UFES - CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA
Prof. Thomas W. Rauber
Estruturas de Dados I – Engenharia de Computação & Ciência da Computação

1º Trabalho 2016/2 – Pokélight

Última atualização: 23 de agosto de 2016, 16:02
Data de entrega: veja `www.inf.ufes.br/~thomas`

Linguagem de Programação para Implementação: C
Grupo de até dois alunos

1 Objetivo

Representação e manipulação de informação estruturada por linguagem de programação de alto nível.

O objetivo do trabalho é a implementação uma versão simplificada de um jogo inspirado no *hype* do momento, o Pokémon Go. Vários jogadores se movimentam em um campo de jogo retangular onde se escondem vários Pokémons que o jogador tenta capturar.

Cada jogador administra seus Pokémons em uma lista encadeada, um elemento da lista para cada Pokémon diferente, com um contador que armazena a quantidade dos Pokémon desta espécie. Se a quantidade de uma espécie de Pokémon ultrapassar uma quantidade predefinida, o Pokémon evoluirá em uma outra espécie. Isto significa que tem que ser eliminado o atual e inserindo o novo Pokémon. A especificação qual evoluirá para qual é do critério do aluno. Pode se inspirar nas regras do jogo normal, veja as referências.

2 Estrutura do Sistema

2.1 Mapa-múndi

Os jogadores podem se movimentar em um campo de jogo retangular. Este campo deve ser organizado em uma matriz de posições, cada posição identificada pela sua coordenada (x, y). De uma posição para outra, o jogador pode chegar através de uma sequência de quatro direções de movimentos possíveis, `esq, dir, cima, baixo`. O jogador não pode ultrapassar os limites do campo. O movimento pode ser arbitrário, ou direcionado, especificando o destino.

A estrutura de dados para implementar deve ser uma matriz de ponteiros, em que cada ponteiro direciona para uma estrutura que administra a ocorrência de todos os Pokémons que se encontram nesta posição.

```
// Proposta:

#define XMAX 20
#define YMAX 10
typedef unsigned int uint;

typedef struct posicao
uint x, y;
...  // todas as informações necessárias
Posicao;

Posicao *campo[XMAX][YMAX];
```

2.2 Dinâmica

Ao passar pelo campo, o jogador, quando encontrar um Pokémon pode adicioná-lo à sua lista, eliminando-o da posição. Tem que considerar eventuais evoluções em caso de aumento de quantidade de uma certa espécie. O campo tem que ser inicializado, i.e. em certas posições um ou mais Pokémons têm que ser colocados na estrutura de dados que administra a informações da posição (x, y).

Como vários jogadores podem jogar simultaneamente, tem que existir um mecanismo de sincronização que divide o tempo em instantes discretos. Cada passagem de um instante de tempo para outro implica em executar um comando de cada jogador. Um movimento de uma posição para uma posição vizinha deve custar um instante de tempo. Por exemplo, se o jogador se movimenta da posição $(3, 2)$ para $(5, 1)$, três instantes de tempo vão passar. Em cada instante, podem surgir eventos, por exemplo o jogador achou um Pokémon na posição $(4, 2)$ e outro em $(5, 1)$.

2.3 Funções

O sistema deve providenciar funcionalidades administrativas e operacionais. Por exemplo como administração, deve ser possível incluir um novo jogador e colocar e eliminar um Pokémon na estrutura da posição na inicialização do jogo. Uma função operacional poderia ser um deslocamento de um jogador da posição atual até uma outra posição, ou uma evolução de um Pokémon. A funcionalidade das unidades envolvidas é ilustrada na tabela 1, separado por operações administrativas e funções operacionais do jogador e dos Pokémons.

Unidade	**Função**	**Parâmetros**
	ADMINISTRATIVAS	
Administrador	Incluir jogador	id jogador, nome jogador, id grupo
Administrador	Eliminar jogador	id jogador
Administrador	Mudar grupo jogador	id jogador, id grupo existente, id grupo novo
Administrador	Inserir Pokémon	id Pokémon, posição (x,y)
	OPERACIONAIS	
Jogador	Movimentar um passo	id jogador, direção
Jogador	Movimentar para posição	id jogador, posição (x,y)
Jogador	Mostrar Pokémons	id jogador
Jogador	Inserir Pokémon	id jogador, id Pokémon
Pokémon	Incrementar quantidade	id jogador, id Pokémon
Pokémon	Evoluir	id jogador, id Pokémon existente, id Pokémon evoluído

Tabela 1: Funcionalidades das unidades.

