Buen día Bento, ¿todo bien? Considero importante aclarar algunas cuestiones sobre mi trabajo, ya que capaz podrían generar duda.

Convenciones

Las convenciones que utilizo en el trabajo son con base en qué me parece más legible, siempre respetando el uso del snake_case .

- Con respecto al código, lo separo en tres partes:

```
...{
declaraciones;

código;

return;
}
```

En caso de no haber una de las partes, hago lo siguiente:

○ Si no hay declaraciones:

```
...{

código;

return;
}
```

○ Si no hay return:

```
...{
declaraciones;

código;
}
```

○ Si solo hay código:

```
...{

código;
}
```

La razón por la que dejo un enter arriba del código pero no debajo es debido a que cuando minimizo el bloque, no me gusta que quede una línea en blanco, ya que si hubiera un enter abajo quedaría como en el de la izquierda y prefiero el de la derecha:

...{
 	
}
	...{
	}

Añado: Es factible que dentro del 'Código' haya vuelto a separar en secciones dependiendo de lo que está realizando ese bloque de código.

Estructuras de control

● Uso del **do while**:

Para el ingreso de datos me parece muy práctico que vuelva a preguntar exactamente lo mismo que siempre, en caso de haber ingresado algo inválido, además del error cometido, y escribir la pregunta afuera del while y dentro del mismo me resulta repetitivo, es el por qué preferí usar do while en todos los casos.

● Uso del **if** y del **switch**:

En caso de tener más de dos situaciones, utilizo el switch, sino el if: me resulta más cómodo.

Estructura del programa

- Se RUEGA usar el programa en pantalla completa, sino no se podrá asegurar la correcta visualización del mismo.

- Las constantes las tengo alineadas en grupos, así me organizo mejor. Es decir, los valores de cada una los tengo a la misma altura solo si son del mismo grupo. Por ejemplo:

```
#define VELOCIDAD_MIN                0.1f
#define VELOCIDAD_MAX                10
#define VELOCIDAD_GRABACION_DEFAULT  1

#define LEER          "r"
#define ESCRIBIR      "w"
```

- La vida de las torres debe ser múltiplo de CANT_CASILLAS_CUADRO ya que sino no podría mostrar las barras de vida. Es una condición que le solicito al usuario cumpla. En caso de no hacerlo, por las dudas prefiero verificarlo, ya que llegado el caso que el valor no cumpla la condición, se podría dar que el programa intente dividir sobre 0, y es mejor prevenir que curar.

- En caso de ingresarse un comando no válido, como "grabacion=", o "grabacion" el programa mostrará al usuario los posibles comandos válidos; se usará la configuración

default (en este caso la no grabación) solo si el usuario ingresa un archivo inválido. Todos los comandos mal colocados serán tomados como eso: un comando inválido.

- Tanto si el comando es inválido como si hubo un problema de ejecución del mismo a causa del ingreso, se le mostrará al usuario todos los posibles comandos. Además, el usuario posee un "readme" el cual puede leer antes de usar el programa así tiene noción de cómo usarlo.

- Los nombres de archivos ingresados deben tener correcta extensión, ya que sino no serán válidos para usarse.

- Convierto en mayúscula la primera letra del nombre del usuario ya que sea cual sea el caso, el nombre o la inicial empezará con mayúscula. No convierto el resto de letras a minúscula ya que podría darse el caso de que el usuario coloque sus iniciales (en mi caso sería AGG), y no estaría bien gaurdarlo como "Agg".

- Todos los ingresos de caracteres son trabajados en mayúscula, perdonando el ingreso al usuario del valor en minúscula.

- Cada defensor posee su propio color, GIMLI/ENANOS con **MAGENTA**, LEGOLAS/ELFOS con **AZUL**.

- En "jugar", configuracion tiene guardado todos los valores ya que así se me facilita la obtención de los mismos.

- En "jugar", el cuadro a mostrar ha sido elaborado bajo un algoritmo que permite su creación sean cualesquiera sus valores.

- En "jugar" al mostrar el terreno en configuración default de caminos, se respetan las posiciones cardinales de las torres y las entradas, con una distancia mínima entre ellas para que no se pisen.

- En "jugar" al mostrar el terreno, este es pasto.

- En "jugar" se mostrarán los ataques causados en cada turno arriba del cuadro. Dentro del mismo se mostrará el nivel de vida de cada torre, los orcos que falten aparecer y los orcos en pantalla.

- En "jugar" los caminos pueden superponerse una vez, indicando esto con color **GRIS**.

- En "jugar" los orcos se mostrarán al usuario según su vida, siendo las posibilidades MUCHA_VIDA, MASO_VIDA o POCA_VIDA.

- En "jugar", si dos orcos se cruzan, he decidido mostrar ambos, siendo las posibles combinaciones:

- DOS_ENEMIGOS_MUCHA_VIDA_MUCHA_VIDA.

- DOS_ENEMIGOS_MUCHA_VIDA_MASO_VIDA.
- DOS_ENEMIGOS_MUCHA_VIDA_POCA_VIDA.
- DOS_ENEMIGOS_MASO_VIDA_MASO_VIDA.
- DOS_ENEMIGOS_MASO_VIDA_POCA_VIDA.
- DOS_ENEMIGOS_POCA_VIDA_POCA_VIDA.

- En "jugar", cuando un enano ataque, su golpe será completamente random: he elegido detectar todos las posibles victimas y atacar a una random entre ellas.

- En "jugar", al atacar primero se evaluará si el golpe será fallido, luego se determinará si el mismo será crítico o no: en caso que las probabilidades de ambos sean 100%, todos los golpes serán fallidos, ya que no habrá ataques para evaluar si serán críticos o no.

- En "crear_configuracion" permito que escriba el nombre de un archivo de camino que no exista ya que puede suceder que quiera crear el camino más tarde: igualmente se le avisa de esto al usuario en caso de no haberse percatado.

- En "crear_camino" el usuario obtiene la posición de las torres y las entradas en el mapa ya que así se respetan las coordenadas de cada nivel.

- En "crear_camino" se permite que el usuario navegue con las letras "**w**", "**a**", "**s**" ó "**d**" **+ enter** ya que el usuario puede llegar a ser impulsivo y así se le otorga la posibilidad de pensar mejor los ingresos. Además, para agilizar, puede ingresar muchas letras de una, por ejemplo "aaaaswwws".

- En ranking, si el usuario ingresa un valor mayor al posible en el comando "listar=", se le mostrarán todos los valores.

- En "animos.c" controlo preguntar y mostrar solo los datos que no tenga de antemano (en caso de que haya una configuracion personalizada, sino pregunto y muestro todo).

- En "animos.c" tengo varios comentarios en élfico para hacer más divertido el programa.

- En "animos.c" poseo una selección de respuestas random de error, para hacer más divertida e interesante la parte de preguntas.

- El TP cuenta con diversas bibliotecas, entre ellas "adornos" la cual es utilizada para las funciones que refieren a lo estético meramente, en la cual algunas funciones trabajan con la configuración del juego.

- Al comienzo de "jugar" hay una animación: se ha elaborado fotograma por fotograma a mano.

- Al comienzo de "jugar" hay barras de carga: las mismas son mera decoración, no hay nada cargando.

- Al comienzo de "jugar" hay una pantalla de instrucciones, es por esto que los datos allí mencionados no serán aclarados nuevamente: al jugar el usuario debe contemplar todo lo dicho allí.

- Los colores fueron elegidos con la idea de una interfaz gráfica **verde**-**negra**-**blanca**.