Complejidad Computacional

Proyecto de la Asignatura

Generador de equipos pokemon

Equipo 3

Integrantes:

1. Manuel Mateo Delgado-Gambino López
2. Diego Melús Caneda
3. Bernardo Quindimil Micó

1 Participación del equipo

El responsable del equipo, que debe estar en la primera fila de la tabla, es el **garante del cumplimiento de lo referido a las prácticas injustas** en el apartado 4 del contrato de enseñanza-aprendizaje de la asignatura.

La información de la participación se refleja en la tabla modelo según lo siguiente:

En la columna participación el responsable del equipo indicara para cada integrante (incluyéndose) alguna de las siguientes opciones:

Opción **SI** (el integrante participa activamente en la elaboración de la asignación)

Opción **NO** (el integrante no ha participado en la elaboración de la asignación)

Opción **%** (el integrante participa en un porcentaje respecto a la Opción SI.

El % implica que ese integrante obtendrá ese porcentaje de la calificación obtenida por el equipo.

Nombre	Participación
Manuel Mateo Delgado-Gambino López	Si
Diego Melús Caneda	Si
Bernardo Quindimil Micó	Si

2 Título

Generador de equipos Pokémon

3 Descripción

Los Pokémon son entidades ficticias que representan una variedad de formas de vida que habitan un mundo conocido como el universo Pokémon creado en 1996 por Nintendo, Game Freak y Creatures Inc. Con el paso del tiempo, se han ido creando nuevos Pokémons que se organizan en generaciones, y a día de hoy hay 9 generaciones; como son una cantidad muy elevada de pokémon, hemos decidido trabajar solo la primera generación que consta de un total de 151 pokémons. Estos Pokémon pueden ser capturados por entrenadores pokémon que los podrán utilizar en combates.

Decidimos invertir tiempo en investigar acerca de un generador de equipos Pokémon debido a que esta franquicia de videojuegos es extremadamente exitosa y ha generado una gran demanda entre jugadores de todas las edades. Un generador de equipos podría satisfacer la necesidad de los jugadores de encontrar combinaciones efectivas de Pokémon para maximizar su rendimiento tanto en un los juegos como a nivel competitivo.

Un combate Pokémon suele ser un 1 contra 1 en el que cada entrenador puede tener como máximo 6 Pokémons. El combate se lleva a cabo por turnos en los que cada jugador elige una acción a realizar, como atacar, usar una habilidad, cambiar de Pokémon o usar un objeto. Estos combates se ven fuertemente influenciados por los tipos de los pokémon. En total hay 15 tipos de pokémon que son: normal, lucha, volador, veneno, tierra, roca, bicho, fantasma, acero, fuego, agua, planta, eléctrico, psíquico y hielo. Cada tipo tiene ventajas y desventajas sobre otros tipos: por ejemplo, los movimientos de tipo agua son eficaces contra los Pokémon de tipo fuego, y a su vez, los movimientos de tipo fuego son débiles contra Pokémons de tipo agua. Cada Pokémon tiene una cantidad de puntos de salud (PS). Cuando un Pokémon recibe daño de un ataque, sus PS disminuyen, y si sus PS llegan a cero, se considera fuera de combate. El combate termina cuándo todos los Pokémons de un jugador están fuera de combate.

Los Pokémon tienen niveles que van aumentando a medida que estos combaten y al llegar a un determinado nivel algunos pueden llegar a evolucionar y hacerse aún más fuertes.

4 Objetivos

Inicialmente, habíamos optado por desarrollar una herramienta de asistencia para equipos Pokémon. Esta herramienta permitiría a los usuarios ingresar un equipo existente y recibir sugerencias sobre qué Pokémon podrían sustituirse por otros para mejorar la competitividad del equipo. Más tarde, decidimos realizar un ajuste y en

vez de que el usuario ingresará su equipo, pensamos que era una mejor idea que el usuario insertase el equipo del rival y que el generador de equipos le devolviese el mejor equipo posible para combatir contra ese rival determinado.

