**UNIVERSIDADE PAULISTA**

**CIÊNCIA DA COMPUTAÇÃO**

**FRANCIS REGINALDO DE GÓES CAMPOS – RA: A62908-4**

**ATIVIDADES PRÁTICAS SUPERVISIONADAS 4º SEMESTRE**

**SOROCABA**

**2012**

**ATIVIDADES PRÁTICAS SUPERVISIONADAS**

Trabalho acadêmico apresentado como exigência para composição de nota de Atividades Práticas Supervisionadas do curso de Ciência da Computação, da Universidade Paulista.

Orientador: Prof. Fabrício Veloso Ramos.

**SOROCABA**

**2012**

**Sumario**

|  |  |
| --- | --- |
| 1. Objetivo do trabalho.................................................................................. | 4 |
| 2. Introdução................................................................................................. | 5 |
| 3. Regras do Jogo”Campo Minado”.............................................................. | 7 |
| 4. Plano de desenvolvimento do jogo..................................................... | 8 |
| 5. Projeto (estrutura) do programa................................................................ | 9 |
| 6. Relatório com as linhas do programa.......................................................  7. Programa em funcionamento.................................................................... | 10  23 |
| 8. Bibliografia................................................................................................ | 28 |
|  |  |

1. **Objetivo do trabalho**

O trabalho consiste no desenvolvimento do famoso “Campo Minado” ou “Minesweeper”. Ele foi inventado por Robert Donner em 1989, seu objetivo é revelar um campo de minas sem detonar nenhuma. Esse jogo possuí varias versões, e a mais popular é a que vem com o Microsoft Windows.

1. **Introdução**

O popular jogo “Campo Minado” ou “Minesweeper” é um jogo de computador para um jogador que foi enventado em 1989 por Robbert Donner. Seu objetivo é revelar um campo de minas sem que nenhuma seja detonada. A sua popularidade deve ao fato de vir com o sistema operacional da Microsoft Windows.

No andamento do jogo, há padrões de quadrados numerados que permitem somente uma determinada configuração de minas. Para terminar o jogo o mais rápido possível, seria necessário utilizar alguns padrões primeiramente, e deixar os padrões mais difíceis para depois. Existem vários métodos para se ganhar o jogo sem ter que usar de adivinhação.

#### Na análise de um quadrado, é quando vemos que o número de quadrados que ainda não foram descobertos ao redor de um quadrado com um número é igual ao número sendo mostrado, indica que quadrados adjacentes são minas. Porem quando o número de quadrados com minas descobertas ao redor de um quadrado numerado é igual ao número sendo mostrado, qualquer outro quadrado adjacente não possui mina.

#### Na análise de dois quadrados, com dois números de um campo minado, denominados x e y, existem três áreas distintas no campo: minas ao redor de tanto x e y; minas ao redor de x somente e; minas ao redor de y somente. Esse método de análise funciona melhor quando os quadrados relativos a x e y são adjacentes, porem pode ser usado em outras situações. Sabe-se que o número de minas exclusivas a x menos o número de minas exclusivas a y é igual a x − y, o que pode ser usado para marcar bandeiras em minas ou descobrir quadrados seguros.

#### Com análise de minas compartilhadas, suponhamos que um determinado campo possui o número um descoberto. Descobre-se também que dois quadrados adjacentes ao um compartilham uma mina. Isso significa que todos os outros quadrados adjacentes ao um não possuem uma mina, exceto pelos dois que compartilham a mina.

#### A análise final é utilizada no final do jogo, ela pode ser usada para descobrir um quadrado seguro quando todos os outros do campo são ou seguros ou marcados como minas.

Em outro casos é necessário utilizar o método da adivinhação. O jogador pode marcar com um ponto de interrogação onde está localizada a mina. Outro caso que demonstra a necessidade de adivinhação é aquele onde um quadrado não descoberto é cercado por campos indicando o número um, ou quando há uma combinação de minas e o perímetro do campo de jogo. Nesse caso, já que nenhum número rodeia o quadrado não descoberto, o jogador não possui informação sobre o conteúdo do quadrado.

1. **Regras do Jogo “Campo Minado”**

#### A área do jogo é formada por um campo que é constituído por quadrados, no caso do nosso jogo, ele possui cem campos, com dez colunas por dez linhas.

