Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación
Algoritmos y Programación (6201)
Semestre (2 -2024)

Proyecto #2

GREMIO DE AVENTUREROS

Tras múltiples incursiones en el laberinto, el gremio de aventureros ha registrado un alto número de bajas debido a los numerosos aventureros atrapados o fallecidos en su interior. En respuesta a esta situación, el gremio ha decidido llevar a cabo una investigación exhaustiva sobre todos los aventureros del país, generando un conjunto de archivos con información detallada sobre cada uno de ellos.

Estos archivos han sido creados gracias a una piedra mágica muy antigua creada por los dioses de otra época, esta piedra fue encontrada en el laberinto, está al entrar en contacto con un aventurero y una superficie, graba automáticamente los atributos de dicho aventurero en ese momento, el gremio utilizando papel ha generado archivos para todos los aventureros.

El formato de estos registros varía considerablemente entre individuos, pero se ha identificado un patrón común que sigue la siguiente estructura:

1. **Información Básica:**
   Cada registro comienza con tres datos fundamentales:
   - `"Clase:"` **:** Define el tipo de aventurero.
   - `"Faccion:"` Puede incluir una o varias afiliaciones separadas por el símbolo " **|** ".
     Ejemplo: "Faccion: Magallaneros | Caraquista".
   - `"Nombre:"` El nombre del aventurero.
2. **Atributos (ATRIB):**
   Tras la información básica, aparece la palabra clave **"ATRIB"**, seguida de una lista de atributos con valores numéricos en el formato `"Atributo: Cantidad"`.
   Ejemplo: `"Fuerza: 999"`.
   La variedad de atributos por aventurero no es estándar, ya que con cada nuevo aventurero se descubren nuevos parámetros como *Magia, Inteligencia, Agilidad, Vitalidad*, entre otros, lo que se sabe, en la actualidad se han identificado solamente 100 atributos.
3. **Código Genético (ADN):**
   Después del listado de atributos, el archivo siempre incluye la palabra clave **"ADN"**, seguida de una matriz numérica cuadrada impar. Esta matriz representa un código único para cada aventurero. Al ser procesada, permite calcular un **“Puntaje”**, donde un número mayor indica un aventurero de mayor nivel. y dicho puntaje se vuelve un atributo.

Ejemplo de ADN:

5	2	3	2	1
5	2	3	2	1
5	2	3	2	1
5	2	3	2	1
5	2	3	2	1

El cálculo del puntaje es el siguiente:

$$puntaje = \left| \frac{\prod_{i=0}^{n} (i = números\ no\ coloreados)}{4} - \sum_{i=0}^{n} (i = números\ coloreados) \right|$$

Ejemplo:

5	2	3	2	1
5	2	3	2	1
5	2	3	2	1
5	2	3	2	1
5	2	3	2	1

5 + 2 + 3 + 2 + 1 + 1 + 2 + 2 + 5 + 3 + 3 + 3 + 3 + 5 + 2 + 2 + 1 = **43**

(Se utilizaron colores distintos para diferenciar las diagonales)

(2 * 2 *1 * 1 * 2 * 2 * 5 *5) / 4 = **100**

$puntaje = |100 - 43| = 57$ **// Puntaje del Aventurero.**

Ejemplo completo de Archivo de Aventurero de puntaje 57:

```
Clase:Guerrero
Faccion: Tribu Amacional | Nacionales
Nombre: Procrastinator Duplicarus
ATRIB
Fuerza:5
Destreza:5
Agilidad:5
Constitucion:5
Inteligencia:5
Sabiduria:5
Carisma:5
Magia:5
Energia:5
Voluntad:5
Edad:25
ADN
5 2 3 2 1
5 2 3 2 1
5 2 3 2 1
5 2 3 2 1
5 2 3 2 1
```

Conociendo todo esto, el gremio de aventureros le solicita colaboración a los estudiantes de materia para desarrollar un software que tenga las siguientes funcionalidades:

1. **CARGAR:** esta función recibe un string con la dirección física de la carpeta donde se encuentra la base de todos los aventureros **(la dirección siempre será válida)**.

