UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
http://www.icmc.usp.br/

Professor: Rodrigo Fernandes de Mello (mello@icmc.usp.br)
Alunos PAE: Lucas Pagliosa (lucas.pagliosa@usp.br)
Felipe Duarte (fgduarte@icmc.usp.br)
Monitores: Victor Forbes (victor.forbes@usp.br)

Trabalho 05: Labirinto

1 Prazos e Especificações

O trabalho descrito a seguir é individual e não será tolerado qualquer tipo de plágio ou cópia em partes ou totalidade do código. Caso seja detectada alguma irregularidade, os envolvidos serão chamados para conversar com o professor responsável pela disciplina.

A entrega deverá ser feita única e exclusivamente por meio do sistema run.codes no endereço eletrônico `https://run.codes` até o dia **08 de novembro de 2016 às 23 horas e 59 minutos**. Sejam responsáveis com o prazo final para entrega, o run.codes está programado para não aceitar submissões após este horário e não serão aceitas entregas fora do sistema.

Leia a descrição do trabalho com atenção e várias vezes, anotando os pontos principais e as possíveis formas de resolver o problema. Comece a trabalhar o quanto antes para que você não fique com dúvidas e que consiga entregar o trabalho a tempo.

2 Descrição do Problema

Seja uma fase de um jogo digital cujo cenário é um labirinto $\mathcal{L}$ 2D constituído de um conjunto $\mathbf{C}$ de NC câmaras que se comunicam por dutos. A forma de uma câmara do labirinto é dada por um polígono, mas, para simplificar, vamos desconsiderar a forma e caracterizar a geometria de uma câmara $c_i \in \mathbf{C}, 0 < i \leq NC$, somente por um ponto (que pode ser, por exemplo, o circuncentro do polígono). Também vamos desconsiderar a forma e representar os dutos do labirinto por um conjunto $\mathbf{S}$ de NS segmentos, onde um segmento $\mathbf{s}_i \in \mathbf{S}, 0 < i \leq NS$, é um caminho em linha reta que conecta dois pontos distintos de $\mathcal{L}$. Portanto, um labirinto $\mathcal{L}$ contém um conjunto $\mathbf{P}$ de NP pontos 2D, onde um ponto $\mathbf{p}_i \in \mathbf{P}$ de coordenadas $(p_i.x; p_i.y), 0 < i \leq NP$, pode ser a posição de uma câmara e/ou uma das extremidades de um ou mais segmentos de dutos de $\mathcal{L}$.

Uma câmara c_i do labirinto tem as seguintes propriedades:

- Índice $0 < c_i.v \leq NP$ do ponto $\mathbf{p}_{c_i.v}$ que define a posição da câmara.
- Número $c_i.n > 0$ de portais da câmara. Um portal pode estar ativo – o que significa que se pode passar por ele – ou inativo. Por um portal pode-se entrar em um segmento de duto (através do qual talvez seja possível alcançar outra câmara) ou sair do labirinto.
- Valor lógico $c_i.o$ que indica, se diferente de zero, que a câmara possui portal de saída do labirinto. Se uma câmara tiver contato com o mundo exterior, apenas um dos portais da câmara será o de saída; os demais são portais para segmentos de dutos de $\mathcal{L}$. Um portal de saída sempre é ativo (ver definição a seguir). Se a câmara tiver um portal de saída, este também pode ser usado, no início da fase, para entrada no labirinto.

Um segmento $\mathbf{s}_i$ do labirinto tem as seguintes propriedades:

