo subir de nivel, lo que desbloquearía nuevos items para intercambiar.

6. Métodos

Para comprobar que las interacciones entre clases se realizaron de manera correcta, se deberá crear un archivo de *output*. En este se deberán escribir datos según el respectivo método a modo de logs. Más detalles a continuación según el método.

- `informacion_aldeanos()`: Este método se encarga de escribir en el archivo de *output* un listado de todos los aldeanos y los items que intercambian, ordenados por tipo (Primero Granjeros, después Herreros y finalmente Bibliotecarios). Dentro de cada tipo, deben estar ordenados de menor a mayor según el id del aldeano, y los items van ordenados de menor a mayor segun el id de estos. Los precios deben salir con el descuento correspondiente. El *output* debe tener el siguiente formato:

```
COMIENZA INFORMACION ALDEANOS
{id aldeano 1} - {tipo aldeano 1} {nombre aldeano 1} (lvl {nivel aldeano 1}):
({id item 1}) {nombre item 1}: {precio item 1}
({id item 2}) {nombre item 2}: {precio item 2}
.
.
.
({id item n}) {nombre item n}: {precio item n}
{id aldeano 2} - {tipo aldeano 2} {nombre aldeano 2} (lvl {nivel aldeano 2}):
({id item 1}) {nombre item 1}: {precio item 1}
({id item 2}) {nombre item 2}: {precio item 2}
.
.
.
(id item n) {nombre item n}: {precio item n}
.
.
.
{id aldeano m} - {tipo aldeano m} {nombre aldeano m} (lvl {nivel aldeano m}):
({id item 1}) {nombre item 1}: {precio item 1}
({id item 2}) {nombre item 2}: {precio item 2}
```

```
.
.
.
(id item n) {nombre item n}: {precio item n}
FIN INFORMACION ALDEANOS
```

ACLARACIONES: Para que un item sea considerado en esta lista, el aldeano debe poder venderlo con descuento (es decir, calzan los tipos) y además, tener un nivel lo suficientemente alto para hacerlo

- `realizar_intercambios()`: Este método se encarga de procesar todos los intercambios del archivo `intercambios.csv`, y escribir en el archivo de *output* los resultados de todos estos. El *output* debe tener el siguiente formato:

```
COMIENZA REALIZAR INTERCAMBIOS
{linea 1}
{linea 2}
.
.
.
{linea n}
FIN REALIZAR INTERCAMBIOS
```

Una linea puede ser cualquiera de las siguientes opciones:

Si no se realiza el intercambio porque el nivel del aldeano no es suficiente, la linea será

```
El jugador {nombre jugador} intento comprar {nombre item} del {tipo aldeano}
↪  {nombre aldeano} y no pudo porque el aldeano no tiene el nivel suficiente
```

Si no se realiza el intercambio porque los rubies del jugador no son suficientes, la linea será

```
El jugador {nombre jugador} intento comprar {nombre item} del {tipo aldeano}
↪  {nombre aldeano} y no pudo porque no tiene rubies suficientes
```

Si se realiza el intercambio, la linea será

```
el jugador {nombre jugador} compro {nombre item} del {tipo aldeano} {nombre
↪  aldeano} por {precio final} rubies
```

Adicionalmente, hay una linea extra que puede aparecer en los logs, y sucede cuando al finalizar un intercambio un aldeano sube de nivel, esa linea será

```
{Nombre aldeano} subio de nivel! ({nivel viejo} -> {nivel nuevo})
```

- `top_jugadores(criterio)`: Este método se encarga de escribir en el archivo de *output* un ranking de las los jugadores segun el parametro çriterio", si este parametro es rubies", los jugadores son ordenados de mayor a menor según la cantidad de rubies que tienen, y si el criterio es .^{ex}periencia", serán ordenados de mayor a menor segun la cantidad de experiencia que tienen. El *output* debe tener el siguiente formato:

```
COMIENZA TOP JUGADORES - {CRITERIO}
{nombre jugador 1}: {cantidad rubies/cantidad experiencia jugador 1}
{nombre jugador 2}: {cantidad rubies/cantidad experiencia jugador 2}
```

```
.
.
.
{nombre jugador n}: {cantidad rubies/cantidad experiencia jugador n}
FIN TOP JUGADORES
```

7. Archivos CSV + TXT

Para cargar los datos se subirán al repositorio oficial del curso los siguientes archivos en formato csv: `jugadores.csv`, `aldeanos.csv`, `objetos.csv`, `intercambios.csv`. Adicionalmente, se te entregará un archivo adicional en formato txt que contendrá las instrucciones 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 la sección 8. Ejecución).

