Departamento de Informática e Ingeniería de Sistemas
Universidad Zaragoza

Examen de Proyecto

22 de junio de 2023

Fundamentos de Informática
Grado en Tecnologías Industriales

Duración: 50m (más 15m para preparación y entrega)

Nombre y apellidos: _____ **NIA:** _____

Recuerda: El examen se realiza de forma individual.
Entrega:

- El examen se entrega en la tarea de Moodle preparada para ello.
- Se entrega un único archivo correspondiente al programa que implementes en este examen.
- Deberá compilar en las mismas condiciones que el proyecto original.

Ejercicio 1 **[10 puntos]**

En el **Ejercicio 3** del **Proyecto** se pidió un programa `pokemon.pas` en el que se trabajaba con ficheros de texto que almacenaban información sobre *pokemons*. Incluimos a continuación un recordatorio de los principales aspectos de dicho ejercicio, y tras ellos la modificación que se pide para el presente ejercicio.

Pokemon es un juego de tipo RPG (*role-playing game*) en el que el jugador debe recorrer una región en busca de convertirse en el nuevo campeón de la Liga Pokemon. Para lograrlo, debe enfrentarse a otros jugadores con un objetivo similar, y que se irán volviendo más fuertes conforme avanza la aventura. Para hacerles frente, el jugador debe capturar y entrenar diferentes pokemon a lo largo de su camino, y utilizarlos para afrontar tales desafíos. En los juegos originales, cada pokemon está definido por un conjunto de características, incluyendo estadísticas, naturaleza, tipo, movimientos, evoluciones, objetos, fortalezas, debilidades, y más (mucho más).

En este ejercicio, se pedía diseñar una versión muy simplificada (pero divertida) de estos juegos utilizando registros y ficheros de texto. Un pokemon estará definido en un fichero de texto como los siguientes:

charizard.txt	**meganium.txt**	**swampert.txt**	**blissey.txt**	**arceus.txt**
Charizard	Meganium	Swampert	Blissey	Arceus
240 90 55 120	320 45 80 100	260 110 90 80	550 40 40 60	350 150 110 110
80 Lanzallamas	60 Hoja Mágica	100 Hidrobomb	0 Amortiguador	100 Cometa draco
90 Onda ígnea	80 Gigradenado	70 Surf	0 Encanto	80 Velocidad extrema
30 Ascuas	20 Látigo cepa	90 Terremoto	80 Terratemblor	100 Puño meteoro
0 Gruñido	10 Alarido	10 Destructor	150 Hiperrayo	120 Sofoco

La estructura de cada fichero de texto es la siguiente:

- La primera línea corresponde a una cadena de texto (`string`) con el nombre del pokemon.
- La siguiente línea contiene cuatro números enteros indicando sus puntos de salud, ataque, defensa, y velocidad, en ese orden.
- Las cuatro siguientes líneas contienen, con uno por línea, la definición de los cuatro posibles movimientos de ese pokemon, especificados mediante su potencia (un entero) y su nombre (una cadena de texto o `string`).

En dicho Ejercicio 3 del Proyecto: (i) se leía un pokemon de un fichero de texto con el procedimiento `leerPokemon`, guardándolo en un dato de tipo `tpPokemon`, (ii) dada una estructura `tpPokemon`, se mostraba por pantalla la información de dicho pokemon con el procedimiento `mostrarPokemon`, (iii) un procedimiento `batallaPokemon` llevaba a cabo la simulación de una batalla entre dos pokemon (y la batalla consistía en un ataque de cada pokemon) y (iv) el programa principal pedía dos nombres de ficheros pokemon y realizaba una batalla entre ellos.

Se pide que modifiques el procedimiento `batallaPokemon` anterior para mostrar una batalla completa, de forma que los pokemon no realicen sólo un ataque cada uno, sino que sigan atacando de forma alterna hasta que uno de los dos tenga 0 o menos puntos de salud. En dicho momento, el programa mostrará por pantalla el ganador de la batalla.

En la batalla empezará atacando el pokemon más rápido (el de mayor valor de velocidad), y, tras resolverse completamente el primer ataque, atacará el pokemon más lento. Cada uno de los ataques tiene los siguientes pasos:

- El pokemon atacante usa uno de sus cuatro movimientos aleatoriamente (ver **Nota** debajo). El daño del ataque se calculará de la siguiente forma, a partir de los datos del pokémon atacante:
  ```
  dano = valor_ataque + potencia_del_movimiento
  ```
- El pokemon atacado recibirá el ataque. El daño que recibe se calculará de la siguiente forma:
  ```
  dano_recibido = dano - valor_defensa
  ```
  Un pokemon nunca recibirá menos de 0 de daño.
- El pokemon atacado perderá tantos puntos de salud como dano reciba:
  ```
  valor_salud = valor_salud - dano_recibido
  ```

Estos pasos se ejecutaran hasta que uno de los dos pokemon tenga 0 o menos puntos de salud. En cuanto el pokemon afectado reciba el suficiente daño se debera mostrar el mensaje por pantalla de victoria y el pokemon ganador correspondiente.

Todo lo demás debe tener la misma funcionalidad que en el programa original del Proyecto. La interacción con el usuario será la misma que en el Ejercicio 3 del Proyecto. A continuación se muestran varios ejemplos de ejecución:

Ejemplos de interacción (en negrita lo que introduce el usuario):

```
Introduce nombre del fichero con el primer pokemon: charizard.txt
Introduce nombre del fichero con el segundo pokemon: swampert.txt

Charizard ataca primero.
Charizard usa  Onda ignea.
Swampert recibe 90 de dano.

Swampert ataca despues.
Swampert usa  Surf.
Charizard recibe 125 de dano.

Estado de los pokemon:
Charizard
Salud: 115
Ataque: 90
Defensa: 55
Velocidad: 120

Swampert
Salud: 170
Ataque: 110
Defensa: 90
Velocidad: 80

Charizard ataca primero.
Charizard usa  Ascuas.
```

```
Swampert recibe 30 de dano.

Swampert ataca despues.
Swampert usa  Terremoto.
Charizard recibe 145 de dano.

Estado de los pokemon:
Charizard
Salud: 0
Ataque: 90
Defensa: 55
Velocidad: 120

Swampert
Salud: 140
Ataque: 110
Defensa: 90
Velocidad: 80

Swampert gana la batalla.
```

Nota: Para generar un número aleatorio puedes usar la función predefinida en Pascal:

```
function random(max: longint): longint;
```

que devuelve un número aleatorio en el intervalo `[0..max)`.

Entrega: `pokemonExamen.pas` con la solución del ejercicio.