- Índices $s_i.b$ e $s_i.e, 0 < |s_i.b| \neq |s_i.e| \leq NP$, dos pontos $\mathbf{p}_{|s_i.b|}$ e $\mathbf{p}_{|s_i.e|} \in \mathbf{P}$ das extremidades do segmento. Se o ponto $\mathbf{p}_{s_i.b}$ for a posição de uma:
  - Câmara c_k de $\mathcal{L}$, então significa que há um portal na câmara para o segmento s_i. Se $s_i.b < 0$, o portal está inativo; caso contrário o portal está ativo e através dele é possível entrar em c_k (vindo do ponto $\mathbf{p}_{s_i.e}$ na extremidade oposta) ou sair de c_k (em direção ao ponto $\mathbf{p}_{s_i.e}$ na extremidade oposta).
  - Extremidade de outro segmento s_j de $\mathcal{L}$, então significa que s_i e s_j fazem parte do mesmo duto de $\mathcal{L}$ e que se pode passar de um para outro. Neste caso, o sinal de $s_i.b$ é ignorado.
  - Caso ambas as condições não sejam satisfeitas para $\mathbf{p}_{|s_i.b|}$, então a extremidade correspondente do segmento s_i representa um "beco sem saída" do labirinto. As mesmas considerações valem para a extremidade do ponto $\mathbf{p}_{|s_i.e|}$.
- Comprimento $s_i.d$, dado pela distância entre os pontos $\mathbf{p}_{|s_i.b|}$ e $\mathbf{p}_{|s_i.e|}$ das extremidades do segmento.

A Figura 1 ilustra um labirinto $\mathcal{L}$ com 18 pontos e 13 câmaras, das quais 5 têm portal de saída (quadrados em verde) e 8 não (quadrados em vermelho), totalizando 21 segmentos. Neste exemplo a câmara inicial, definida como c_s, foi dada pela câmara 1. Os traços em vermelho próximos às câmaras nos pontos 3, 4, 9 e 16 indicam os quatro portais inativos (há algum bloqueio que não permite o acesso) do labirinto.

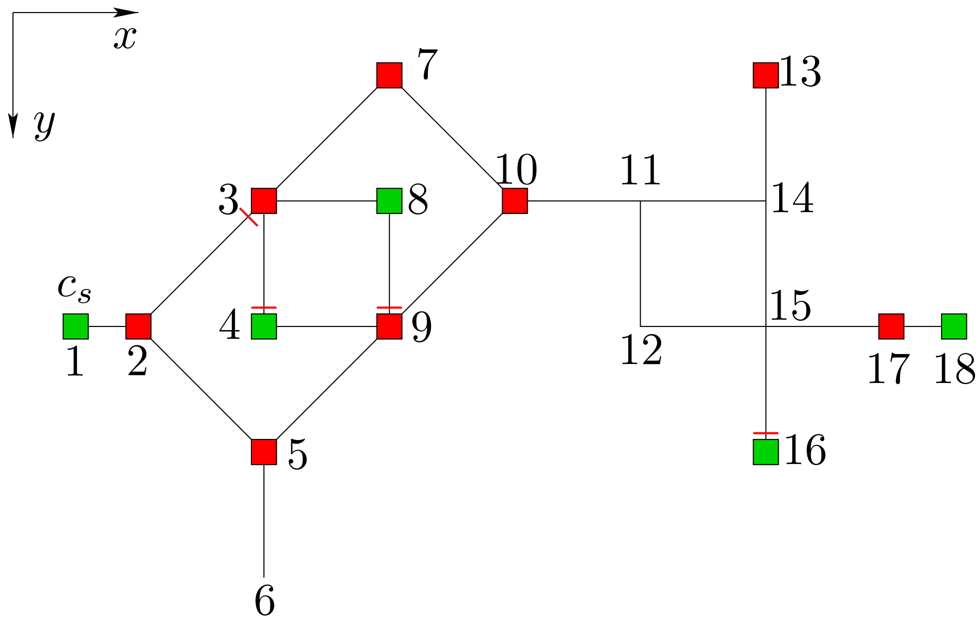

Figura 1: Exemplo de labirinto. Quadrados verdes e vermelhos representam câmaras com e sem saídas, respetivamente. Traços vermelhos ortogonais ao segmento s_i, próximos a uma câmara c_k, indicam que c_k está inativa por meio de s_i.

O trabalho consiste na implementação de uma aplicação **não recursiva** na linguagem C que, dados um labirinto $\mathcal{L}$ e uma câmara inicial c_s de $\mathcal{L}$ com um portal de saída, determina quais os caminhos para todas as saídas do labirinto (sempre haverá no mínimo uma saída, dada pela própria posição inicial de entrada) que podem ser alcançadas a partir de c_s através dos dutos de $\mathcal{L}$, tal como ilustrado na Figura 2 (a solução trivial, isto é, o próprio ponto inicial, embora não apareça na figura, também deve ser fornecido como parte da saída).