#### Quando clicado em um campo, é feito a verificação se ele contém uma mina ou não. Se possuir uma bomba, então o jogo se acaba, mas se não tiver uma bomba, pode acontecer duas coisas. Uma é aparecer um número indicando quantas casas adjacentes a ele possui uma bomba. Outra coisa é não aparecer nada e revelar os quadrados adjacentes a ele que não possuem nada também.

#### O jogo é ganho quando todos os quadrados que não têm minas são revelados.

#### Caso o jogador acredite que em um determinado campo possui uma bomba, então opcionalmente ele pode clicar com o botão direito do mouse e fica indicado com uma bandeira. Em contrapartida, o jogo é ganho quando todos os campos onde contem bombas são indicados pela bandeira.

1. **Plano de desenvolvimento do jogo**

#### Para desenvolvermos o jogo, utilizaremos o Netbeans IDE 7.0.

#### Este jogo, em sua janela principal, possui um frame, um grid, cento e um botões, dois rótulos e uma barra de menu.

#### O jogo possui uma janela principal, com as dimensões 500x401. Dentro dela há um frame.

#### No frame há um menu com os menus “Sobre” e “Sair”. Possui um rótulo com o texto “Nº de bombas:” e embaixo outro rotulo indicando o número de bombas. Há um botão em que o jogador poderá reiniciar o seu jogo, esse botão possui uma.

#### No frame há também um Grid, onde ficam os botões indicando os campos do jogo. Suas dimensões são 490x300.

#### Ao clicar em “Sobre”, na barra de menu, abrira uma outra janela mostrando informações. Nela há seis rótulos.

#### 

1. **Projeto (estrutura) do programa**

#### No método construtor do projeto, são criados, adicionados e configurados os componentes visuais do projeto.

#### O jogo possui as seguintes funções:

#### CentralizarFrame: é responsável por centralizar o aplicativo no centro da tela;

#### InserirMinas: Distribui as minas de forma aleatoria no campo do jogo;

#### InserirNumeros: Faz a verificação se existe bombas adjacentes ao campo selecionado, retornando o numero de bombas que estão ao redor do campo;

#### Novocampo: apaga todas as informações dos campos e insere as minas novamente;

#### NovocampoII: idêntico ao novocampo, porem apaga e insere novamente os botões(campos);

#### Verifica: verifica se no campo clicado existe uma bomba:

#### Vitória: mostra uma caixa de mensagem informando que o jogador ganhou e o jogo é reiniciado;

#### Derrota: mostra uma caixa de mensagem informando que o jogador perdeu e o jogo é reiniciado.

#### Na classe “main” é instanciada o jogo.

**6. Relatório com as linhas do programa**

package CodFonte;

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

public class CampoMinado extends JFrame implements ActionListener, MouseListener {

int linha = 10;

int coluna = 10;

int numMinas = 10;

GridLayout quadro = new GridLayout(linha, coluna);

private JLabel lblbombas;

private JLabel lblNbombas;

private JButton btnNovo = new JButton();

boolean[] Minas = new boolean[linha \* coluna];

boolean[] clique = new boolean[linha \* coluna];

boolean perdeu = false;

boolean ganhou = false;

int[] vetor = new int[linha \* coluna];

JButton[] botao = new JButton[linha \* coluna];

boolean[] clicado = new boolean[linha \* coluna];

Container ct;

JPanel painel = new JPanel();

public CampoMinado() {

setSize(500, 401);

setTitle(":: Campo Minado ::");

CentralizarFrame(this);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setResizable(false);

ct = getContentPane();

ct.setLayout(null);

painel.setBounds(3, 71, 490, 300);

painel.setLayout(new GridLayout(10, 10, 2, 2));

novocampoII();

for (int i = 0; i < (linha \* coluna); i++) {

painel.add(botao[ i]);

}

JMenuBar menu = new JMenuBar();

menu.setBounds(0, 0, 100, 20);

JMenuItem mnuSobre = new JMenuItem();

