UNIVERSIDADE DO ESTADO DO AMAZONAS
ESCOLA SUPERIOR DE TECNOLOGIA
NÚCLEO DE COMPUTAÇÃO - NUCOMP

ALGORITMOS E ESTRUTURAS DE DADOS II 2017/2
PROF. FLÁVIO JOSÉ MENDES COELHO
09/10/2017

PROJETO PRÁTICO 1

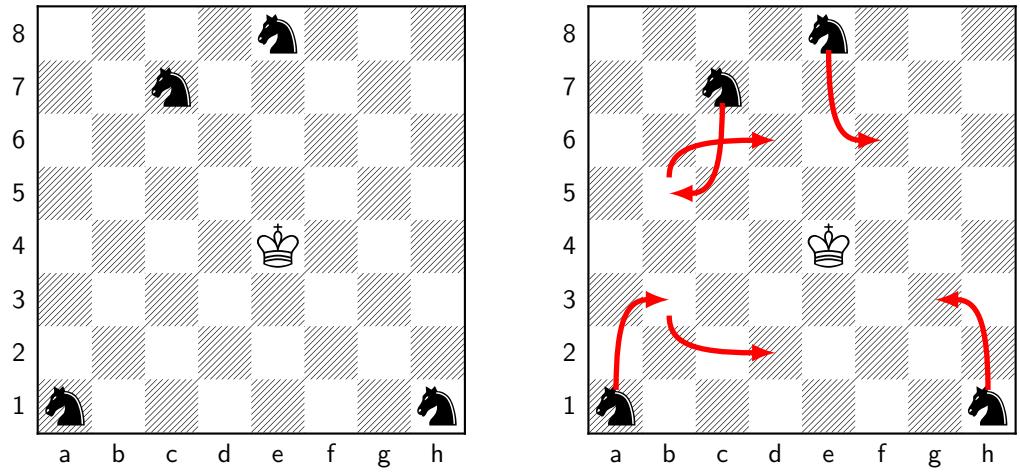
1 Objetivos

Este **Projeto Prático 1 – PP1**, tem o objetivo de exercitar e avaliar suas habilidades em:

- Codificar o tipo abstrato de dados grafo na linguagem de programação exigida neste enunciado, aplicar busca em largura e solucionar subproblemas relacionados;
- Demonstrar seu domínio sobre o código que você desenvolveu neste projeto e mostrar que sabe realizar modificações locais neste código à pedido de um avaliador;
- Apresentar argumentos lógicos, razoáveis, para questões práticas ou teóricas levantadas por um avaliador sobre o seu **PP1**.

2 Descrição do problema

Em um tabuleiro de xadrez há quatro cavaleiros negros e um rei branco em posições aleatórias. O objetivo é descobrir quais cavaleiros podem ameaçar o rei, realizando o mínimo de movimentos. Uma posição de ameaça ao rei é aquela posição (possivelmente alcançada após alguns movimentos) a partir da qual o cavaleiro alcaçaria a posição do rei com apenas mais um movimento. Note que um cavaleiro movimenta-se em "L" (consulte um manual de Xadrez).



Observe no tabuleiro à direita, que:

- O cavaleiro negro na posição `h1` move-se para a posição `g3`, e ameaça o rei.
- O cavaleiro negro na posição `c7` move-se para `b5` e, depois, para `d6`, e ameaça o rei.
- O cavaleiro negro na posição `e8` move-se para a posição `f6`, e ameaça o rei.
- O cavaleiro negro na posição `a1` move-se para a posição `b3`, em seguida, move-se para a posição `d2`, e ameaça o rei.

No exemplo dado, somente os cavaleiros em `h1` e `e8` fazem o número mínimo de movimentos até uma posição de ameaça ao rei. Utilize grafos e busca em largura para codificar uma solução para este problema. As diversas listas da lista de adjacência devem ser ordenadas em tempo inferior a $O(n^2)$, onde n é o número de vértices adjacentes a um determinado vértice.

3 Entradas e saídas do problema

Entrada. As posições dos quatro cavaleiros negros, e a posição do rei branco, separados por espaço e seguidos de um `enter`. Entrada para o exemplo da figura:

```
h1 c7 e8 a1 e4
```

Saída. O número de movimentos e a sequência de casas de movimentação mínima de cada cavaleiro, por linha, na ordem em que os cavaleiros foram fornecidos na entrada:

```
1 h1 g3
1 e8 f6
```

4 Requisitos do projeto

1. **Equipes**. Este projeto deve ser desenvolvido por uma equipe de três ou dois estudantes. Não serão aceitas equipes com menos de dois ou mais de três participantes. A violação deste requisito do projeto penalizará a equipe com a subtração de 50% dos pontos totais obtidos no projeto.

2. **Ferramentas e técnicas**. O projeto deve ser codificado em C++11 utilizando programação orientada a objetos com encapsulamento, pelo menos nos TADS empregados.

3. **Pontuação**. O **PP1** vale de 0,0 (nota mínima) a 10,0 (nota máxima), e será avaliado em duas fases:

   Fase 1 - Avaliação funcional. Pontuação mínima: 0,0; pontuação máxima: 5,0. O código do projeto será submetdido ao juiz online **run.codes**, e obterá a nota máxima desta fase, se passar em todos os casos de teste (cada caso terá uma pontuação correspondente a uma fração da nota máxima desta fase). Os detalhes sobre a submissão serão repassados pelo professor, por e-mail.

