Universidade Federal Rural de Pernambuco – UFRPE
Curso: Bacharelado em Sistemas de Informação – BSI
Disciplina: Introdução à Programação – Turma SI2 - Prof: Leopoldo Teixeira

Primeiro Projeto – 2014-1 – Valor: 10% (10 pts) da 1ª V.A.

Instruções:
- O projeto deve ser feito em grupos de até 2 alunos.
- Data de Entrega: 02/06/2014 até às 23h59m por email.
- Apresentação pelo grupo: dia 03/06/2014 no horário da aula.
- Até a data de entrega, fazer o upload do projeto em uma conta no GitHub (www.github.com) e enviar o link para o email leo@leopoldomt.com com o assunto: [Projeto1-IP-2014-1]. Informar os nomes dos componentes do grupo no conteúdo do email.
- As equipes podem (e devem!) tirar dúvidas com o professor antes da entrega do projeto, apresentando as dificuldades encontradas para discutirmos possíveis ideias e soluções;
- Apresentar o código ao professor, em sala, no dia da aula
- Condições para receber nota 0 (zero):
    1. Entrega fora do prazo estabelecido;
    2. Código com erros de sintaxe;
    3. Programa incompleto;
    4. Código com alta similaridade com algum de outro(s) grupo(s);
    5. Não saber explicar o código.

Neste projeto, sua equipe irá desenvolver um jogo inspirado em"Batalha Naval".
Este jogo deve considerar as seguintes opções de configuração:

- O tamanho do grid é definido no momento que o jogo é iniciado, mas deve ser maior que 3x3;

- Cada Navio ocupa apenas uma posição (1x1) no grid. A quantidade de navios é definida após a definição do tamanho do grid, e deve ser menor que a quantidade de posições do grid;

- Após a definição do tamanho do grid e quantidade de navios, deve-se distribuir aleatoriamente os navios em posições do grid. Por exemplo, em um grid 3x3 com 3 navios, uma possível distribuição é ilustrada abaixo (pontos representam posições sem navios e a letra N representa a posição de um navio):

```
    1   2   3
1   .   .   N
2   .   N   .
3   N   .   .
```

Neste caso, as posições dos navios correspondem às coordenadas `(1,3), (2,2), (3,1).` Para distribuir aleatoriamente os navios, usem a biblioteca `random` de Python. No início do código incluam `"from random import *"`. Para gerar um número aleatório utilize o comando `randint(inicio,fim)`. Por exemplo, o comando a seguir gera um número aleatório entre 1 e 10: `num = randint(1,10)`. Mais detalhes e exemplos em https://docs.python.org/2/library/random.html. Cuidado para não colocar dois navios na mesma posição, isto não deve ser permitido!

- Finalmente, o programa deve perguntar a quantidade de erros que o usuário poderá efetuar. Um erro consiste em tentar acertar uma coordenada que não está ocupada por um navio. A quantidade de erros deverá ser maior do que a quantidade de navios distribuídos no grid e menor do que o número de posições sem navios no grid. Quando o usuário acerta um navio, a quantidade de erros não é alterada.

Com todas estas opções definidas, o jogo então é iniciado com a exibição do grid preenchido por pontos. Por exemplo, no caso de um grid 3x3, com os navios posicionados conforme ilustrado acima, o programa exibirá:

```
    1   2   3
1   .   .   .
2   .   .   .
3   .   .   .
```

O usuário então deverá digitar as coordenadas X e Y do próximo tiro. A equipe é livre para definir a maneira como as coordenadas serão informadas. Após informadas as coordenadas, o programa deverá atualizar o mapa com a ação correspondente. No caso de acertar um navio, a posição deverá exibir **`X`**, e no caso de acertar a água, a posição deverá exibir **`O`**.

Continuando o exemplo acima, caso o usuário atire na coordenada `(2,2)` deverá ser exibida uma mensagem informando que o usuário acertou um navio. Após a mensagem, o mapa deverá ser exibido novamente, com a atualização das posições, e será solicitada uma nova coordenada.

```
   1   2   3
1  .   .   .
2  .   X   .
3  .   .   .
```

Continuando o exemplo acima, caso o usuário atire agora na coordenada `(1,2)` deverá ser exibida uma mensagem informando que o usuário acertou a água e outra mensagem informando a quantidade de tentativas restantes para o usuário. Após as mensagens, o mapa deverá ser exibido novamente, com a atualização das posições, e será solicitada uma nova coordenada.

```
   1   2   3
1  .   O   .
2  .   X   .
3  .   .   .
```

O usuário deve ser informado quando tenta atirar em posições já utilizadas.
O jogo termina quando a quantidade máxima de erros é alcançada ou quando o jogador destrói todos os navios.

A equipe é livre para fazer versões mais elaboradas do jogo, como a possibilidade de jogar contra o computador ou contra outro usuário. Nestes casos de mais de um usuário, deve haver mais de um mapa, o seu e o do oponente. No seu mapa a posição dos navios pode ser definida manualmente. No do oponente, caso seja o computador, a posição permanece sendo gerada de forma aleatória. Outras modificações incluem navios com tamanhos diferentes, além de sugestões que a equipe pode fazer ao professor. A equipe pode ser bonificada com ponto extra de acordo com o esforço adicional de implementação destas funcionalidades.