page1image16850816Final

Consignas

Coliseo - Gladiadores:

En un pueblo muy conocido de Italia, se desarrolla un campeonato dentro de un coliseo. En el mismo, habrá gladiadores esclavos por el tirano emperador que estarán luchando para conseguir su premio final, su libertad.

Los gladiadores pelearán contra los luchadores del emperador que defienden el imperio italiano, Por parte de los gladiadores son 3 luchadores. Y por parte de los luchadores gladiadores del imperio, son 5 luchadores. Cada uno de los luchadores tiene como valor nominal 100 de vida. Es una batalla por turnos. En cada turno, sean esclavos o luchadores del emperador, restaran de 1 a 5 de vida de los oponentes (valor aleatorio). Pero por parte de los esclavos, cuentan con Máximo, un gladiador especializado en armas, que resta de los oponentes entre 5 y 10 de vida de los oponentes.

Se contará con un array (arreglo) de estructuras, en la que cada estructura cuenta con el nombre del gladiador, su vida y su arma. O si lo cree conveniente utilizar memoria dinámica u otro método con/sin array/estructuras donde pueda colocar lo informado (mismo archivos si lo desea).

Sería recomendado para mejorar el sistema (no mandatorio) realizar un menu para que permita el ingreso por teclado de la descripción informada.

Deberá realizar dos procesos, donde usted decidirá si será un proceso el coliseo y otro proceso los luchadores (dos u ocho) o si realiza dos procesos (un para luchadores esclavos y otro luchadores del emperador). Para comunicar debe utilizar cola de mensajes (por ejemplo MSJ\_LUCHADOR+i).

Cada 200ms se debe pasar de turno.

Se debe informar que luchadores perecen, y cuales salen invictos en todo momento, en los procesos. También DEBE GUARDARSE EN UN ARCHIVO.

En caso de que los luchadores esclavos ganen, dentro del coliseo se encuentra el tirano emperador, quien decidirá si les da la libertad (a uno o los que hayan sobrevivido) mediante un valor aleatorio (de 1 a 2) y lo cual se informará al gladiador. Luego de esto los programas terminan.

Se debe mostrar la actividad de cada proceso.

Usar los eventos que crea necesario.

Pensar bien que/quien es cada proceso! Comenzar por lo mínimo, lo más importante es el sincronismo y que compile!.

Si utiliza memoria dinámica debe liberar la memoria!

Utilizar memoria compartida o un bucle, para siempre poder realizar las inicializaciones correspondientes, y no depender de correr un proceso determinado primero.

Utilizar Makefile con la estructura de archivos vista en la cursada.

Utilizar los delay (sleep) que correspondan.