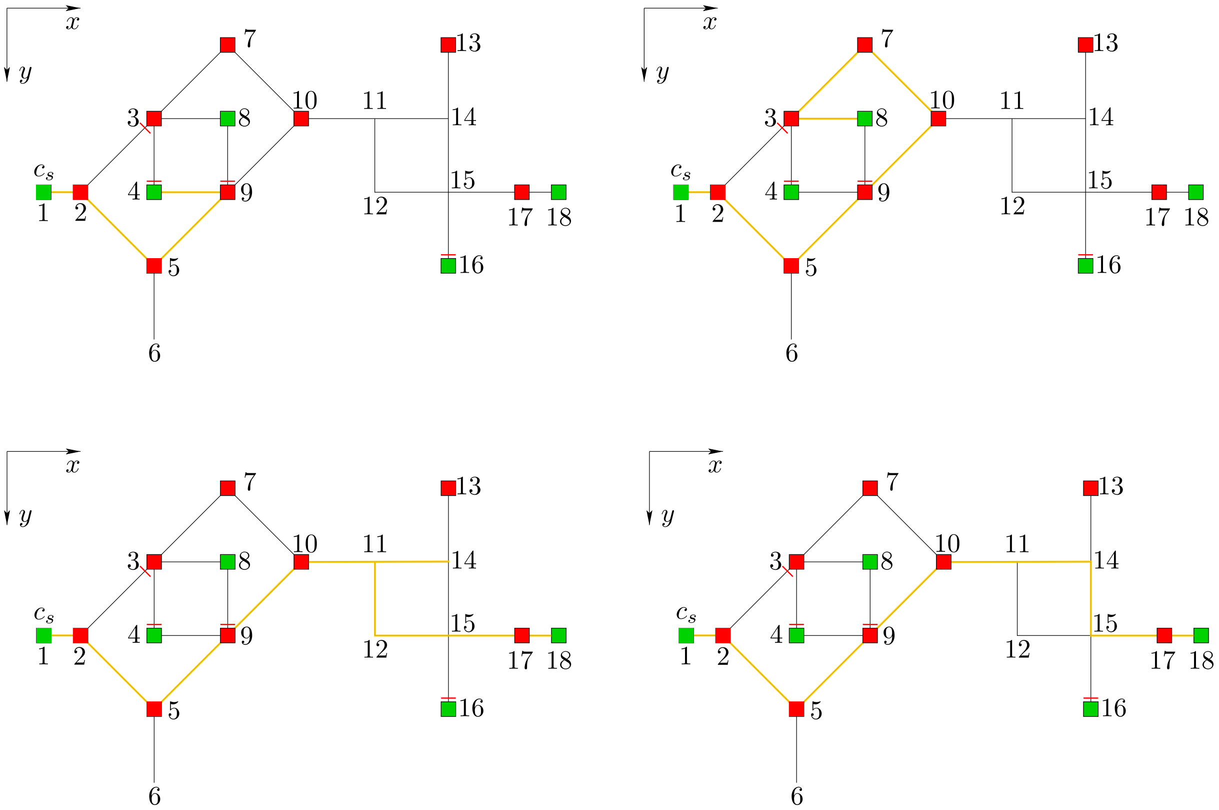

Figura 2: Caminhos possíveis (em amarelo) para o labirinto da Figura 1.

3 Arquivos de Entrada e Saída

Os dados do arquivo de entrada definem os pontos, as câmaras e os dutos de um labirinto, tal como definido na Seção 2, assim como a câmara inicial, no seguinte formato:

- Número de pontos NP
- `float` $p_1.x$ $p_1.y$
- `float` $p_2.x$ $p_2.y$
- $\vdots$
- `float` $p_{NP}.x$ $p_{NP}.y$
- Número de câmaras NC
- $c_1.v$ $c_1.o$
- $c_2.v$ $c_2.o$
- $\vdots$
- $c_{NC}.v$ $c_{NC}.o$
- Número de segmentos NS
- $s_1.b$ $s_1.e$
- $s_2.b$ $s_2.e$
- $\vdots$

- $s_{NS}.b$ $s_{NS}.e$
- $c_s.v$

Vale a pena seu programa fazer os seguintes testes de integridade:

- Para cada câmara c_i, se $0 < c_i.v \leq NP$.
- Para cada segmento s_i, se $0 < |s_i.b| \neq |s_i.e| \leq NP$ e $s_i.d \neq 0$.
- Se $0 < c_s.v \leq NP$ e $c_s.o \neq 0$.