5 Metodología

En un principio habíamos pensado realizar un ayudante de equipos pokemon en el que primero establece una ponderación a cada uno de los pokémon dependiendo de las estadísticas (ataque, ataque especial, defensa, defensa especial, cantidad de PS, velocidad), la cantidad de daño que hagan los movimientos, la cantidad de fortalezas y debilidades. Luego recorrería el vector del equipo que le insertaría el usuario. En el caso de que hubiese dos tipos repetidos, este le quitaría el peor de esos dos pokémon, ya que para que un equipo sea competitivo contra la mayoría de equipos, este tiene que ser variado. Después, el ayudante recorrería el árbol mediante la técnica de backtracking, escogiendo la mejor opción posible para completar el equipo. Este árbol estaría formado por un primer nodo inicial y sus hijos serían los 15 tipos de Pokémon. Recorrería el primer tipo, si ya hay un pokémon de ese tipo en la lista, entonces realiza backtracking y continúa por el segundo tipo, en caso contrario, recorre los hijos de ese nodo, que serían todos los pokémon de dicho tipo. De esta manera, iría recorriendo el árbol hasta encontrar la mejor opción posible que sería el Pokémon con mayor ponderación que tiene un tipo diferente a los otros 5 pokémon del equipo.
Esta idea no nos convenció y decidimos realizar un generador en el que el usuario ingresa el equipo del rival al que se tiene que enfrentar y este le devuelve el mejor equipo posible para hacerle frente teniendo en cuenta una nueva variable denominada puntos de combate que depende de las ponderaciones de cada Pokémon y las debilidades y fortalezas de los tipos del equipo rival.

6 Desarrollo y avances del proyecto

Primero hemos escogido de los 151 pokémons de la primera generación a 62, los que se corresponden con la última etapa evolutiva. A estos 62 pokémon le hemos otorgado una ponderación como ya explicamos previamente. Lo hemos calculado realizando la media de las 6 estadísticas sumado más la media del daño que hacen los 4 movimientos. Si el movimiento no es de daño, es decir, que modifica el estado del rival, por ejemplo, un movimiento que le disminuye el ataque, solo se le otorga un valor de 80 al ataque. A esto se le suma la cantidad de fortalezas menos la cantidad de debilidades y todo ello se multiplica por 10,2 (el 10,2 es debido a que la cantidad de fortalezas y debilidades es un número muy bajo pero que tiene mucha repercusión al momento de dar valor a un pokémon) y luego le restamos 100:

```
competitiveValue = round + pokemonStats + moveStats + typeAdvantage*10,2
-100
```

Después almacenamos en un Json los 62 Pokémon junto a su ponderación y su tipo. Cuando el usuario le introduce al algoritmo el equipo del rival al que se tiene que

enfrentar, este le otorga la ponderación, los tipos de sus Pokémon y las fortalezas y debilidades de esos tipos.
Por ejemplo, le insertamos Charizard, Venusaur, Raticate, Machamp, Dugtrio, Alakazam:

```
Los Pokemon del equipo rival son:
Charizard
Venasaur
Raticate
Machamp
Dugtrio
Alakazam
```

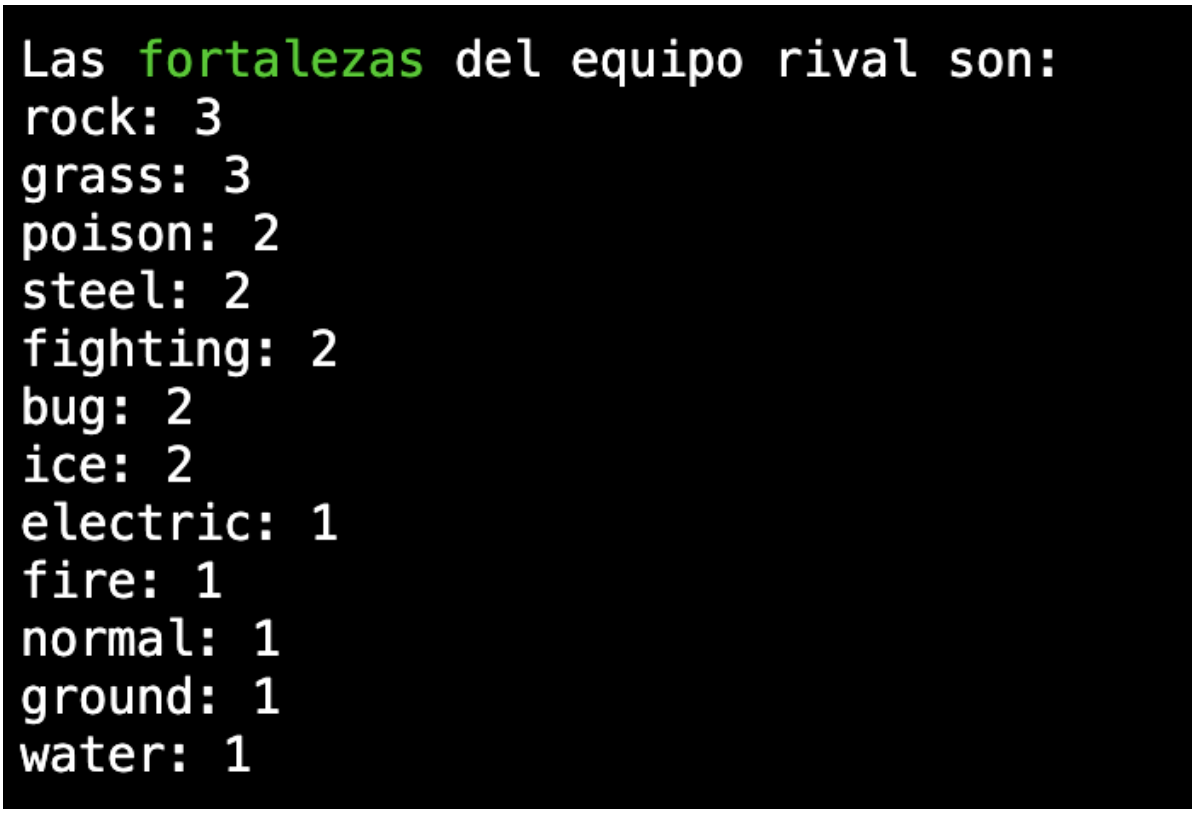
Este nos devuelve la ponderación total (que es simplemente sumar las ponderaciones):

```
El equipo rival tiene un peso total de 481.17
```