JMenuItem mnuSair = new JMenuItem();

mnuSobre.setText("Sobre");

mnuSair.setText("Sair");

menu.add(mnuSobre);

menu.add(mnuSair);

lblbombas = new JLabel(String.valueOf(numMinas));

lblbombas.setFont(new Font("Tahoma", Font.BOLD, 14));

lblbombas.setForeground(Color.red);

lblbombas.setBounds(30, 38, 100, 30);

lblNbombas = new JLabel("N° de bombas:");

lblNbombas.setFont(new Font("Tahoma", Font.BOLD, 10));

lblNbombas.setBounds(3, 20, 100, 30);

btnNovo.setBounds(230, 28, 36, 36);

btnNovo.setFocusable(false);

btnNovo.addActionListener(this);

btnNovo.setIcon(new ImageIcon("o-o.jpg"));

ct.add(menu);

ct.add(lblbombas);

ct.add(lblNbombas);

ct.add(painel);

ct.add(btnNovo);

this.setVisible(true);

mnuSair.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

mnuSairActionPerformed(evt);

}

});

mnuSobre.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

mnuSobre(evt);

}

});

}

private void CentralizarFrame(JFrame frame) {

Dimension paneSize = frame.getSize();

Dimension screenSize = frame.getToolkit().getScreenSize();

frame.setLocation((screenSize.width - paneSize.width) / 2, (screenSize.height - paneSize.height) / 2);

}

private void mnuSobre(java.awt.event.ActionEvent evt) {

JLabel CampoMinado = new JLabel("Campo Minado");

CampoMinado.setFont(new Font("Tahoma", Font.BOLD, 16));

CampoMinado.setBounds(125, 10, 300, 20);

JLabel Desenvolvido = new JLabel("Desenvolvido por:");

Desenvolvido.setFont(new Font("Tahoma", Font.BOLD, 12));

Desenvolvido.setBounds(130, 50, 300, 20);

JLabel Francis = new JLabel("Francis Reginaldo de Góes Campos");

Francis.setFont(new Font("Tahoma", Font.BOLD, 14));

Francis.setBounds(60, 120, 400, 20);

JLabel aps = new JLabel("Atividades Práticas Supervisionadas");

aps.setFont(new Font("Tahoma", Font.BOLD, 12));

aps.setBounds(20, 180, 400, 20);

JLabel cc = new JLabel("Unip - Ciência da Computação");

cc.setBounds(100, 200, 300, 20);

JFrame sobre = new JFrame();

sobre.setBounds(3, 3, 400, 300);

sobre.setLayout(null);

sobre.getContentPane().add(CampoMinado);

sobre.getContentPane().add(Desenvolvido);

sobre.getContentPane().add(Francis);

sobre.getContentPane().add(aps);

sobre.getContentPane().add(cc);

sobre.setTitle(":: Sobre Campo Minado ::");

CentralizarFrame(sobre);

sobre.setVisible(true);

}

private void mnuSairActionPerformed(java.awt.event.ActionEvent evt) {

this.dispose();

}

public void InserirMinas() {

int needed = numMinas;

while (needed > 0) {

int x = (int) Math.floor(Math.random() \* linha);

int y = (int) Math.floor(Math.random() \* coluna);

if (!Minas[ (linha \* y) + x]) {

Minas[ (linha \* y) + x] = true;

needed--;

}

}

}

public void InserirNumeros() {

for (int x = 0; x < linha; x++) {

for (int y = 0; y < coluna; y++) {

int cur = (linha \* y) + x;

if (Minas[ cur]) {

vetor[ cur] = 0;

continue;

}

int temp = 0;

boolean e = (x - 1) >= 0;

boolean d = (x + 1) < linha;

boolean c = (y - 1) >= 0;

boolean b = (y + 1) < coluna;

int esquerda = (linha \* (y)) + (x - 1);

int direita = (linha \* (y)) + (x + 1);

int acima = (linha \* (y - 1)) + (x);

int acimaesquerda = (linha \* (y - 1)) + (x - 1);

int acimadireita = (linha \* (y - 1)) + (x + 1);

int abaixo = (linha \* (y + 1)) + (x);

int abaixoesquerda = (linha \* (y + 1)) + (x - 1);

int abaixodireita = (linha \* (y + 1)) + (x + 1);

if (c) {

if (Minas[ acima]) {

temp++;

}

if (e) {

if (Minas[ acimaesquerda]) {

temp++;

}

}

if (d) {

if (Minas[ acimadireita]) {

temp++;

}

}

}

if (b) {

if (Minas[ abaixo]) {

temp++;