Los archivos tienen la siguiente estructura:

`jugadores.csv`

Este archivo contiene la información de los jugadores que interactuan con los aldeanos. Cada fila contiene el id del jugador, el nombre, la experiencia, la reputacion y los rubíes que posee en su inventario, separados por una coma (`","`). Cabe destacar que el id es único y no se repetirá por jugador. Un ejemplo del archivo `jugadores.csv` es el siguiente:

```
id,nombre,experiencia,reputacion,rubies
1,Felipe,100,0.5,64
2,Josefa,150,0.3,32
3,Maggie,120,-0.2,45
4,Catalina,20,0,100
5,Nicole,70,0.3,8
```

`aldeanos.csv`

Este archivo contiene la información de los aldeanos que trabajan en la aldea. Cada fila contiene el id del aldeano, un numero que representa su tipo (1: Granjero, 2: Herrero, 3: Bibliotecario), y su experiencia, separados por una coma (`","`). Cabe destacar que el id es único y no se repetirá por aldeano. Un ejemplo del archivo `aldeanos.csv` es el siguiente:

```
id,nombre,tipo,experiencia
1,Ignacio,1,5
2,Martin,2,192
3,Andres,3,101
4,Carlos,1,39
.
.
.
```

`intercambios.csv`

Este archivo contiene la información de los intentos de intercambios realizados entre los jugadores y los aldeanos (IMPORTANTE: que un intercambio este en este archivo no significa que haya sido exitoso!).

Cada fila contiene el id del intercambio, el id del jugador que realizó el intercambio, y los objetos comprados separados por dos puntos (":") junto con el id del aldeano al que se le intentó comprar ("-"), todo separado por una coma (","). Cabe destacar que el id es único y no se repetirá por intercambio. Un ejemplo del archivo `intercambios.csv` es el siguiente:

```
id,jugador,objetos
1,1,1-8:5-4:11-1
2,3,1-9
3,5,27-3:1-5
4,3,29-9
5,1,16-7:19-8:1-1
```

Por ejemplo, el set de intercambios con id 1, el jugador que los realizó fue el jugador con id 1, este intento comprarle el objeto con id 1 al aldeano con id 8, el objeto con id 5 al aldeano con id 4, y el objeto con id 11 al aldeano con id 1.

`objetos.csv`

Este archivo contiene la información de los objetos que pueden ser comprados. Cada fila contiene el id del objeto, el nombre, la categoria, la subcategoria y el precio base del objeto separado por una coma (","). Cabe destacar que el id es único y no se repetirá por objeto. Un ejemplo del archivo `objetos.csv` es el siguiente:

```
id,nombre,categoria,subcategoria,preciobase
1,Trigo,1,1,10
2,Galleta,1,4,15
3,Pedernal,2,2,20
4,Hacha Hierro,2,3,20
5,Papel,3,1,5
6,Libro Encantado,3,5,30
```

El archivo de instrucciones tendrá la siguiente forma:

```
informacion_aldeanos
hacer_intercambios
top_jugadores_rubies
top_jugadores_experiencia
```

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

8. Ejecución

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

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

Los archivos .csv con los datos siempre se llamarán según lo especificado en la sección 7. Archivos CSV + TXT, pero irán dentro de la carpeta `{datasets_folder}`. El archivo instrucciones tendrá el nombre

especificado en `{instructions_file}`, y el archivo al cual escribir el output tendrá el nombre especificado en `{output_file}`.

De manera visual, si nuestro repositorio tuviera la siguiente forma:

```
repo/
----|data/
--------|aldeanos.csv
--------|intercambios.csv
--------|jugadores.csv
--------|objetos.csv
----|instr.txt
----|main.rb
```

El comando a correr sería:

```
ruby main.rb data instr.txt output.txt
```

Generando un archivo `output.txt` con los outputs esperados, dejando el repositorio de la siguiente forma:

```
repo/
----|data/
--------|aldeanos.csv
--------|intercambios.csv
--------|jugadores.csv
--------|objetos.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 podrán utilizar para verificar que, efectivamente, están cumpliendo con esto.

9. Foro

Cualquier duda que surja respecto a la tarea pueden preguntarla en el foro oficial del curso, **escribiendo en el título de la issue "[Tarea] - Contenido pregunta"**.

10. Entrega

La entrega de la tarea se realiza mediante un *assignment* en Canvas, donde se deberá subir un archivo .zip **sin contener archivos .csv ni .txt**, dentro del cual deben incluir el o los archivos que utilicen en su programa. El plazo es hasta el jueves 24 de marzo a las 23:59:59 horas.