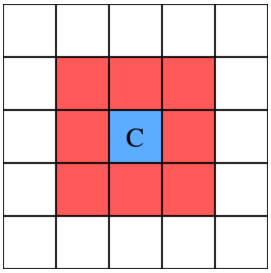
**Universidade Federal do ABC**
**BCM0505–15 — Processamento da Informação — Prática**
**Projeto — Campo Minado**
Primeiro Quadrimestre de 2018

**Professores**
Thiago Ferreira Covões <thiago.covoes@ufabc.edu.br>
Emilio Francesquini <e.francesquini@ufabc.edu.br>

**Versão Atualizada: 09/04/2018**
- Função **escolheAuto** teve sua assinatura e descrição atualizada.

# 1 Introdução

Neste projeto você desenvolverá o famoso jogo do **Campo Minado**. Este é um jogo bastante simples e de muito sucesso. O jogo consiste em um campo de tamanho retangular ($N$ linhas por $M$ colunas). Inicialmente todas as casas (células) deste campo têm seu conteúdo "escondido". Cada casa contém uma das seguintes opções: (i) uma bomba; (ii) um número de 0 a 8 indicando o número de bombas nas casas vizinhas. Dada uma casa $C$ — posição $(x, y)$ — as casas vizinhas são definidas pelas 8 casas ao seu redor, conforme ilustrado na Figura 1.




O jogador deve escolher uma casa cujo conteúdo deve ser revelado a cada turno. A escolha é realizada ao clicar com o botão esquerdo do mouse sobre a casa. Caso a casa escolhida contenha uma bomba, o jogador perde o jogo. Ao escolher uma casa que tenha conteúdo 0, ou seja, que não tenha nenhuma bomba no seu entorno, todas as casas vizinhas são reveladas. Se uma dessas casas vizinhas que foram reveladas também contiver um 0, o processo se repete e as suas casas vizinhas também deverão ser reveladas. Esse processo de abertura em cascata deve se repetir até que nenhuma célula com valor 0 seja alcançada. O jogador ganha o jogo se:

- todas as casas sem bomba forem reveladas;

- todas as casas com bombas forem **marcadas** corretamente.

Para marcar uma casa o jogador deve clicar com o botão direito sobre ela. Ao fazer isso, aquela casa recebe uma marcação especial impedindo que seja revelada por acidente nas jogadas futuras. Caso o jogador marque um número de casas igual ao número de bombas no campo, ele vence o jogo caso as tenha marcado corretamente e perde caso contrário. Para desmarcar uma casa basta clicar com o botão direito novamente.

## 2 Projeto

Para desenvolver esse jogo sua equipe terá acesso a um projeto do NetBeans com algumas partes do jogo já implementadas, por exemplo, a interface gráfica. O trabalho da sua equipe é desenvolver as funções específicas que serão chamadas durante o jogo. Essa é uma excelente oportunidade para você conhecer um pouco mais sobre Java, no entanto, para o desenvolvimento do projeto não é necessária a compreensão em detalhes do código fornecido.
Adicionalmente às funções tradicionais do Campo Minado, você também implementará uma função que escolhe a próxima jogada a ser feita. Se sua função for realmente boa, o uso repetido dela ganhará o jogo com frequência!
Você deve finalizar a implementação dos métodos da classe CampoMinadoUtils. Abaixo você pode verificar a assinatura de cada método.

```
1  public class CampoMinadoUtils{
2
3      public static int[][] geraCampo(int numMinas,
4              int numLinhas,
5              int numColunas) {/*...*/}
6
7      public static boolean temBombaNoEntorno(int [][]campo,
8              int linha, int coluna) {/*...*/ }
9
10     public static int numBombasNoEntorno(int [][]campo,
11             int linha, int coluna) {/*...*/}
12
13     public static boolean revela(int [][]campo,
14             int linha, int coluna) {/*...*/}
15
16     public static void marcou(int [][]campo,
17             int linha, int col) {/*...*/}
```

```
18
19      public static void escolheu(int [][]campo,
20              int linha, int col) {/*...*/}
21
22      public static int[] escolheAuto(int [][] campo) {/*...*/}
23
24      public static boolean fimJogo(int [][]campo) {/*...*/}
25  }
```

O campo de jogo será representado pela matriz `int[][] campo`. Os valores válidos para cada uma das posições nesta matriz variam de 0 até 29 e têm os seguintes significados:

| Valor | Estado | Descrição |
|---|---|---|
| 0 a 8 | Fechada | Número de bombas nas células ao redor |
| 9 | Fechada | Célula contém uma bomba |
| 10 a 18 | Aberta | Número de bombas nas células ao redor |
| 19 | Aberta | Célula contém uma bomba |
| 20 a 28 | Fechada e Marcada | Número de bombas nas células ao redor |
| 29 | Fechada e Marcada | Célula contém uma bomba |

Segue a descrição do que cada função deve fazer:

**geraCampo:** esta função é chamada no início de cada jogo e deve sortear a localização e preencher o campo de jogo com `numMinas` minas. As células que contiverem minas serão marcadas atribuindo-se a elas o valor 9. Após essa distribuição das minas pelo campo, a matriz deve ser modificada de forma que cada uma das células que não contém uma mina contenha o valor de 0 a 8 indicando o número de bombas nas sua vizinhança.