}

if (e) {

if (Minas[ abaixoesquerda]) {

temp++;

}

}

if (d) {

if (Minas[ abaixodireita]) {

temp++;

}

}

}

if (e) {

if (Minas[ esquerda]) {

temp++;

}

}

if (d) {

if (Minas[ direita]) {

temp++;

}

}

vetor[ cur] = temp;

}

}

}

public void novocampoII() {

for (int x = 0; x < linha; x++) {

for (int y = 0; y < coluna; y++) {

Minas[ (linha \* y) + x] = false;

clicado[ (linha \* y) + x] = false;

clique[ (linha \* y) + x] = true;

botao[ (linha \* y) + x] = new JButton( /\*"" + ( x \* y )\*/);

botao[ (linha \* y) + x].setPreferredSize(new Dimension(45, 45));

botao[ (linha \* y) + x].addActionListener(this);

botao[ (linha \* y) + x].addMouseListener(this);

}

}

InserirMinas();

InserirNumeros();

}

public void novocampo() {

for (int x = 0; x < linha; x++) {

for (int y = 0; y < coluna; y++) {

Minas[ (linha \* y) + x] = false;

clicado[ (linha \* y) + x] = false;

clique[ (linha \* y) + x] = true;

botao[ (linha \* y) + x].setEnabled(true);

botao[ (linha \* y) + x].setText("");

}

}

InserirMinas();

InserirNumeros();

for (int x = 0; x < 100; x++) {

botao[x].setIcon(null);

}

perdeu = false;

}

public static void main(String[] args) {

new CampoMinado();

}

public void actionPerformed(ActionEvent e) {

if (!ganhou) {

for (int x = 0; x < linha; x++) {

for (int y = 0; y < coluna; y++) {

if (e.getSource() == botao[ (linha \* y) + x] && !ganhou && clique[ (linha \* y) + x]) {

doCheck(x, y);

break;

}

}

}

}

if (e.getSource() == btnNovo) {

novocampo();

ganhou = false;

return;

}

Verifica();

}

public void mouseEntered(MouseEvent e) {

}

public void mouseExited(MouseEvent e) {

}

public void mousePressed(MouseEvent e) {

if (e.getButton() == 3) {

int n = 0;

for (int x = 0; x < linha; x++) {

for (int y = 0; y < coluna; y++) {

if (e.getSource() == botao[ (linha \* y) + x]) {

clique[ (linha \* y) + x] = !clique[ (linha \* y) + x];

}

if (!clicado[ (linha \* y) + x]) {

if (!clique[ (linha \* y) + x]) {

botao[(linha \* y) + x].setIcon(new ImageIcon("band.gif"));

n++;

} else {

botao[ (linha \* y) + x].setText("");

botao[(linha \* y) + x].setIcon(new ImageIcon(""));

}

}

}

}

}

}

public void mouseReleased(MouseEvent e) {

}

public void mouseClicked(MouseEvent e) {

}

public void doCheck(int x, int y) {

int cur = (linha \* y) + x;

boolean l = (x - 1) >= 0;

boolean r = (x + 1) < linha;

boolean u = (y - 1) >= 0;

boolean d = (y + 1) < coluna;

int esquerda = (linha \* (y)) + (x - 1);

int direita = (linha \* (y)) + (x + 1);

int acima = (linha \* (y - 1)) + (x);

int acimaesquerda = (linha \* (y - 1)) + (x - 1);

int acimadireita = (linha \* (y - 1)) + (x + 1);

int abaixo = (linha \* (y + 1)) + (x);

int abaixoesquerda = (linha \* (y + 1)) + (x - 1);

int abaixodireita = (linha \* (y + 1)) + (x + 1);

clicado[ cur] = true;

botao[ cur].setEnabled(false);

if (vetor[ cur] == 0 && !Minas[ cur] && !perdeu && !ganhou) {

if (u && !ganhou) {

if (!clicado[ acima] && !Minas[ acima]) {

clicado[ acima] = true;

botao[ acima].doClick();

}

if (l && !ganhou) {

if (!clicado[ acimaesquerda] && vetor[ acimaesquerda] != 0 && !Minas[ acimaesquerda]) {

clicado[ acimaesquerda] = true;

botao[ acimaesquerda].doClick();