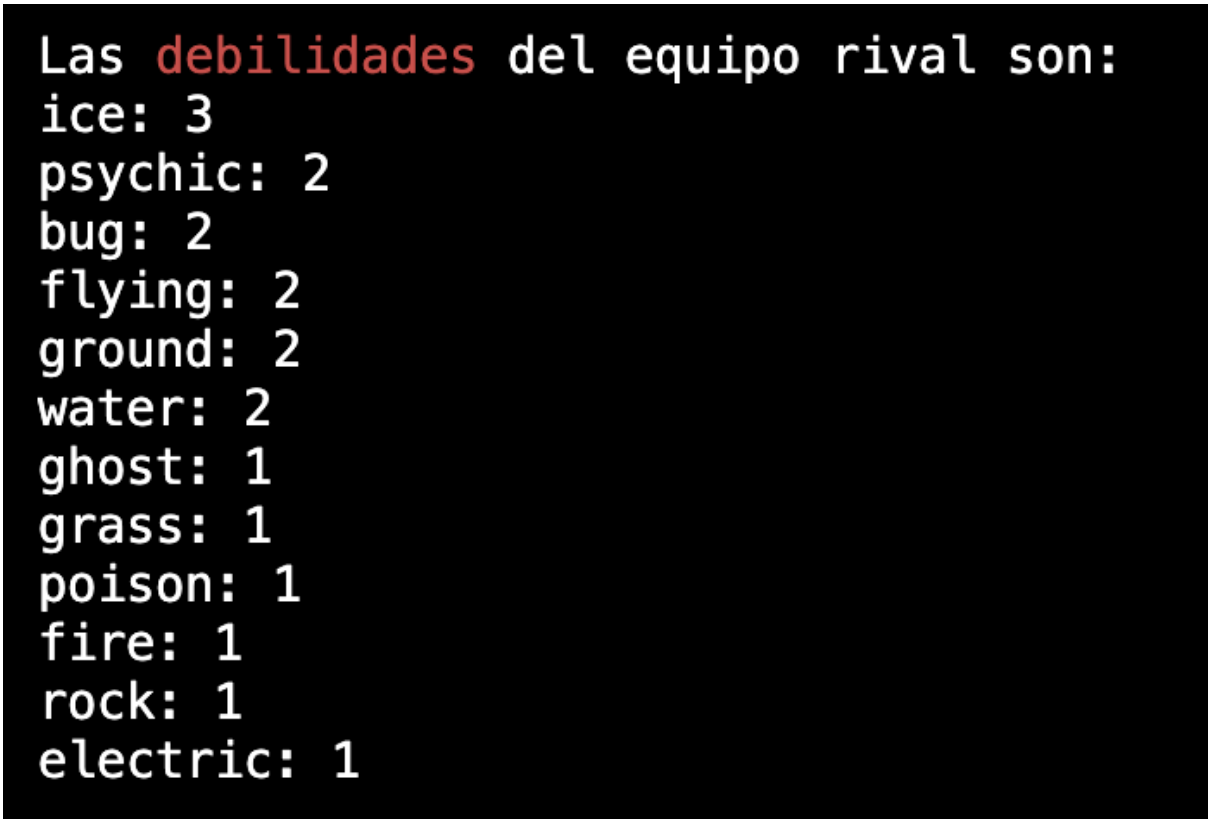
Los tipos de ese equipo:

```
Los tipos de Pokemon que tiene el equipo rival son:
Psychic: 1
Ground: 1
Fighting: 1
Normal: 1
Poison: 1
Grass: 1
Flying: 1
Fire: 1
```

Las fortalezas teniendo en cuenta la tabla de tipos:



Y las debilidades:



Luego, el algoritmo calcula los puntos de combate de todos los pokemon para el equipo del rival en concreto realizando la ponderación multiplicada por 1 más la cantidad de debilidades del equipo rival contra un Pokémon en concreto dividido

entre 1 más la cantidad de fortalezas, (le sumamos 1 debido para que no de 0 abajo y cuando el rival no tenga fortalezas contra un pokémon, los puntos de combate no sean infinito).
Más tarde, en el Json nuevo con los pokémon con sus nuevos puntos de combate contra ese equipo en específico, este se reordena mediante quicksort.

