INF101 – Introdução à Programação II

Roteiro de Prática: 18 de fevereiro de 2021

Introdução

Campo minado é um jogo muito popular para computador. É projetado para um jogador que tem de revelar um campo contendo minas explosivas sem detonar nenhuma. Foi inventado por Robert Donner em 1989.

A área de jogo (tabuleiro) consiste em um campo retangular contendo células (quadrados), ou seja, é uma matriz. Cada célula possui:

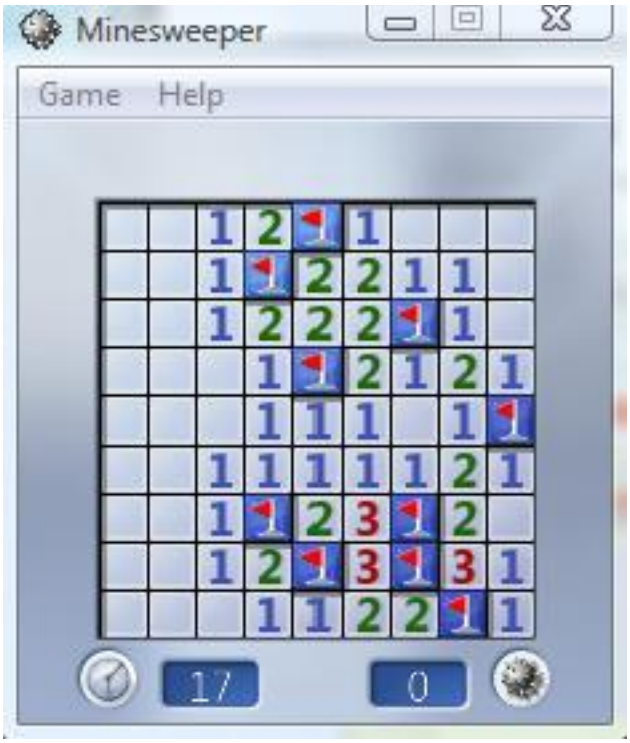
- ou uma mina explosiva
- ou um número indicando a quantidade de células adjacentes que contêm minas

Uma célula é adjacente a outra se um dos pontos da lateral está conectado a um ponto da lateral de outra célula. Assim sendo, cada célula pode ter até 8 células adjacentes:

1. superior à esquerda
2. superior ao centro
3. superior à direita
4. à esquerda
5. à direita
6. inferior à esquerda
7. inferior ao centro
8. inferior à direita

Observe que células que estão nas bordas do tabuleiro podem não conter todas as células adjacentes.

Na imagem a seguir, cada bandeira vermelha representa a posição de uma mina. Os números em cada célula representam a quantidade de minas nas células vizinhas (adjacentes). As células em branco indicam que não há minas nas células vizinhas.



Instruções

1. Baixe do Sistema de Entrega de trabalhos o seguinte arquivo: `p03_esqueleto.py`. Para que a saída do programa possa rodar de maneira satisfatória na sua máquina pessoal, adicione a biblioteca `termcolor` na sua plataforma de desenvolvimento de programas. Por exemplo, em uma janela de terminal ou *prompt* de comando, rode o comando:

   ```
   pip install termcolor
   ```

   Quem não conseguir rodar a implementação com a biblioteca `termcolor`, comente a linha do `from termcolor import colored`. Apague a implementação da função `imprimeTabuleiro`. E use a função `imprimeTabuleiroSimples` nas duas chamadas de impressão do tabuleiro na função `main`.
2. Abra o IDLE e crie um novo arquivo-fonte denominado `p03.py`. Copie para ele o conteúdo de `p03_esqueleto.py`. Não se esqueça de salvá-lo de tempos em tempos, porque pode ocorrer falha de energia elétrica durante a aula prática.
3. Preencha os comentários obrigatórios (nome, matrícula, data e uma breve descrição sobre o que o programa faz).
4. Estruture seu programa em seis funções: `main(numero_linhas, numero_colunas, numero_bombas)`, `imprimeTabuleiro(tabuleiro)`, `sorteiaPosicao(tabuleiro, semente=None)`, `inicializaTabuleiro(linhas, colunas)`, `posicionaBombas(quantidade, tabuleiro, semente=None)` e `calculaTabuleiro(tabuleiro)`.
5. A função `main` já está pronta (favor não mexer nela) e faz o seguinte: inicializa o tabuleiro, posiciona aleatoriamente as bombas (minas), imprime o tabuleiro inicial com as bombas posicionadas, calcula (analisa) o número de minas adjacentes a cada célula e, finalmente, imprime o tabuleiro analisado.
6. A função `imprimeTabuleiro` ou `imprimeTabuleiroSimples` também já está pronta (não mexer nelas). Veja como foi implementada. Se houver alguma dúvida, pergunte ao professor/assistente.
7. A função `sorteiaPosicao` também já está implementada (favor não mexer nela). Analise-a. Se houver dúvida, pergunte ao professor/assistente.
8. Implemente a função `inicializaTabuleiro(linhas, colunas)`. Ela cria um arranjo bidimensional de tamanho `linhas` × `colunas` de números inteiros todo zerado. E depois retorna este arranjo.
9. Implemente a função `posicionaBombas(quantidade, tabuleiro, semente=None)`. De acordo com a `quantidade` de bombas, você sorteia uma posição `(i, j)` por meio da função (já pronta) `sorteiaPosicao` e atribui -1 a esta posição no `tabuleiro`.
10. A implementação da função `calculaTabuleiro(tabuleiro)` é a mais exigente da aula de hoje. Você deve fazer uma varredura completa do `tabuleiro`. Se a posição `(i, j)` tiver uma bomba (valor -1), não há nada a fazer. Caso contrário, você tem de considerar os vizinhos de cada célula `(i, j)`:

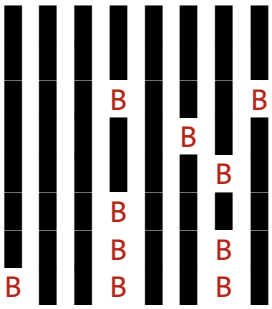
		j	
	(i-1, j-1)	(i-1, j)	(i-1, j+1)
i	(i, j-1)	**(i, j)**	(i, j+1)
	(i+1, j-1)	(i+1, j)	(i+1, j+1)

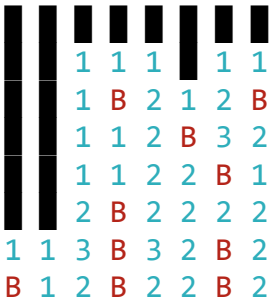
   Além disso, nenhuma linha ou coluna pode estar fora do tabuleiro. O valor mínimo de uma linha ou coluna é 0 (zero) e o valor máximo é o tamanho do tabuleiro `linhas` × `colunas`,

respectivamente. Para saber o número de bombas, faça uma contagem das bombas ao redor da célula `(i, j)`, conforme o esquema dados acima.

11. Teste seu programa com uma semente pré-fixada, digamos 111, na hora de chamar a função `posicionaBombas` dentro da função `main`.
12. Se seu programa entrar em *laço infinito,* digite CTRL-C na janela do *Shell*, para interromper a execução do programa. Após, conserte-o.

Exemplo de execução do programa





☞ Não se esqueça de preencher o cabeçalho do código fonte com seus dados, a data de hoje e uma breve descrição do programa.

Após certificar-se de que seu programa esteja correto, envie o arquivo do programa fonte (`p03.py`) através do sistema de entrega do LBI.