75.41 95.15 95.12 Algoritmos y Programación II Curso 4
TP2 - Simulador de entrenamiento Pokemon

3 de diciembre de 2021

1. Introducción

Gracias a tus esfuerzos, el hospital Pokemon quedó equipado con uno de los mejores sistemas de gestión de pacientes de la región. Sin embargo ahora se presenta un nuevo desafío. El hospital fue ampliado para poder atender una mayor cantidad de Pokemon y esto trajo asociado algunos problemas de falta de personal. Se intentó contratar nuevas enfermeras Pokemon pero aún les falta experiencia. Se presenta entonces el desafío de desarrollar un sistema de simulación que permita, trabajando con el sistema de gestión de pacientes, ayudar al entrenamiento de las nuevas enfermeras para el tratamiento de Pokemon.

2. Tratamiento de los Pokemon

Como todos saben, es muy importante administrar el tratamiento adecuado a cada Pokemon, y parte de ello involucra aplicar la cantidad de medicación correspondiente según el nivel del Pokemon. Muchas veces (mas que nada por ego) los entrenadores mienten y exageran el nivel de sus Pokemon. El objetivo es lograr un sistema de simulación que permita a las enfermeras (basados en datos de pacientes reales y verificados) practicar y adquirir experiencia estimando el nivel de un Pokemon.

El simulador permitirá que las enfermeras intenten adivinar el nivel de un Pokemon y les indicará si adivinaron o no, guiándolas hacia la respuesta correcta. Las pistas recibidas y los puntos ganados varían según el nivel de dificultad seleccionado. Diferentes dificultades pueden también asignar diferente cantidad de puntos al llegar a la respuesta correcta.

2.1. Funcionalidad pedida y restricciones

Se pide implementar la funcionalidad del simulador presentada en el archivo **simulador.h** y los eventos especificados en **eventos.txt**. Adicionalmente se debe entregar un archivo **main.c** que haga uso del simulador en una aplicación por consola. Se aconseja que la aplicación funcione en base a un menú con el cuál se pueda interactuar ingresando opciones.

La idea de este trabajo es demostrar los conocimientos adquiridos en la materia y por lo tanto se alienta el uso de las estructuras y **TDA** vistos en clase para solucionar el problema de la mejor manera posible. Por lo tanto queda **terminantemente prohibido usar vectores estáticos o dinámicos** para almacenar datos. La única excepción son vectores dinámicos de char para almacenar strings.

2.2. Entrega

Una vez completada la implementación, tenes que subirla al sistema de entregas, como siempre, para verificar su correcto funcionamiento. Además de la implementación, tenes que escribir un informe y entregarlo (en formato PDF). En el informe **tenés que explicarle a tu corrector** qué estructuras usaste para resolver cada uno de los problemas y por qué resulta la mejor alternativa y una explicación de cómo funciona tu TP. **El informe es parte de la nota**.

Las recomendaciones de siempre (repetidas, claro) y otras nuevas:

- Buenas prácticas de programación y sarasa (modularización, código y variables con nombre claro, no duplicar código, etc).
- Una aplicación por consola debe permitirle al usuario en todo momento conocer qué es lo que está sucediendo. Tiene que ser, dentro de lo posible, intuitivo de utilizar. Utilice carteles de ayuda y descripciones que sean de utilidad.
- Pruebas, muchas pruebas.

- Si corres el programa y algo funciona mal, el primer paso es siempre escribir una prueba que reproduzca el error. Primero la prueba, después se arregla el error.
- Las pruebas son parte de la nota.
- Si se modifican **TDA** ya implementados, se deben incluir las pruebas de los mismos y agregar las pruebas de las nuevas funcionalidades. Además se debe explicar en el informe los cambios hechos y explicar su propósito.
- Las pruebas correspondientes al **TP1** deben correr correctamente.
- El **zip** debe contener **TODOS LOS ARCHIVOS NECESARIOS PARA QUE COMPILE**.