ALGORITMO(*)	COMPLEJIDAD CASO MEDIO	COMPLEJIDAD PEOR CASO
Inserción	N^2	N^2
Burbuja	N^2	N^2
Selección	N^2	N^2
Heapsort	$N \cdot \log N$	$N \cdot \log N$
Mergesort	$N \cdot \log N$	$N \cdot \log N$
Quicksort	$N \cdot \log N$	N^2

Por último, hemos ordenado la lista de pokémon de mayor a menor puntos de combate con tres métodos de ordenación para evaluar cual es el que lo realiza en menos tiempo que son timsort built-in Python, Quicksort y Bublesort.

```
Timsort Built-in Python
El tiempo de ejecucion ha sido: 2.002716064453125e-05

Quicksort:
El tiempo de ejecucion ha sido: 9.942054748535156e-05

Bubblesort:
El tiempo de ejecucion ha sido: 0.00023674964904785156
```

Tras haber realizado varias pruebas hemos determinado que el peor de los 3 es bubblesort. El que nos pareció más cómodo y fácil de implementar fué el timsort que es el algoritmo determinado que usa python y simplemente hay que usar el comando sorted.

Timsort es un algoritmo de clasificación que es eficiente para datos del mundo real y no creado en un laboratorio académico. Tim Peters creó Timsort para el lenguaje de programación Python en 2001. Timsort primero analiza la lista que intenta ordenar y luego elige un enfoque basado en el análisis de la lista.
Desde que se inventó el algoritmo, se ha utilizado como el algoritmo de clasificación predeterminado en Python, Java , la plataforma Android y en GNU Octave.

El tiempo de clasificación de Timsort es el mismo que el de Mergesort, que es más rápido que la mayoría de los otros tipos que pueda conocer. Timsort en realidad utiliza la ordenación por inserción y la ordenación por combinación, como verá pronto.

Peters diseñó Timsort para usar elementos ya ordenados que existen en la mayoría de los conjuntos de datos del mundo real. A estos elementos ya ordenados los llama “ejecuciones naturales”. Itera sobre los datos recopilando los elementos en ejecuciones y fusionando simultáneamente esas ejecuciones en una sola.

Después de tener ordenada la lista mediante un método que escoja el usuario, el algoritmo simplemente selecciona los 6 primeros Pokémons, que sería el mejor equipo posible para luchar contra el equipo del rival. Hemos decidido utilizar un algoritmo de ordenación de búsqueda porque si el usuario no tiene alguno de esos 6 pokemon o no le gustan, puede tener una lista ordenada de cuáles son los mejores para combatir.

Finalmente, el algoritmo devuelve los 6 pokémon junto con sus datos, su imagen en pixelart mediante la librería de github pokemon-colorscript y se reestructuran los datos para que se vean mejor en la terminal.

7 Problemas y soluciones

Durante el desarrollo del proyecto al realizarlo mediante backtracking, como teníamos pensado hacer en un principio, aparecieron muchos problemas y no nos convenció. Por eso, decidimos cambiar la perspectiva y aplicar algoritmos de ordenación.

Bibliografía

Brandon. (2018, 26 junio). *Timsort: el algoritmo de clasificación más rápido del que nunca has oído hablar*. HackerNoon. https://hackernoon.com/es/timsort-el-algoritmo-de-clasificacion-mas-rapido-del-que-nunca-has-escuchado-36b28417f399

Desarrollo y avances del proyecto

- **Título definitivo**. Confirmación o ajuste del título propuesto en función del desarrollo y enfoque del proyecto.
- **Resumen.** Breve resumen de los trabajos realizados hasta el momento y los resultados preliminares obtenidos.
- **Objetivos.** Revisión de los objetivos planteados inicialmente, indicando cualquier ajuste o precisión necesarios.

- **Metodología.** Detalle de la metodología y técnicas aplicadas durante el período de avance. Cambios o ajustes a la metodología preliminar, si los hubiera.
- **Desarrollo y avances del proyecto**. Descripción detallada del trabajo realizado hasta la fecha.
- **Problemas y soluciones.** Problemas encontrados durante el desarrollo del proyecto y soluciones aplicadas o propuestas.
- **Próximos pasos.** Descripción de las actividades y objetivos hasta la entrega final.
- **Bibliografía preliminar.** Fuentes bibliográficas consultadas hasta la fecha en formato APA7.