Jogo da Velha

Programação de Computadores 1

ABI/TAI

Prof. Daniel Saad

1 Introdução

O jogo da velha é um jogo bem conhecido, especialmente por suas regras serem bem simples. Ele é jogado por dois jogadores e sobre um tabuleiro 3×3. Cada jogador possui uma marcação, geralmente são utilizados símbolos: 'O' e 'X'. Alternadamente, os jogadores colocam sua marcação sobre um espaço vazio do tabuleiro. Aquele jogador que conseguir três símbolos consecutivos primeiro, seja na vertical, horizontal ou diagonal, é declarado vencedor. Caso nenhum jogador conseguir a façanha, é declarado um empate.

O objetivo deste projeto é, dada uma configuração válida de um jogo da velha, presente um um arquivo de entrada, produzir todas as configurações vitoriosas válidas em um arquivo de saída do jogador cuja marcação é o 'O'.

2 Especificação

Os caminhos dos arquivos de entrada e saída devem ser capturados, nesta ordem, através da linha de comando durante a invocação do programa. O arquivo de entrada contém a configuração do jogo da velha, enquanto o arquivo de saída conterá todas as configurações vitoriosas válidas a partir da configuração inicial.

Deve-se considerar que a próxima jogada é realizada pelo jogador detentor da marcação 'O'.

2.1 Entrada

O arquivo de entrada possui três linhas, cada uma com três caracteres, descrevendo o jogo da velha.

Restrições

Os símbolos podem ser 'O', indicando uma marcação do primeiro jogador, 'X', indicando uma marcação do segundo jogador, ou ''B' indicando um espaço em branco.

2.2 Saída

Deverá ser impresso no arquivo de saída todas as configurações válidas de vitória do jogador detentor da marcação 'O' a partir da configuração inicial. É permitido que as configurações se repitam de acordo com o número de vezes em que for possível gerá-las a partir da configuração inicial.

2.3 Exemplos

Arquivo de Entrada

```
OXB
BOB
XOX
```

Arquivo de Saída

```
OXX
OOO
XOX

OXX
OOO
XOX
```

Arquivo de Entrada

```
XBO
XBB
OBX
```

Arquivo de Saída

```
XOO
XOX
OBX

XOO
XOB
OXX

XBO
XOB
OBX

XXO
XOO
OBX

XBO
XOO
OXX

XXO
XOB
OOX

XBO
XOX
OOX
```

2.4 Modularização

O sistema deverá ser dividido em módulos, cada um para cumprir uma tarefa. Estes módulos podem ser organizados internamente através de várias funções e eles correspondem aos seguintes:

- Módulo de tratamento de erros: trata erros associados aos argumentos transmitidos via linha de comando ou referentes à manipulação de arquivos.
- Módulo de entrada: efetua a leitura dos dados do arquivo de entrada.
- Módulo de saída: efetua a impressão das respostas para cada arquivo de entrada no arquivo de saída..
- Módulo de processamento: realiza o processamento da entrada para composição da saída.
- Módulo principal: contém a função `main` e as chamadas das funções exportadas pelos outros módulos. O ideal é que este módulo possua uma quantidade muito pequena de código, já que ele vai utilizar funções que estão presentes nos outros módulos.

Os módulos devem ser organizados em arquivos separados, com seus respectivos arquivos de cabeçalho e implementação.

2.5 Construção do sistema

Um `Makefile` deverá ser produzido para a compilação dos códigos-fontes no executável e deverá ser distribuído junto ao código.

2.6 Documentação

O código deve ser bem documentado, com presença de comentários explicando os trechos mais complexos do código. Além disso, um arquivo README deve ser providenciado com a devida identificação do autor descrevendo o projeto e instruindo como o código deve ser compilado através da ferramenta `make`.

3 Critérios de correção

Deve ser utilizada a linguagem de programação `C` para a implementação do caça-palavras.

Para validação da correção do algoritmo, testes automatizados serão realizados, então é **crucial** que a saída esteja conforme o especificado.

Serão descontados pontos dos códigos que não possuírem indentação.

3.1 Ambiente de Correção

Para a correção dos projetos, será utilizada uma máquina de 64-bits com sistema operacional `GNU/LINUX` e compilador `GCC >= 10.2.0`, logo é imprescindível que o sistema seja capaz de ser compilado e executado nesta configuração.

4 Considerações

- o `GDB`, `Valgrind` e ferramentas gráficas associadas podem ajudar na depuração do código.
- Este trabalho deve ser feito **individualmente**.
- Não serão avaliados trabalhos que não compilem ou sem a presença de um `Makefile`.
- Como a correção é automatizada, deverá ser impresso apenas o que a especificação pede. Atentem-se para a formatação da saída.
- A incidência de plágio será avaliada automaticamente com nota 0 para os envolvidos. Medidas disciplinares também serão tomadas.
- O trabalho deve ser entregue dentro de uma pasta zipada com a devida identificação do aluno no prazo combinado pelo ambiente virtual de aprendizagem da disciplina.