COLÉGIO DO SAGRADO CORAÇÃO DE MARIA LISBOA

APLICAÇÕES INFORMÁTICAS B

Projeto 2 – Jogo das Minas

Projeto 2.º Período Ano letivo 2015/2016

Aluno (a) _____ N.º_____

Data ___/ ___/ ____ Ano ___.º Turma _____

O Docente _____

Introdução

Este projeto tem como objetivo fundamental o de mobilizar os conhecimentos adquiridos na área da programação orientada a eventos de modo a construir o Jogo das Minas.

Condições e Duração

Este projeto, a pares (podendo haver exceções), tem a duração de 6 aulas de 60 minutos e deve ser realizado maioritariamente em sala de aula com a supervisão do docente.

Requisitos

Deverá seguir as seguintes indicações que contém os requisitos do projeto.

O programa deverá cumprir ainda os seguintes requisitos:

#1. O programa deverá ser configurado para suportar multilíngua; *[5 pontos]*

#2. O programa deverá contemplar, pelo menos os idiomas português e inglês (*pt* e *en*); *[15 pontos]*

#3. O *Form* principal deverá ter o título de "Sagrado * Minas"; *[5 pontos]*

#4. Deverá conter um menu principal com os seguintes itens: *[40 pontos]*

4.i. File > Exit – deverá sair da aplicação *[5 pontos]*

```
Application.Terminate;
```

4.ii. View > Reveal Board – deverá revelar o conteúdo do tabuleiro quando estiver selecionado; *[10 pontos] (útil para debug do programa)*

4.iii. Game > New – deverá limpar os pontos e o tabuleiro e iniciar um novo jogo *[5 pontos]* segundo as regras definidas em #6.

4.iv. Help > About – Deverá abrir um novo form que deverá conter: Logo (original e personalizado) , Versão do jogo e Autores; *[15 pontos]*

#5. Deverá conter uma *label* que vai mostrando o número de bandeiras que já foram colocadas no jogo; Deverá conter sempre o *valor de número de minas – número de bandeiras colocadas*; *[5 pontos]*

5.i. A *label* deverá começar com o valor de 10 (minas);

#6. Regras do jogo:

6.i. Deverá ser definido um tabuleiro com 8x8; *[5 pontos]*

6.ii. Num novo jogo, deverão ser colocadas aleatoriamente 10 minas no tabuleiro; *[10 pontos]*

6.iii. Deverão ser calculadas, para cada casa que não contenha uma mina, quantas minas existem nas em todas as células adjacentes (8 no caso da casa não se encontrar numa aresta ou vértice); *[25 pontos]*

6.iv. O tabuleiro deve começar completamente tapado com todas as casas fechadas; *[5 pontos]*

6.v. Ações do rato:

6.v.1. Casa selecionada com o botão do lado esquerdo do rato:

6.v.1.A. Se esta casa contiver uma mina, o jogo deverá terminar e deverá ser apresentada uma mensagem a indicar que o utilizador perdeu; *[10 pontos]*

6.v.1.B. Se esta casa contiver um número de minas nas casas adjacentes **superior a 0**, então esta casa deverá ficar aberta e deverá mostrar esse valor; *[10 pontos]*

6.v.1.C. Se esta casa contiver um número de minas nas casas adjacentes **igual a 0**, então esta casa deverá ser aberta e deverá abrir todas as casas adjacentes seguindo estas mesmas regras definidas nestes pontos 2 e 3; *[10 pontos]*

6.v.2. Casa selecionada com o botão do lado direito do rato:

6.v.2.A. Se a casa **estiver** marcada com uma bandeira, então deverá ser desmarcada; *[5 pontos]*

6.v.2.B. Se a casa **não estiver** marcada com uma bandeira, então deverá ser ficar marcada com uma; *[5 pontos]*

6.vi. O jogo termina nas seguintes condições:

6.vi.1. Quando o jogador abrir uma casa que contém uma mina; *ver 6.v.1.A;* Neste caso deverá aparecer uma mensagem a indicar que o jogo acabou;

6.vi.2. Quando o número casas fechadas no tabuleiro é igual ao número de minas; Neste caso deverá aparecer uma mensagem a indicar que o jogador ganhou! *[10 pontos]*

#7. Outros requisitos *[15 pontos]*:

7.i. O design utilizado em toda a aplicação, nomeadamente no desenho do tabuleiro *[5 pontos]*

7.ii. Possibilidade da utilização de um *timer* onde o jogador tem um determinado tempo para completar o jogo, findo o qual o jogador perde; *[10 pontos]*

Avaliação

O seguintes itens serão considerados na avaliação: *[25 pontos]*

- O código fonte: *[18 pontos]*
  - Deverá estar devidamente documentado com comentários pertinentes; *[6 pontos]*
  - Deverá estar identado e formatado segundo as regras de estilo que estão a ser utilizadas nas aulas e que podem ser verificadas no portfólio do professor; *[6 pontos]*
  - Deverá estar estruturado o mais possível em sub-rotinas (*functions* e *procedures*); Cada *function* ou *procedure* deverá ter um comentário onde se descreve o seu objetivo. *[6 pontos]*
- O programa deve compilar:
  - Sem erros (*errors*); *[5 pontos]*

- Sem avisos (*warnings*); *[2 pontos]*

- O programa deverá correr sem erros de lógica cumprindo integralmente todas as funcionalidades pedidas nos requisitos técnicos;
- No final os alunos poderão ser chamados a “defender” os eu trabalho, podendo a classificação final ser alterada consoante o desempenho do aluno.

Algumas dicas

Para criar um retângulo com os cantos arredondados:

```
Bitmap.Canvas.RoundRect(x1,y1,x2,y3, dx, dy);
Exemplo:
Bitmap.Canvas.RoundRect(1,1,50,50, 20, 20); //os dois últimos parâmetros indicam o nível de arredondamento dos cantos.
```

Para colocar o Bitmap em modo antialias (esbatimento para disfarçar a serrilha das linhas)

```
Bitmap.Canvas.AntialiasingMode := amOn;
```

Bom Trabalho!!