75.41 95.15 95.12 Algoritmos y Programación II Curso 4
TP1 - Sala de escape Pokemon

22 de marzo de 2022

1. Introducción

Bienvenido al maravilloso mundo de los Pokémon, donde. . . ¿Eh? ¿Dónde estás? Te despertas en una habitación desconocida y no recordas lo que estabas haciendo. Miras a tu alrededor y notas una única puerta cerrada, la única posibilidad para salir afuera y escapar de este lugar. ¿Podrás lograrlo?

2. Logrando escapar

Para lograr escapar vas a poder utilizar todos los objetos que encuentres en la habitación, desde lapiceras, pokebolas, anillos, y hasta cucharas. Cada elemento puede ser fundamental para lograr el escape. Por ello te facilitamos un archivo de texto que contiene la información de todos los objetos existentes dentro de la sala. Tu primer tarea va a ser abrir este archivo y representar en memoria la información contenida en el mismo.

Pero un momento, aún hay más. Cómo sabemos que es posible hacer que diferentes objetos interactúen entre sí (por ejemplo usar una llave para abrir un cajón) o incluso que se pueda realizar una acción particular sobre un objeto (como inflar un globo o examinar una nota), también te facilitamos un archivo que contiene toda la información de las interacciones posibles de cada objeto.

3. Formato de los archivos

Ambos archivos a utilizar son archivos de texto donde cada línea contiene varios campos separados por el caracter ';'. El archivo de objetos tiene 3 campos por línea mientras que el de interacciones tiene 4. En el caso del archivo de objetos, la estructura de cada línea es la siguiente:

```
NOMBRE_OBJETO;DESCRIPCION;FLAG
```

Dónde nombre y descripción son strings arbitrarios mientras que el flag es uns tring que solamente puede ser true o false.

En el caso del archivo de interacciones los campos son:

```
NOMBRE_OBJETO;VERBO;NOMBRE_OBJETO2;ACCION
```

Donde los nombres de los objetos y el verbo son strings arbitrarios. En el caso del segundo nombre de objeto, existe un caso particular donde el string es literalmente '_'. Este caso significa que la interacción no necesita un segundo objeto para poder realizarse. El campo acción también es un string, esta vez separado por ':', con 3 campos como se muestra a continuación:

```
TIPO_ACCION:NOMBRE_OBJETO:MENSAJE
```

Donde el tipo es un caracter que indica el tipo de acción, y el nombre del objeto y el mensaje son strings arbitrarios. En este caso el objeto también puede ser '_' si no es necesario. Los tipos de acción posibles son: **d** para descubrir objetos, **r** para reemplazar objetos, **e** para eliminar objetos y **m** para mostrar un mensaje.

3.1. Funcionalidad pedida

En esta primer instancia se pide implementar la lectura de archivos para la correcta creación de la sala de escape. Para ello se proveen los archivos **.c** y **.h** básicos junto con un par de archivos ejemplo de **objetos** e **interacciones**.

Los archivos **.h provistos no deben ser modificados**. Se pueden agregar mas archivos **.c** o **.h** dentro del directorio **src**. El archivo **escape_pokemon.c** puede ser modificado de cualquier forma siempre que cumpla con los siguientes requerimientos:

- La función main debe recibir por parámetro 2 nombres de archivo. El primer archivo corresponde al archivo de objetos y el segundo al de interacciones.
- Se debe crear una sala de escape con los archivos provistos por parámetro.
- Se deben listar por pantalla todos los objetos leídos del archivo.
- Se debe mostrar por pantalla la validez de por lo menos 4 interacciones diferentes entre objetos.
- Se debe liberar correctamente toda la memoria.
- No se puede incluir el archivo **src/estructuras.h**. Este archivo es considerado privado de la implementación y no debe ser utilizado.

3.2. Salida esperada

Si la implementación de tu TP es correcta, deberías ver por pantalla algo parecido a lo siguiente:

```
>./sala ejemplo/objetos.txt ejemplo/interacciones.txt
Objetos...
0: habitacion
1: mesa
2: interruptor
3: pokebola
4: cajon
5: cajon-abierto
6: llave
7: anillo
8: puerta

Interacciones...
Examinar la habitacion: Válido
Abrir pokebola: Válido
Usar llave en el cajon: Válido
Quemar la mesa: Inválido
```

3.3. Entrega

Una vez completada la implementación, Hay que subirla al sistema de entregas, para verificar su correcto funcionamiento. Además de la implementación, tenes que entregar un informe (ya sea en formato PDF o video de 5 minutos o menos). En el informe **tenés que explicarle a tu corrector** cómo se compila y corre tu programa, cómo funciona la lectura de archivos, cómo manejas la memoria dinámica y debes incluir diagramas de memoria o de algún otro tipo que ayuden a entender tu implementación. **El informe es parte de la nota**.

Algunas recomendaciones:

- Se puede asumir que la longitud máxima de las líneas de los archivos de texto a leer es de 1024 caracteres.
- No todo en la vida es memoria dinámica. Úsenla donde sea apropiado. El stack sirve para colocar variables locales, úsenlo.
- La modularización es importante (¿Ya lo había dicho?). No me hagan un choclo.
- Los nombres de función son importantes. Mejor **'buscar_posición_separador'** que **'buscar'**.
- **NO** está permitido recorrer los archivos de texto mas de una vez.
- Recordá que la cátedra utiliza el estilo de código propuesto por el equipo de desarrollo del **Kernel de Linux** y dicho estilo (indentación, espaciado, convención de nombres, etc) debe ser respetado para que la entrega sea válida.