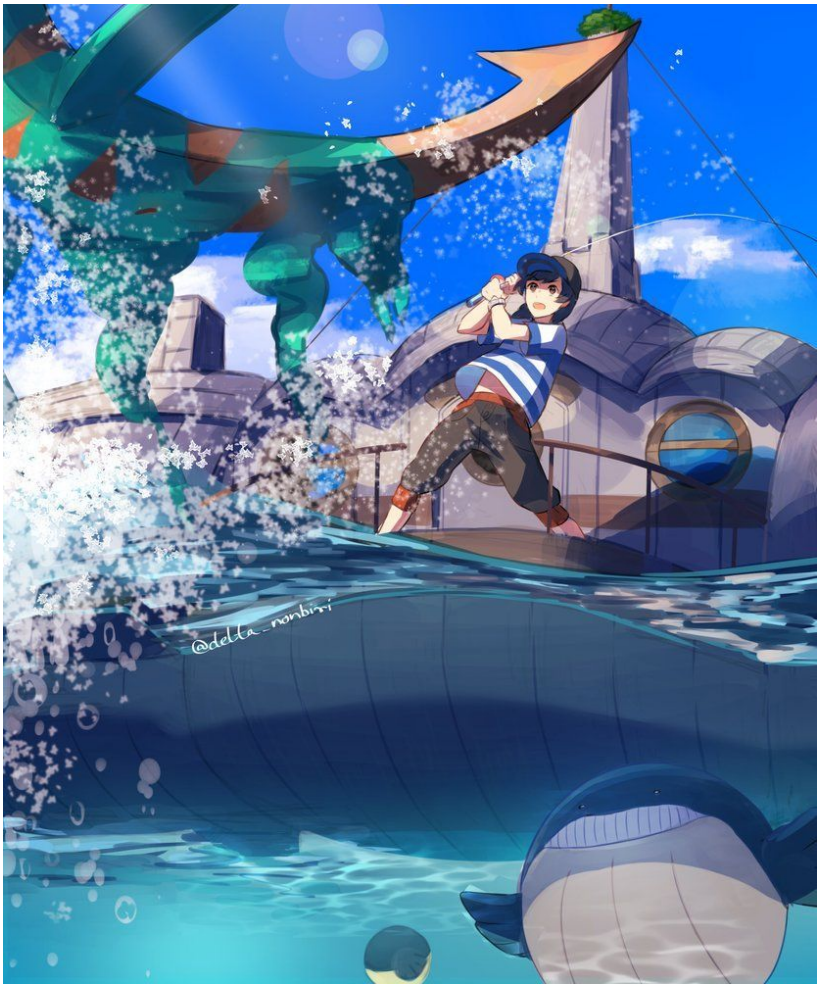
Algoritmos y Programación II
Curso Mendez

.UBAfiuba
FACULTAD DE INGENIERÍA

¡Gran pesca Pokémon!

TP1



Fecha de Presentación: 09/10/2020
Fecha de Vencimiento: 16/10/2020

1. Introducción

Los pokémon son unas criaturas que se encuentran en la naturaleza. Se clasifican en 18 tipos (planta, fuego, agua, volador, tierra, etc). Cada pokémon tiene una cantidad determinada de movimientos o ataques que puede aprender, los cuales se pueden clasificar por su tipo o elemento (por ejemplo, el ataque rayo burbuja es un ataque tipo agua). Además, los pokémon cuentan con diferentes atributos que los distinguen de otros pokémon de la misma especie: Velocidad, Color, Peso, entre otros.

Los **entrenadores pokémon**, tienen como meta cumplir dos objetivos: capturar a todas las especies de pokémon disponibles y completar así la información de todos los pokémon en la Pokédex (Enciclopedia Pokémon). Por otro lado, deben **entrenarlos y enfrentarlos** a pokémones pertenecientes a otros entrenadores para demostrar sus habilidades, fortalezas, talentos y así convertirse en un **Maestro Pokémon**.

Para cumplir la primer meta, los entrenadores viajan a lo largo y ancho de las regiones del mundo, capturando los diferentes pokémon que encuentran en su camino. Hoy los entrenadores se encontrarán con la oportunidad de capturar pokémon de tipo agua de una manera distinta a la usual...