   Ejemplos:

CARGAR “../../../../proyectos/aventureros/.*”
CARGAR “C:\Users\Usuario\OneDrive\Desktop\Aventureros”
CARGAR “.\Aventureros”

2. **BUSCAR**: Permite hacer búsquedas entre los aventureros. Cada línea luego de indicar la función está conformada por 2 parámetros: el primer parámetro **(clase, nombre, facción)** seguido de una cadena de caracteres (delimitada con comillas simples ‘) por cada criterio de búsqueda para realizar las búsquedas. Dicha búsqueda inicia con la palabra **INICIAR**.

Ejemplo:

```
BUSCAR
nombre ‘procas’
faccion ‘aciona’
INICIAR
```

3. **ORDENAR**: modifica el orden en que se manejan los datos, cada línea luego de indicar la función está conformada por 2 parámetros, siguiendo el formato **atributo símbolo**. Dicha función se ejecuta con la palabra INICIAR. Si el atributo no se encuentra en uno o más de los aventureros, se procede a considerar el siguiente, habiendo dejado ordenados los del atributo anterior. En caso de que el aventurero no tenga ninguno de los atributos considerados, se elimina del resultado.

   Ejemplo:

```
ORDENAR
fuerza >
inteligencia  <
INICIAR
```

4. **SELECCIONAR:** filtra el listado de aventureros que contengan los atributos señalados. cada línea luego de indicar la función está conformada por 1 o 3 parámetros, siguiendo el formato **atributo** o **atributo símbolo cantidad.** Considere que si un parámetro no posee **símbolo** ni **cantidad** debe priorizarse su presencia en los atributos del aventurero. La función se ejecuta con la palabra **INICIAR**.

   Ejemplo:

```
SELECCIONAR
fuerza < 5
carisma
INICIAR
```

5. **IMPRIMIR:** Dada la secuencia completa de acciones realizadas durante la ejecución del programa debe generarse en una carpeta “operaciones” con los archivos “operacionesX.out” donde X significa el número de operación guardada, dicho archivo debe contener un resumen y el resultado final de las operaciones, enumerando los aventureros.

Formato Ejemplo:

```
operaciones realizadas: N
aventureros encontrados: M
lista de aventureros:
#1
Puntaje: 57
Clase:Guerrero
Faccion: Tribu Amacional|Nacionales
Nombre: Procrastinator Duplicarus
Atributos
Fuerza:5
Destreza:5
Agilidad:5
Constitucion:5
Inteligencia:5
Sabiduria:5
Carisma:5
Magia:5
Energia:5
Voluntad:5
Edad:25
#2…
```

EJEMPLO DE APLICACIÓN DE FUNCIONES EN LA MISMA EJECUCIÓN:

Cada sección con letra en la tabla, simboliza un “Aventurero” leído desde la base, para el ejemplo visual.

	FUNCION 1		FUNCION 2		FUNCION 3	
Aventureros	ORDENAR		SELECCIONAR			
base completa	vida <	Aplicar primera funcion	magia	Aplicar segunda funcion	IMPRIMIR	operaciones1.out
	INICIAR		INICIAR			
	→		→		→	operaciones realizadas: 2 aventureros encontrados: 3 lista de aventureros: #1 A vida:5 magia:1 #2 G vida:5 magia:12 #3 E: vida:6 magia1
A		**A**		**A**		
vida:5		vida:5		vida:5		
magia:1		magia:1		magia:1		
B		**G**		**G**		
vida:14		vida:5		vida:5		
agilidad:1		magia:12		magia:12		
C		**E**		**E**		
vida:7		vida:6		vida:6		
inteligencia:4		magia:1		magia:1		
D		**C**				
vida:15		vida:7				
agilidad:1		inteligencia:4				
E		**F**		EJEMPLO PRACTICO DE APLICACION		
vida:6		vida:7		DE FUNCIONES - SALIDA REFERENCIAL		
magia:1		inteligencia:4				
F		**B**				
vida:7		vida:14				
inteligencia:4		agilidad:1				
G		**D**				
vida:5		vida:15				
magia:12		agilidad:1				

CASO #1 EJEMPLO:

Archivos /aventureros.