3 Representação de Informação

Os Pokémons existentes com todas os seus atributos necessários, podem ser armazenados em um vetor de estrutura.

O campo de jogo deve ser uma matriz de ponteiros para uma estrutura que é capaz de armazenar toda a informação desta posição. Uma lista encadeada de Pokémons parece ser a mais apropriada.

Os jogadores participantes devem ser uma lista encadeada, gerenciada pelo administrador.

O jogador deve ser uma estrutura com todas as informações necessárias (id, nome, etc.) e uma lista encadeada dos seus Pokémons.

4 Interface

Toda a interação deve funcionar através de arquivos de entrada e saída. O usuário (aluno ou professor) não deve digitar absolutamente nada além do comando de execução do jogo com um nome de um arquivo de entrada como parâmetro. A sintaxe dos comandos do arquivo de entrada deve ser conforme às funções definidas na tabela 1.

Cada comando tem uma composição fixa da forma

```
<Unidade> <Ação> <Parâmetros>
```

```
Exemplos:

admin incluirjogador 3 "José Manuel"1
admin incluirjogador 5 "Maria de Lourdes"1
admin incluirjogador 2 "João Miguel"2
admin excluirjogador 3

jogador andarpasso 3 up
jogador andarpos 3 9,4
jogador exibir

pokemon evoluir 3, 14, 17
pokemon evoluir 3, 16, 19
```

Os elementos da linha de comando são separados por espaço em branco ou carácter de tabulação (' ' ou '\t') na linguagem C. Nomes em forma de cadeias de caracteres devem aparecer entre aspas, como no exemplo acima. O arquivo de saída deve emitir uma confirmação ou rejeição, em caso de inconsistência, para cada comando.

5 Elaboração e Documentação

A pasta do código contém uma moldura para ser estendida. Pode-se modificar o projeto, por exemplo dividir o código que falta em vários arquivos e modificar o `Makefile` em correspondência.

O resultado deve ser um sistema que lê um arquivo de entrada com a lista de comandos, um por linha, e produz o arquivo de saída. (Quem quiser aceitar o desafio de uma interface gráfica, esteja a vontade, mas não é facultativo).

Os alunos obrigatoriamente devem fornecer arquivos de exemplos testados. O usuário (neste caso o professor) não deve ter o trabalho de digitar nada, além da linha de comando da seguinte forma

```
prog <arquivo_de_entrada> <arquivo_de_saída>
```

O produto final deve ser um arquivo no formato `zip` com a seguinte sintaxe: `aluno1+aluno2.zip`. Atenção: Nenhum executável ou código objeto pode estar dentro do projeto, pois os serviços de e-mail, como, por exemplo, Hotmail recusam o transporte de tais arquivos por razões de segurança. O aluno se responsabiliza pelo envio e recepção correta. Em caso de problemas maiores de tráfego de rede (serviços UFES fora do ar), o aluno deve mandar novamente o arquivo original (encaminhamento da mensagem original) quando o serviço voltar. O arquivo deve conter uma **única** pasta com o nome `aluno1+aluno2`. Duas subpastas devem conter o código fonte e a documentação do projeto. O código deve ter um `Makefile` que me permite a compilação facilitada. O arquivo `aluno1+aluno2.zip` deve ser mandado como anexo **exclusivamente** copiando a hiperligação seguinte no browser ou cliente de E-mail:

```
mailto:thomas@inf.ufes.br?subject=Estruturas%20de%20Dados%20I:%20Entrega%20de%20trabalho%20
```

A documentação deve ser em forma de descrição de projeto, preferencialmente gerado por LaTeX, contendo os seguintes tópicos:

- Capa do Projeto
  - Título
  - Autoria

  - Data
  - Resumo
- Introdução
- Objetivos
- Metodologia
- Resultados e Avaliação
- Referências Bibliográficas

Rigidez na administração da memória dinamicamente alocada: Recomenda-se fortemente usar a ferramenta `valgrind`, e consequentemente uma variação do sistema operacional Unix. O vazamento e/ou violação de memória constitui uma degradação de qualidade do software e se refletirá na avaliação do trabalho.

Referências

- Pokémon Go
  `https://pt.wikipedia.org/wiki/Pok%C3%A9mon_GO`
- Lista de Pokémon
  `https://pt.wikipedia.org/wiki/Lista_de_Pok%C3%A9mon`

Bom trabalho!