2. Objetivo

El presente trabajo tiene como finalidad que el alumno repase algunos de los conocimientos adquiridos en Algoritmos y Programación I, así como también que comience a familiarizarse con las herramientas a utilizar a lo largo de la materia, tales como el manejo de memoria dinámica y la utilización de punteros a funciones.

3. Enunciado

La líder de gimnasio Misty organizó un evento de pesca que se llevará a cabo en su acuario, ubicado en Ciudad Celeste. Para ello necesitará muchos y muy variados pokémon de tipo agua. El lugar indicado en Kanto para obtener a todos estos pokémon es el arrecife que rodea a las Islas Espuma.

Para realizarlo, te pide que la ayudes a trasladar los pokémon seleccionados desde el arrecife a su acuario, donde será realizado el evento.

A Misty le preocupa perjudicar la población de las especies más vulnerables, es por ello que te pide realizar una simulación del arrecife, pudiendo ver como quedará su estado dependiendo de qué especies se elijan para el evento.

Las funcionalidades que se deben implementar se detallan a continuación:

```c
/*
 * Función que dado un archivo carga los pokémon que viven en el arrecife
 * reservando la memoria necesaria para el mismo. Se debe intentar leer la mayor
 * cantidad posible de registros del archivo. Al encontrar el primer registro
 * erróneo (o al llegar al final del archivo) se deben finalizar la lectura y
 * crear el arrecife con los pokémon leídos exitosamente. En caso de no
 * encontrar ningún registro con el formato correcto, se debe devolver error.
 * Devuelve un puntero a un arrecife válido o NULL en caso de error.
 * Es importante notar que tanto
 */
arrecife_t* crear_arrecife(char ruta_archivo[]);

/*
 * Función que crea un acuario vacío reservando la memoria necesaria para el mismo.
 * Devuelve el acuario o NULL en caso de error.
 */
acuario_t* crear_acuario();

/*
 * Función que deberá sacar del arrecife a todos los pokémon que satisfagan la
 * condición dada por el puntero a función (que devuelvan true) y trasladarlos
 * hacia el acuario. El parámetro cant_seleccion especifica la cantidad máxima
 * de pokémon que serán trasladados. En caso de que haya menos pokémon trasladables en el
 * arrecife que los pedidos, no se deberá mover ninguno para conservar los pocos existentes.
 * El vector de pokemones del arrecife quedará solo con aquellos que no fueron
 * trasladados (su tamaño se ajustará luego de cada traslado).
 * El vector de pokemones del acuarió quedará con aquellos que fueron
 * trasladados esta vez más los que ya había en el
 * acuario (su tamaño se ajustará luego de cada traslado).
```

```c
 * Devuelve -1 en caso de error o 0 en caso contrario.
 */
int trasladar_pokemon(arrecife_t* arrecife, acuario_t* acuario, bool (*seleccionar_pokemon)
    (pokemon_t*), int cant_seleccion);


/*
 * Procedimiento que dado un arrecife deberá mostrar por pantalla a todos los pokemon que contiene.
 */
void censar_arrecife(arrecife_t* arrecife, void (*mostrar_pokemon)(pokemon_t*));


/*
 * Función que dado un acuario guarda en un archivo de texto a los pokemones que contiene.
 * Devuelve 0 si se realizo con éxito o -1 si hubo algun problema para guardar el archivo.
 */
int guardar_datos_acuario(acuario_t* acuario, const char* nombre_archivo);


/*
 * Libera la memoria que fue reservada para el acuario.
 */
void liberar_acuario(acuario_t* acuario);


/*
 * Libera la memoria que fue reservada para el arrecife.
 */
void liberar_arrecife(arrecife_t* arrecife);
```

A su vez, se cuenta con los siguientes registros a utilizar en el presente trabajo:

```c
typedef struct pokemon{
        char especie[MAX_ESPECIE];
        int velocidad;
        int peso;
        char color[MAX_COLOR];
} pokemon_t;

typedef struct acuario{
        pokemon_t* pokemon;
        int cantidad_pokemon;
} acuario_t;

typedef struct arrecife{
        pokemon_t* pokemon;
        int cantidad_pokemon;
} arrecife_t;
```

Los pokémon pertenecientes al arrecife se encontrarán en el archivo **arrecife.txt** cuya ruta llega como parámetro.