}

}

if (r && !ganhou) {

if (!clicado[ acimadireita] && vetor[ acimadireita] != 0 && !Minas[ acimadireita]) {

clicado[ acimadireita] = true;

botao[ acimadireita].doClick();

}

}

}

if (d && !ganhou) {

if (!clicado[ abaixo] && !Minas[ abaixo]) {

clicado[ abaixo] = true;

botao[ abaixo].doClick();

}

if (l && !ganhou) {

if (!clicado[ abaixoesquerda] && vetor[ abaixoesquerda] != 0 && !Minas[ abaixoesquerda]) {

clicado[ abaixoesquerda] = true;

botao[ abaixoesquerda].doClick();

}

}

if (r && !ganhou) {

if (!clicado[ abaixodireita] && vetor[ abaixodireita] != 0 && !Minas[ abaixodireita]) {

clicado[ abaixodireita] = true;

botao[ abaixodireita].doClick();

}

}

}

if (l && !ganhou) {

if (!clicado[ esquerda] && !Minas[ esquerda]) {

clicado[ esquerda] = true;

botao[ esquerda].doClick();

}

}

if (r && !ganhou) {

if (!clicado[ direita] && !Minas[ direita]) {

clicado[ direita] = true;

botao[ direita].doClick();

}

}

} else {

botao[ cur].setText("" + vetor[ cur]);

if (!Minas[ cur] && vetor[ cur] == 0) {

botao[ cur].setText("");

}

}

if (Minas[ cur] && !ganhou) {

//botao[ cur].setText("0");

botao[cur].setText(null);

botao[cur].setIcon(new ImageIcon("bomba.gif"));

Derrota();

}

}

public void Verifica() {

for (int x = 0; x < linha; x++) {

for (int y = 0; y < coluna; y++) {

int cur = (linha \* y) + x;

if (!clicado[ cur]) {

if (Minas[ cur]) {

continue;

} else {

return;

}

}

}

}

Vitoria();

}

public void Vitoria() {

if (!perdeu && !ganhou) {

ganhou = true;

JOptionPane.showMessageDialog(null, "Você ganhou!\nInicie outro jogo", "::Você ganhou :D ::", JOptionPane.INFORMATION\_MESSAGE);

novocampo();

ganhou = false;

return;

}

}

public void Derrota() {

if (!perdeu && !ganhou) {

perdeu = true;

for (int i = 0; i < linha \* coluna; i++) {

if (!clicado[ i]) {

botao[ i].doClick(0);

}

}

JOptionPane.showMessageDialog(null, "Você perdeu!\nInicie outro jogo", ":: Você Perdeu :( ::", JOptionPane.ERROR\_MESSAGE);

novocampo();

