Documentação do Trabalho Prático 1 - Programa e Desenvolvimento de Software 1

Nome: Erick Pedrosa Barreto

Matrícula: 2024102608

Introdução

Este trabalho prático tem como objetivo a implementação de uma simulação de batalha de Pokémon entre dois jogadores, utilizando exclusivamente a linguagem C. O código foi desenvolvido para atender aos requisitos especificados, aplicando os conceitos aprendidos durante o curso, como tipos de dados, ponteiros, variáveis compostas, estruturas de controle e repetição, funções e modularização.

Descrição do Problema

A tarefa envolve a simulação de batalhas Pokémon, onde:

- Cada jogador possui de 1 a N Pokémon (com N ≤ 100).
- Cada Pokémon tem as seguintes características:
    - Nome
    - Ataque
    - Defesa
    - Vida
    - Tipo (como Água, Fogo, Pedra, etc.).
- O combate ocorre em turnos alternados, onde o jogador atacante tenta reduzir a vida do Pokémon adversário até zero.
- O tipo do Pokémon influencia a eficácia dos ataques (super efetivo, não muito efetivo ou normal).
- A entrada do programa é um arquivo `.txt`, e a saída detalha o vencedor, os Pokémon sobreviventes e derrotados de cada jogador.

Implementação

Estrutura de Dados

Foi utilizada uma estrutura chamada `pokemon`, que contém os atributos:

```
typedef struct pokemonCompleto {
    char nome[TAM_MAX_NOME];
    int ataque;
    int defesa;
```

```
    int vida;
    char tipo[TAM_MAX_TIPO];
} pokemon;
```

Essa estrutura armazena todas as informações necessárias para cada Pokémon, permitindo fácil manipulação e acesso durante a execução da batalha.

Lógica Principal

A lógica do programa está estruturada da seguinte forma:

1. **Leitura de Arquivo**: Os dados de entrada (número de Pokémon e atributos de cada um) são lidos de um arquivo `.txt`.
    - Foi utilizado o comando `fscanf` para leitura.
    - As informações lidas são armazenadas em arrays dinâmicos para cada equipe de Pokémon.
2. **Sistema de Batalha**:
    - A batalha ocorre em turnos alternados.
    - O ataque é calculado considerando a relação entre os tipos dos Pokémon atacante e defensor.
    - Se o ataque for maior que a defesa do adversário, a vida do Pokémon defensor é reduzida pela diferença; caso contrário, a vida é reduzida em 1 unidade.
    - Um Pokémon é considerado derrotado se sua vida for menor ou igual a zero.
    - Quando todos os Pokémon de um jogador são derrotados, o jogo termina, e o vencedor é anunciado.
3. **Cálculo de Dano**:
    - A função `verificaDano` determina o multiplicador de ataque baseado nos tipos dos Pokémon (super efetivo, não efetivo ou normal).
    - A função `verificaTipo` converte o tipo em um valor numérico para facilitar comparações.

Exemplo de Saída

Dado o arquivo de entrada:

```
3 2
Squirtle 10 15 15 agua
Vulpix 15 15 15 fogo
Onix 5 20 20 pedra
Golem 20 5 10 pedra
Charmander 20 15 12 fogo
```

A saída gerada será:

Squirtle venceu Golem
Charmander venceu Squirtle
Vulpix venceu Charmander
Jogador 1 venceu
Pokemon sobreviventes:
Vulpix
Onix
Pokemon derrotados:
Squirtle Golem Charmander

Decisões de Implementação

- **Relações de Tipos**: Seguindo a tabela do enunciado, foi utilizado um sistema simplificado de comparação para calcular a eficácia do ataque.
- **Modularização**: Funções auxiliares como `verificaDano` e `verificaTipo` foram criadas para melhorar a legibilidade e a organização do código.
- **Alocação Dinâmica**: Para permitir equipes de tamanho variável, as estruturas de Pokémon foram alocadas dinamicamente utilizando `calloc`.

Conclusão

O programa foi desenvolvido para atender integralmente às especificações do trabalho prático. Foram aplicadas boas práticas de programação em C, como modularização e manipulação de ponteiros, e a lógica de batalha foi devidamente testada com diferentes cenários para garantir sua funcionalidade.