El archivo tendrá la siguiente información:

```
especie1;velocidad1;peso1;color1
especie2;velocidad2;peso2;color2
especie3;velocidad3;peso3;color3
...
```

Un ejemplo se muestra a continuación

```
Magikarp;9;10;rojo
Goldeen;23;15;blanquirrojo
Feebas;11;8;marron
Luvdisc;27;9;rosa
...
```

Los pokémon que estén registrados en el acuario de Misty serán guardados en un archivo **acuario.txt** con el siguiente formato:

```
especie1;velocidad1;peso1;color1
especie2;velocidad2;peso2;color2
especie3;velocidad3;peso3;color3
...
```

4. Resultado esperado

Se espera que el trabajo creado sea compilado sin errores con la siguiente línea:

```
gcc *.c -Wall -Werror -Wconversion -std=c99 -o evento_pesca
```

Luego, que sea ejecutado y permita realizar el translado al acuario de los pokémon que fueron seleccionados, utilizando las funciones de selección de pokémon que se especifiquen.

Para verificar el correcto funcionamiento, se recomienda utilizar un programa principal con el siguiente flujo:

```
/*
* Funciones que determinan los Pokémon que serán movidos al acuario:
* - Como mínimo es necesario crear 5.
* - Ejemplos:
* - - Sólo pokémon pertenecientes a la especie "Magikarp".
* - - Sólo pokémon que tengan color rojo y velocidad menor a 10.
* - - Etc. sean creativos!
*/

int main(){
        /* Crear las estructuras a utilizar */
        /* Trasladar por 1ra vez con cierta función */
        /* Listar a los pokémon que continúan en el arrecife */
        /* Trasladar por 2da vez con cierta función */
        /* Listar a los pokémon que continúan en el arrecife */
        ...
        /* Trasladar por N-écima vez con cierta función */
        /* Listar a los pokémon que continúan en el arrecife */
        /* Guardar los pokémon del acuario */
        /* Liberar memoria */
    return 0;
}
```

Y, ser corrido con **valgrind**:

```
valgrind --leak-check=full --track-origins=yes --show-reachable=yes ./evento_pesca
```

Por ejemplo, si partimos del siguiente archivo arrecife.txt, el cual solo contenia 4 pokemon...

```
Magikarp;9;10;rojo
Goldeen;23;15;blanquirrojo
Feebas;11;8;marron
Luvdisc;27;9;rosa
```

Y elegimos la función en la que se trasladan sólo pokémon de la especie Magikarp, debería obtenerse el siguiente resultado:

```
==5602== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==5602== Command: ./evento_pesca
==5602==

Bla bla bla no perdiste memoria bla bla bla

==5602==
==5602== HEAP SUMMARY:
==5602==     in use at exit: 0 bytes in 0 blocks
==5602==   total heap usage: ¿9 allocs?, ¿9 frees?, ¿6,592 bytes allocated? -> ver
==5602==
==5602== All heap blocks were freed -- no leaks are possible
==5602==
==5602== For counts of detected and suppressed errors, rerun with: -v
==5602== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

5. Entrega

La entrega deberá contar con todos los archivos necesarios para compilar y ejecutar correctamente el TP. Se recomienda que la misma conste de un archivo .c en donde transcurra el flujo principal del programa, y **otro** que contenga las implementaciones de la biblioteca brindada por la cátedra.

Dichos archivos deberán formar parte de un único archivo **.zip** el cual será entregado a través de la plataforma de corrección automática **Chanutron2021** (patente pendiente).

El archivo comprimido deberá contar además con:

- Un **README.txt** que contenga:
  - Una breve introducción sobre el funcionamiento del trabajo presentado.
  - Una explicación de como compilarlo (línea de compilación) y como ejecutarlo (línea de ejecución).
  - Explique brevemente y de forma concisa los siguientes conceptos:
    1. Punteros.
    2. Aritmética de punteros.
    3. Punteros a funciones.
    4. Malloc y Realloc.
- El enunciado.

6. Referencias

- https://pokemon.fandom.com/es/wiki/Estadisticas
- http://www.juegosdb.com/guia-del-entrenador-pokemon-caracteristicas-de-un-pokemon-nintendo-ds-nintendo-3ds/
- https://es.wikipedia.org/wiki/Pokémon