Informe trabajo práctico 1
algoritmos y programación II

Integrantes:

Emiliano Pellegrino
Ivan D'angelo
Oscar Héctor Frasson

Decisiones de diseño:

Nos basamos en un diseño simple donde elegimos que los elementos y los tipos de ataque serán enumeraciones,además consideramos necesario que la lógica de un ataque esté contenida en una clase propia que calcule tanto los bonus de daño por un elemento que es débil contra el que atacamos como tanto las atenuaciones de daño con un elemento que es fuerte contra el que atacamos basándonos en los esquemas propuestos por la consigna.
Otra decisión que tomamos es la de manejar el juego mediante un método main con un menú de opciones en impresiones de pantalla que le piden al usuario ingresar un nombre, elegir el par de elementos y una vez iniciado el juego se van alternando ataques entre jugadores hasta que la vida de uno llega a cero.

Descripción de archivos:

- *Ataque:*esta clase recibe como parámetro un tipo de ataque y un elemento. Posee el método "atacar" que recibe como parámetro un monstruo y retorna el daño del ataque, se calcula este mismo dependiendo el elemento y el tipo de ataque y se le resta a la vida del monstruo pasado como parámetro, además se tiene en cuenta que un ataque especial solo se puede usar 4 veces por monstruo.
- *Monstruo:*la clase monstruo posee un atributo vida que representa la vida del monstruo y recibe como parámetro dos elementos.
  Tiene cuatro opciones para atacar usando un vector de la clase Ataque(ataque simple con el primer elemento,ataque especial con el primer elemento,ataque simple con el segundo elemento, ataque especial con el segundo elemento).
  Su método "atacar" recibe una posición del vector y un monstruo al cual atacar, invocando al método atacar de la clase Ataque desde una posición del vector seleccionada por el usuario.

- *Juego:* recibe como parámetro el nombre del jugador uno y dos, también los elementos para el monstruo del jugador uno y dos.
  Posee la interfaz para que se desarrolle el juego mediante un menú de opciones donde el juego transcurre hasta que la vida de algún jugador llega a 0, entonces se muestra el nombre de ese jugador por pantalla.
- *NumeroDeRangoEquivocadoException:* excepcion que sera lanzada si el usuario selecciona algo fuera del rango de los números del 1 al 4 cuando se le pide que elija un ataque.
- *JuegoTests:* posee los test de la clase “Juego”.
- *MonstruoTests:* posee los test de la clase “Monstruo”.
- *Elemento:* posee los tipos de elemento agua tierra aire fuego.

Conclusiones:

Si bien tenemos varias herramientas que vimos durante esta primera parte de la cursada para lograr realizar el tp, optamos por usar las más sencillas ya que con estas pudimos cumplir la funcionalidad que pedía para el juego el enunciado y no quisimos arriesgarnos a usar cosas más complejas que quizás nos dificultan la realización del tp. Además pensamos que para la complejidad del tp sería muy difícil usar herramientas más sofisticadas.
La temática del trabajo práctico nos pareció muy entretenida y el nivel de complejidad del enunciado de acuerdo con el de la cursada.
Por último nos parece que deben haber formas más eficientes de realizar este trabajo práctico que la nuestra, pero que no se nos ocurrieron en el tiempo que tuvimos para hacer el tp por eso no las implementamos.