**temBombaNoEntorno:** esta função deve retornar `true` no caso de alguma célula no entorno de `campo[linha][coluna]` conter uma bomba, e `false` caso contrário;

**numBombasEntorno:** esta função deve retornar o número de bombas na vizinhança de `campo[linha][coluna]`;

**revela:** esta função será chamada quando for preciso revelar/abrir uma célula do jogo. O valor de `campo[linha][coluna]` deve ser atualizado para refletir que a célula foi revelada. Células fechadas contêm valores de 0-9, células abertas/reveladas contêm valores de 10-19. Valores `v % 10 == 9` indicam, portanto, que aquela célula contém uma mina.

**fimJogo:** esta função deve devolver `true` se o jogo terminou, e `false` caso contrário;

**marcou:** esta função é chamada quando o usuário pede para marcar (clica com o botão direito) a célula `campo[linha][coluna]`. Note que uma célula marcada deve conter valores entre 20 e 29 indicando a marcação; e o número de bombas nas células vizinhas ou a presença de uma bomba nela própria.

**escolheu:** esta função é chamada quando o usuário pede para revelar (clica com o botão esquerdo) na célula `campo[linha][coluna]`.

**escolheAuto:** esta função é chamada quando o usuário pede que o seu programa jogue automaticamente a próxima rodada, ou seja, escolha qual célula deve ser revelada e a revele automaticamente. **Atenção:** esta função recebe o campo *modificado*, todos os valores que ainda não foram revelados (valores entre 0 e 9) são substituídos por -1 e todas as marcas feitas são consideradas como marcas certas (valor 29) *mesmo se estiverem erradas*. Portanto, seu código deve escolher uma das posições com valor -1 e *retornar* as coordenadas em um vetor de inteiros de duas posições (linha e coluna, respectivamente).

**fimJogo:** esta função deverá devolver `true` caso o estado atual do jogo indique que ele terminou. O jogo termina, por exemplo,quando o jogador marcou todas as bombas (correta ou incorretamente) ou quando ele já tiver revelado todas as casas que não contêm bombas.

# 3 Entrega e avaliação

- O projeto deve ser feito por grupos com 2 ou 3 integrantes.
- Você deverá enviar o arquivo `CampoMinadoUtils.java` contendo a sua implementação da classe `CampoMinadoUtils` até o dia **25/04/2018 às 23:55** via atividade no TIDIA.
- Arquivos enviados com erros de compilação (ou que não contenham o código fonte) receberão nota **zero**.
- A avaliação do projeto será feita em duas partes. Uma automática (que contará inclusive com detector automático de plágio) e outra durante a apresentação para o professor.
- A apresentação do projeto pelo grupo deverá ser **agendada** com o professor por email entre os dias **27/04 e 11/05**. **Todos** os integrantes do grupo deverão estar presentes.

- Caso algum grupo não possa apresentar em um dia/horário fora do horário da aula, este(s) grupo(s) deverão apresentar para o professor seu trabalho obrigatoriamente no dia **27/04**.
- As apresentações deverão ter duração total de 10 minutos. 5 minutos para o grupo e 5 minutos de perguntas elaboradas pelo professor. Durante a fase de arguição o professor determinará qual aluno deverá responder a cada pergunta. No caso de algum integrante do grupo falhar em responder uma pergunta a nota do trabalho (e não apenas a do aluno em questão) será zero.
- Fraudes de qualquer tipo não serão toleradas. Caso seja determinado que houve fraude durante a elaboração do trabalho, **todos** os envolvidos serão **reprovados** (conceito F) na disciplina.

# 4 Bônus

Todos os programas passarão por uma avaliação automática. Durante este processo a função `escolheAuto` será utilizada para fazer o seu programa jogar sozinho algumas dezenas de rodadas. Caso o seu programa seja capaz de ganhar da estratégia aleatória (ou seja, caso seu programa seja melhor do que simplesmente abrir as casas do campo a esmo) o seu projeto receberá um ponto adicional, em outras palavras você pode ficar com nota 11 no projeto. Caso o seu programa seja o melhor da turma, ele poderá receber até 2 pontos extras (total 12).