aventureroA.in

```
Clase:Guerrero
Faccion: Tribu Amacional | Nacionales
Nombre: Procrastinator Duplicarus
ATRIB
Fuerza:5
Destreza:5
Agilidad:5
Constitucion:5
Inteligencia:5
Sabiduria:5
Carisma:5
Energia:5
Voluntad:5
Edad:25
ADN
5 2 3 2 1
5 2 3 2 1
5 2 3 2 1
5 2 3 2 1
5 2 3 2 1
```

AventureroB.in

```
Clase:Mago
Faccion: Elfos Arcanos | AntiNacionales
Nombre: Lord Valdemoro
ATRIB
Fuerza:1
Destreza:2
Agilidad:2
Inteligencia:5
Sabiduria:23
Carisma:6
Magia:35
Energia:25
Voluntad:25
Edad:41
ADN
5 2 3 2 1
5 2 3 2 1
5 2 3 2 1
5 2 3 2 1
5 2 3 2 1
```

ENTRADA (Input console)

```
ORDENAR
magia
INICIAR
SELECCIONAR
sabiduria < 20
INICIAR
IMPRIMIR
```

operaciones1.out

```
operaciones realizadas: 2
aventureros encontrados: 1
lista de aventureros:
#1
Puntaje: 57
Clase:Mago
Faccion: Elfos Arcanos| AntiNacionales
Nombre: Lord Valdemoro
ATRIB
Fuerza:1
Destreza:2
Agilidad:2
Inteligencia:5
Sabiduria:23
Carisma:6
Magia:35
Energia:25
Voluntad:25
```

Consideración de funcionamiento:

- Los parámetros se aplican en cada función en el orden que son ingresados a las funciones.
- los símbolos para ordenamiento son **<** (menor que , menor a mayor), **=** , **>** (mayor que, de mayor a menor) , **#** (el # significado diferente de **!=**).

Condiciones de entrega , restricciones y evaluación:

- Toda lógica, estructuras lógicas, resolución de problema u operación debe ser implementada por el alumno, no se acepta utilización de librerías externas, exceptuando: cstdlib, stdlib, iostream, fstream, math.h o csmath**.**
- Se puede utilizar solamente el contenido visto en la materia.
- Todas las cadenas de caracteres en los archivos de entrada tienen como límite un tamaño de 100 caracteres.
- Su proyecto debe ser realizado usando el enfoque de programación orientada a objetos
- Todo el código debe ser entregado en un sólo archivo cpp.
- Debe realizarse un informe donde se explique la implementación, procesamiento , almacenado, análisis y enfoques utilizados, el mismo tiene ponderación en la evaluación y condiciona la evaluación del proyecto y adicionalmente, debe realizarse un diagrama de clase donde como **POO** en su código.
- El proyecto se puede realizar sólo con alumnos de la misma sección (individual o en pareja).
- El informe y el código debe ser entregado en un zip. con el siguiente formato:

  [AyP] C1-NOMBRE1-APELLIDO1-CEDULA1_NOMBRE2-APELLIDO2-CEDULA2.

  Este archivo comprimido debe ser enviado a las direcciones de correo: **aypucv@gmail.com**

  El asunto debe replicar el nombre del archivo, de no cumplir el formato pedido su proyecto será sancionado. destacar que C1 es una sección de ejemplo donde debe indicar la sección de los participantes.
  EJEMPLO: **C1-RAFFAELE-RANALDO-22333444_DANIEL-DA-COSTA-33444555.zip**
- La fecha de entrega de proyecto estipulada es: **05/03/2025 11:59pm,** la misma puede estar sujeta a modificación por el **GDAyP.**
- **Si su proyecto no es capaz de realizar lectura/salida por archivos el mismo NO SERÁ CORREGIDO.**
- **Si al realizar la evaluación del proyecto, el mismo tiene problemas de compilación este no podrá ser evaluado y la nota estará sujeta solo al informe.**
- **El no cumplir las condiciones generará la aplicación de penalizaciones.**
- **En caso de detección de copia y/o utilización de herramientas de IA, se le asignará la calificación mínima, sin posibilidad de apelación además de sanciones adicionales.**