   Fase 2 - Inspeção de código. Pontuação mínima: 0,0; pontuação máxima: 5,0. Nesta etapa, o professor escolherá um membro da equipe para defender o projeto (qualquer membro ausente no momento da escolha do membro defensor receberá a nota mínima integral no projeto). Na inspeção de código, serão feitas perguntas sobre detalhes de implementação do projeto de acordo com os seguintes critérios:

   - **Legitimidade** (critério eliminatório). A constatação de que o projeto é plágio de outros implica na atribuição automática da nota mínima integral para o projeto incluindo as duas etapas de avaliação. Será considerado legítimo o projeto que tiver similaridade de código entre equipes menor do que 40% (comparador do juiz online do run.codes).
   - **Segurança na defesa** ao responder as perguntas do avaliador e explicar as estratégias utilizadas.
   - **Cumprimento de requisitos**: o projeto atende ao que foi solicitado neste enunciado.
   - **Qualidade de código**: indentação, uso de TADS orientados a objetos (encapsulados), uso do C++11*, programação genérica (quando aplicável), qualidade de código (implementação simples e eficiente das estruturas de dados, boa nomeação de identificadores, escolha das estruturas de dados e algoritmos mais eficientes em desempenho), e criatividade.
   - **Uso de bibliotecas**: o projeto poderá empregar somente os *conteiners* `pair` e `vector` da STL. Qualquer outro recurso da STL ou de outras bibliotecas não deve ser utilizado, com exceção de: `iostream`, `cstring`, `cstdlib` e `limits`. O uso de qualquer outro arquivo de cabeçalho padrão deve ser verificado com o professor.

     *** Compile seu projeto em vários compiladores online (além do compilador do juiz online) para garantir que o mesmo não apresente problemas sintáticos/semânticos (nem *warnings*). Localmente, aplique as diretivas de compilação mais restritas do seu compilador**.

5 Datas

- Emissão deste enunciado: 09/10/2017 às 15h (hora local).
- Abertura do juiz online: 12/10/2017 às 12h (hora local).
- Fechamento do juiz online: 20/10/2017 às 23h (hora local).
- Inspeção de código: 23 e 25/10/2017 de 14h40 às 17h10.

CÓDIGO DE ÉTICA

Este projeto deve ser concebido, projetado, codificado e testado pela equipe, com base nas referências bibliográficas fornecidas beste enunciado e nas aulas de Estruturas de Dados, ou por outras referências bibliográficas indicadas pelo professor. Portanto, não copie código pronto da Web para aplicá-lo diretamente a este projeto, nem copie código de colegas de outras equipes, ou mesmo permita que terceiros produzam este trabalho em seu lugar. Isto fere o código de ética desta disciplina e implica na atribuição da nota mínima ao trabalho.

Referências

[1] COELHO, Flávio. Slides das aulas de *Algoritmos e Esruturas de Dados II*. Disponível em https://est.uea.edu.br/fcoelho. Universidade do Estado do Amazonas, Escola Superior de Tecnologia, Núcleo de Computação - NUCOMP. Semestre letivo 2016/2.

[2] C++. In: *WIKIPÉDIA, a enciclopédia livre*. Flórida: Wikimedia Foundation, 2016. Disponível em: <https://pt.wikipedia.org/w/index.php?title=C%2B%2B&oldid=45048480>. Acesso em: 17 abr. 2016.

[3] C++. In: *cppreference.com*, 2016. Disponível em <http://en.cppreference.com/w/>. Acesso em: 17 abr. 2016.

[4] CORMEN, T. H., Leiserson, C. E., Rivest, R. L., Stein C. *Introduction to Algorithms*, 3rd edition, MIT Press, 2010

[5] KNUTH, Donal E. *Fundamental Algorithms*, 3rd.ed., (vol. 1 de The Art of Computer Programming), Addison-Wesley, 1997.

[6] KNUTH, Donal E. *Seminumerical Algorithms*, 3rd.ed., (vol. 2 de The Art of Computer Programming), Addison-Wesley, 1997.

[7] KNUTH, Donal E. *Sorting and Searching*, 2nd.ed., (vol. 3 de The Art of Computer Programming), Addison-Wesley, 1998.

[8] Spada E, Sagliocca L, Sourdis J, et al. Use of the Minimum Spanning Tree Model for Molecular Epidemiological Investigation of a Nosocomial Outbreak of Hepatitis C Virus Infection. Journal of Clinical Microbiology. 2004;42(9):4230-4236. doi:10.1128/JCM.42.9.4230-4236.2004.

[9] STROUSTRUP, Bjarne. *The C++ Programming Language*. 4th. Edition, Addison-Wesley, 2013.

[10] STROUSTRUP, Bjarne. *A Tour of C++*. Addison-Wesley, 2014.

[11] SZWARCFITER, Jayme Luiz et. alii. *Estruturas de Dados e seus Algoritmos*. Rio de Janeiro. 2a. Ed. LTC, 1994.

[12] WIRTH, Niklaus. *Algoritmos e Estruturas de Dados*. Rio de Janeiro. 1a. Ed. Prentice - Hall do Brasil Ltda., 1989.

[13] ZIVIANI, Nívio. *Projeto de Algoritmos com Implementação em Java e C++*. 2a. Edição. Cengage Learning, 2010.

[14] ZIVIANI, Nívio. *Projeto de Algoritmos com Implementação em Pascal e C*. 3a. Ed. São Paulo: Cengage Learning, 2012.