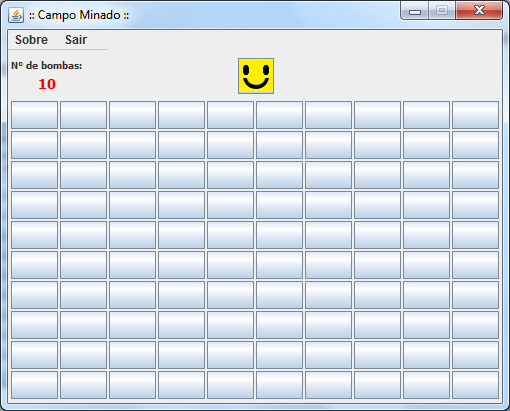
}

}

}

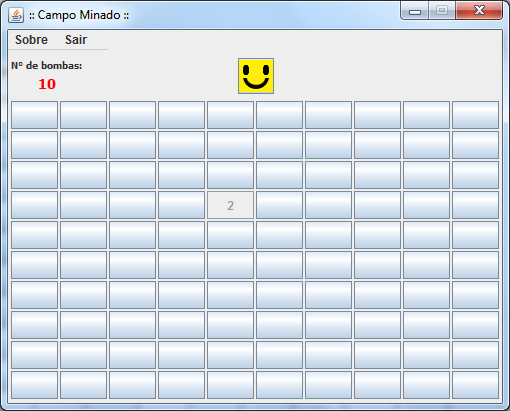
**7. Programa em funcionamento**

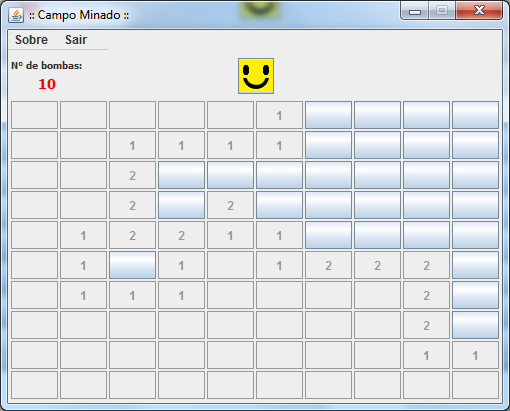
Ao iniciar o jogo aparece a tela como mostra a figura1, inicialmente mostrando no rotulo número de bombas e os campos ainda não revelados.

  
figura1

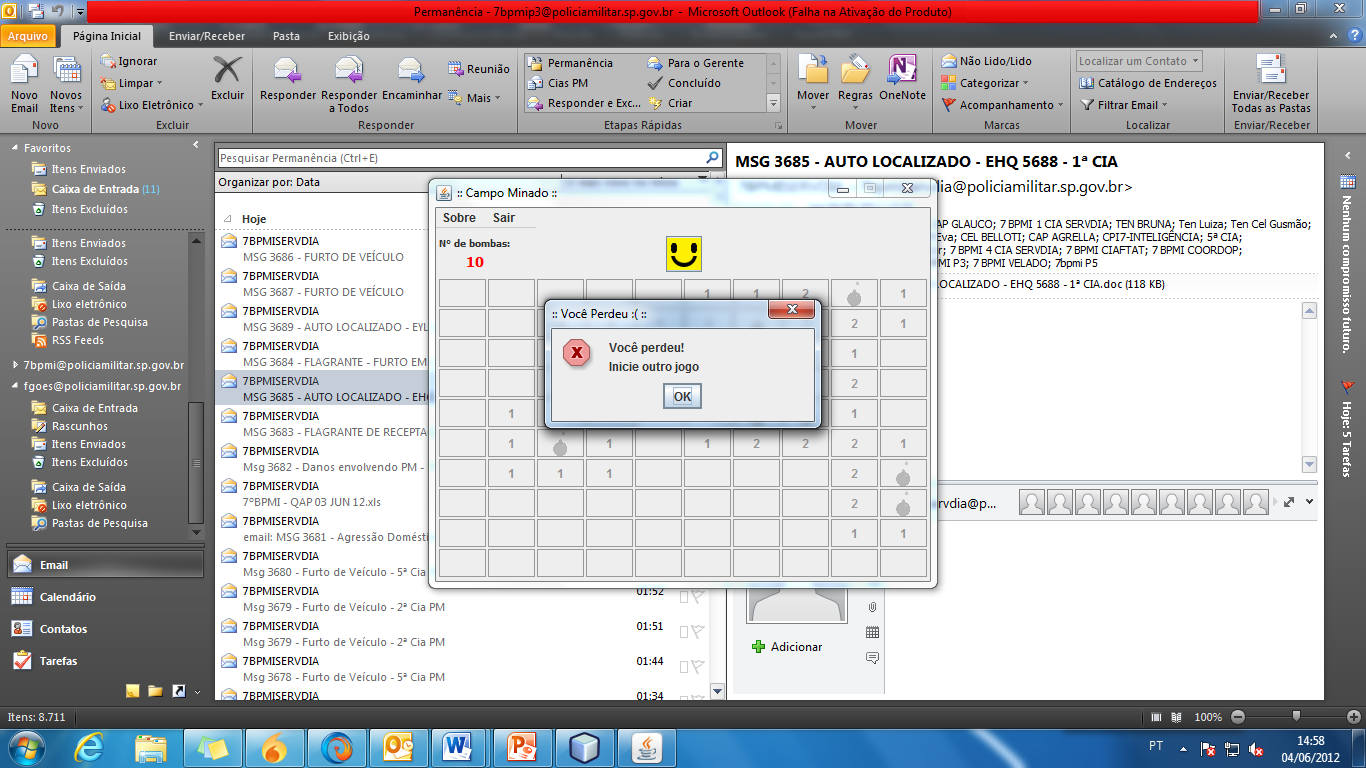
Foi em um determinado campo e ele revelou que não possui uma bomba, porem possui dois campos adjacentes que possuem uma bomba e no campo que mostra o número de campos restantes indica o número noventa e nove, como mostra a figura2.