Os dados de entrada para o labirinto da Figura 1 são:

```
18
1.0 5.0
2.0 5.0
4.0 3.0
4.0 5.0
4.0 7.0
4.0 9.0
6.0 1.0
6.0 3.0
6.0 5.0
8.0 3.0
10.0 3.0
10.0 5.0
12.0 1.0
12.0 3.0
12.0 5.0
12.0 7.0
14.0 5.0
15.0 5.0
13
1 1
2 0
3 0
4 1
5 0
7 0
8 1
9 0
10 0
13 0
16 1
17 0
18 1
21
1 2
2 -3
2 5
3 -4
5 6
3 7
3 8
4 9
```

```
5 9
7 10
8 -9
9 10
10 11
11 12
11 14
12 15
13 14
14 15
15 -16
15 17
17 18
1
```

O arquivo de saída contém, para cada caminho solução $\{pw_1, pw_2, \cdots, pw_n\}$, uma linha contendo i) o número n de pontos do caminho; ii) a sequência de índices $w_i, 0 < i \leq NP$; iii) a soma de todos os segmentos percorridos (faça um `cast` para um número inteiro, a fim de evitar qualquer tipo de erro de arredondamento no run.codes); e iv) uma quebra de linha. A ordem de saída dos caminhos, seguindo este modelo, é dada pela seguinte regra:

- Os menores caminhos segundo a distância Geodésica dos pontos, isto é, pela soma dos comprimentos dos segmentos desde a câmara inicial até uma câmara saída.
- Se houver empate no caso anterior, ordenar pelo número de pontos desde a câmara inicial até uma câmara saída.
- Se ainda houver empate no caso anterior, ordenar levando em conta a ordem crescente dos índices dos pontos do caminho solução. Por exemplo, suponha que você encontre os seguintes caminhos soluções $\{5, 3, 2\}$ e $\{5, 4, 1\}$ saindo da câmara inicial $c_s = 5$ e eventualmente encontrando as câmaras de saída 1 e 2. Suponha ainda que estes caminhos possuírem a mesma distância Geodésica. Como existem 3 pontos da câmara inicial até a câmara final, você irá informar no arquivo de saída as soluções $\{5, 3, 2\}$ e $\{5, 4, 1\}$, nesta ordem, pois $5 = 5, 3 < 4$. Você não deve ordenar e comparar os caminhos como $\{2, 3, 5\}$ e $\{1, 4, 5\}$ e informar as saídas na ordem $\{5, 4, 1\}$ e $\{5, 3, 2\}$.

A saída esperada para o labirinto da Figura 1 (conforme ilustrado na Figura 2) é:

```
1 1 0\n
5 1 2 5 9 4 8\n
8 1 2 5 9 10 7 3 8 17\n
10 1 2 5 9 10 11 12 15 17 18 18\n
10 1 2 5 9 10 11 14 15 17 18 18\n
\32
```

4 Dicas

Há dois problemas para encontrar uma saída. O primeiro é que, a partir de um ponto **p**, pode haver mais de um caminho possível a ser tomado. Qual deles você deve seguir? E se um caminho não levar a uma saída, o que fazer? A resposta é que, sem nenhuma outra pista, você pode tomar qualquer caminho, mas deve ser capaz de voltar ao ponto **p** a fim de tentar os outros trajetos. Para isso, você pode usar uma pilha com o intuito de lembrar quais são os caminhos alternativos a partir de um ponto **p**. O segundo problema é que você deve tratar caminhos já trilhados anteriormente e, assim, não "andar em círculos". Portanto, você deve lembrar por onde já passou a fim de não repetir indefinidamente o mesmo trajeto. Para isso, outra estrutura de dados deve ser usada.

5 Observações importantes

- Programe as impressões na tela EXATAMENTE como exemplificado no decorrer deste documento. Tome cuidado com pulos de linha, tabs, espaços, etc.
- Coloque dentro do zip todos os arquivos de código (`*.h *.c`), o makefile e um arquivo texto com o nome e número USP.