Universidade Federal do Cariri - CCT
Professor: Luis Henrique Bustamante
Ciência da Computação
luis.bustamante@ufca.edu.br

Disciplina: Algoritmos e Estrutura de Dados I
T02/T04- MC/CC
26 de abril de 2022

Primeiro Trabalho

O objetivo desse trabalho é criar uma Pokedex[1] simplificada. Neste projeto de Pokedex o usuário deve acessar os pokemons por tipo e por meio do seu score total de forma crescente, isto é, do menor para o maior score total, sendo o score total igual à hp + ataque + defesa + ataque_esp + defesa_esp + velocidade. Essas são características dos pokemons. Cada pokemon tem um identificador único e pertence tipo (normal, fogo (Fire), água (Water), grama (Grass), voador (Flying), lutador (Fighting), veneno (Poison), elétrico (Electric), terra (Ground), pedra (Rock), psíquico (Psychic), gelo (Ice), inseto (Bug), fantasma (Ghost), ferro (Steel), dragão (Dragon), sombrio (Dark) e fada (Fairy)). Existe uma lista de todos os [POKEMONS] nas referências no final desse arquivo.

Para isto, você deverá gerenciar uma estrutura que contém um array (ou mesmo uma outra lista encadeada) de listas encadeadas de pokemons. Cada lista conterá todos os pokemons de um determinado tipo. Assim, essas listas são não circulares e com nó-descritor. Na representação em memória de cada pokemon haverá um ponteiro para o próximo pokemon de seu tipo da lista (cujo score total será maior ou igual ao do pokemon atual).

Uma possível estrutura para guardar um pokemon pode ser considerada a seguir:

```
struct pokemon
{
  char* nome;
  int id, score_total;
  int tipo, hp, ataque, defesa, ataque_esp, defesa_esp, velocidade;
  struct Pokemon *prox;
};
```

Você pode modificar ligeiramente a estrutura do pokemon tomando o cuidado para não alterar essas partes do pokemon. Por exemplo, caso queira gerenciar evoluções de pokemons, você poderá criar um ponteiro para a próxima evolução de cada pokemon, caso ela exista. Mas, não é necessário para o que é pedido no trabalho.

Dentre as operações previstas para a Pokedex estão:

- criação e destruição de uma pokedex;
- busca por um pokemon pelo identificador;
- imprime os pokemons de um mesmo tipo inseridos na Pokedex;
- consulta à quantidade de pokemon de um único tipo;
- consulta à quantidade de pokemons diferentes na pokedex (tamanho);
- inserção de um novo pokemon;
- remoção de pokemon;
- alteração dos dados de um pokemon.

Para este trabalho, você deverá implementar um conjunto de funções de gerenciamento de listas utilizando principalmente os conceitos: array de listas e listas ligadas ordenadas não circulares e com nó-descritor. Você é livre para implementar qualquer função que considerar necessária.

[1] https://www.pokemon.com/us/pokedex/

0.1 Assinatura das funções necessárias

```
Pokedex* criar_Pokedex(int Qtd_Tipos)
```

Essa função recebe um inteiro e cria um array[Qtd_Tipos] com a quantidade de tipos de pokemons passados como parâmetro. Considere o tipo de um pokemon os elementos da coluna *Type_1* do arquivo .cvs em [POKEMONS]. Considere o tipo de um Pokemon dado como uma string no arquivo .cvs como um tipo inteiro, i.e, codifique esse o tipo de um pokemon como um inteiro.

```
void libera_Pokedex(Pokedex* pd)
```

Essa função deve liberar o espaço de memória para todos as listas de pokemons e para o endereço reservado à Pokedex.

```
int busca_pokemon(Pokedex* pd, int id, struct pokemon *saida);
```

Essa função deve buscar na Pokedex um pokemon com a id passada como parâmetro e passar o endereço de memória do pokemon para a variável saída.

```
void imprime_por_Tipo(Pokedex* pd, int tipo);
```

Essa função deve imprimir todos os pokemons de uma lista de um único tipo determinado pelo tipo.

```
int qtd_por_Tipo(Pokedex* pd, int tipo);
```

Essa função retorna a quantidade de pokemons de um determinado tipo.

```
int tamanho(Pokedex* pd);
```

Essa função calcula a quantidades de pokemons em uma Pokedex

```
int insere_Pokemon(Pokedex pd, struct pokemon pk)
```

A função que recebe o endereço de uma pokedex e um pokemon, e retorna um valor inteiro.

```
int remove_Pokemon(Pokedex pd, int id)
```

Essa função que recebe um endereço para uma pokedex, o identificador de um pokemon. A função deverá retornar 0 caso não haja na lista um pokemon com o identificador passado como parâmetro. Caso contrário, o pokemon deve ser removido da Pokedex.

```
int atualiza_Pokemon(Pokedex *pd, int id)
```

Esta função recebe como parâmetro o endereço de uma Pokedex e o identificador de um pokemon e deverá retornar 0, caso não exista um pokemon com o identificador passado como parâmetro. Caso contrário, deverá alterar os campos do respectivo pokemon, reorganizar a respectiva lista ordenada, caso seja necessário, e retornar 1. Observe que, caso o valor total do pokemon atual se torne estritamente menor do que o valor total do pokemon anterior ou estritamente maior do que o valor total do próximo pokemon, ele deverá mudar de posição na respectiva lista ligada ordenada (de forma que a lista continue ordenada).

0.2 Informações gerais

Os trabalhos devem ser feitos em dupla que devem ser submetidos por um dos membros da dupla via atividade no Google Classroom até às 23:55h do dia 7 de Maio (com margem de tolerância de 60 minutos).

Envie três arquivos: pokedex.h, pokedex.c e um pokemon.c.

O arquivo pokemon.c deve apresentar alguns testes que você achar necessário para o funcionamento das funções de uma Pokedex. Ele também servirá para avaliar melhor a sua Pokedex.

Suas funções serão testadas individualmente e em conjunto. Todos os trabalhos passarão por um processo de verificação de plágios. Em caso de plágio, todos os alunos envolvidos receberão nota zero.

①

[EDA] J.L. Szwarcfiter and L. Markezon, "Estruturas de Dados e seus Algoritmos," *LTC*, 2010, 3 ª ed.

[EDD] A. Backes, "Estruturas de Dados Descomplicada - Em Linguagem C," *Elsevier*, 2016, 1ª ed.

[POKEMONS] G. Armand, " ", Pokemons.cvs.