Em seguida foi clicado no campo da primeira linha com a primeira coluna, e revelou uma serie de casas adjacentes que não possuem uma bomba, ou casas adjacentes a uma ou mais bombas, mostrada na figura3.

  
figura2

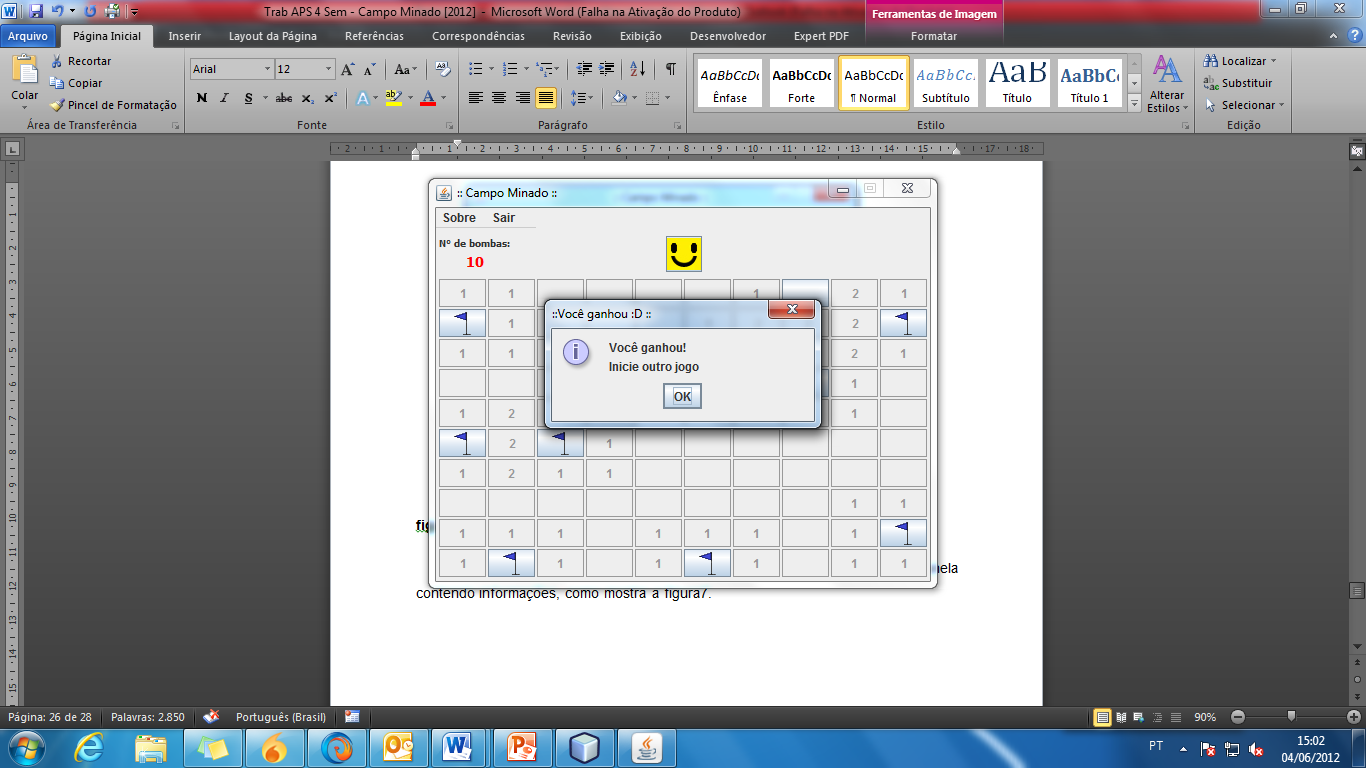
  
figura3

Quando é clicado em um campo que possui uma bomba, o campo clicado mostra a imagem de uma bomba, aparece uma caixa com a seguinte mensagem: “Você perdeu! Inicie outro jogo”, como é mostrado na figura4.

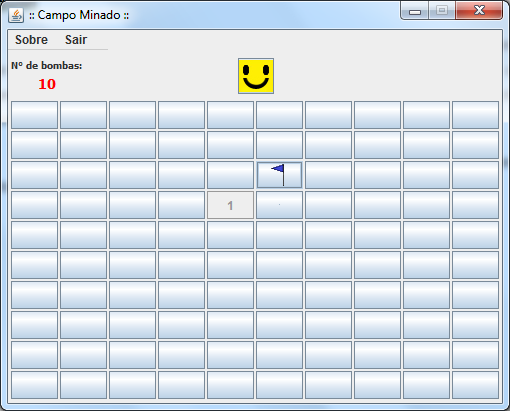
  
figura4

Após clicar em “OK”, a caixa de mensagem se fecha e é iniciado um novo jogo.

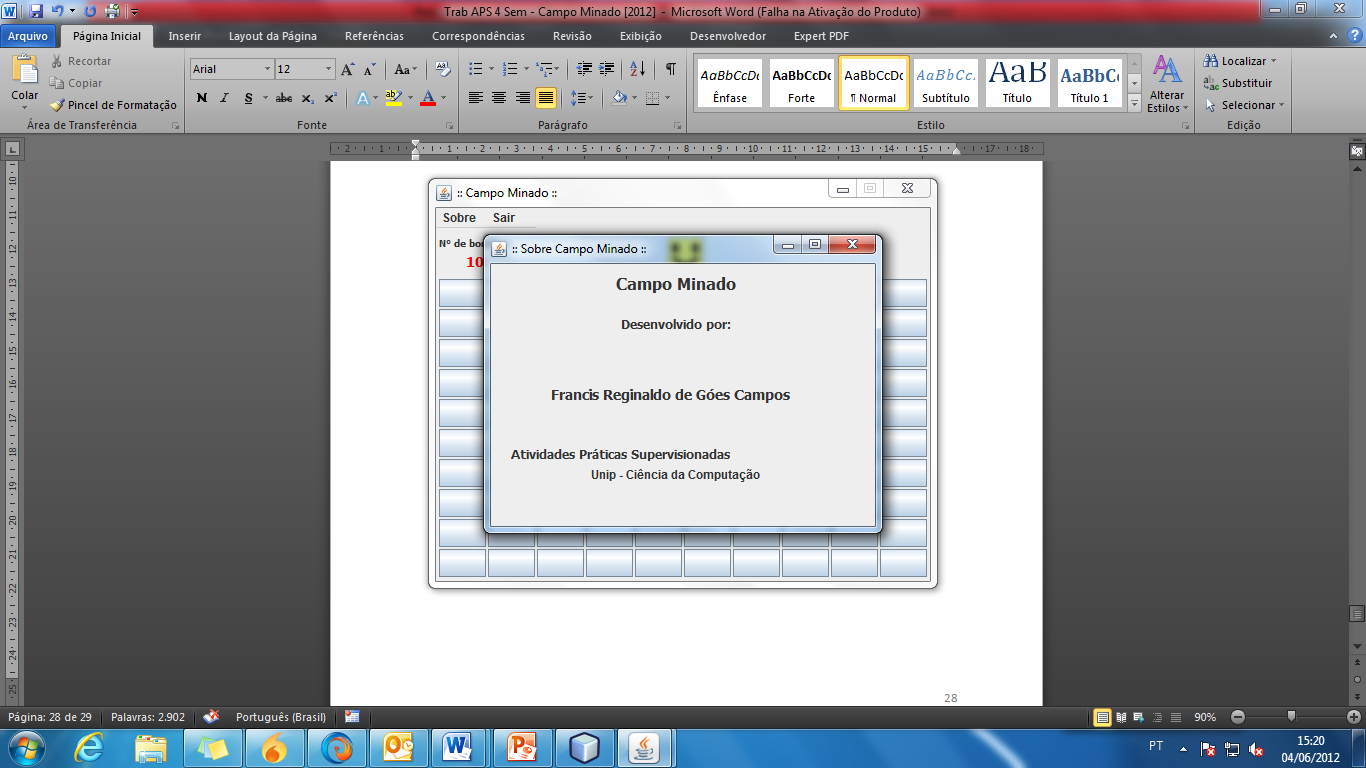
E após descobrir todos os campos que não possuem bomba, e então restar apenas dez campos, o jogo é ganho, e aparece uma caixa de mensagem dizendo: “Você ganhou! Inicie outro jogo”, e ao clicar em ok é iniciado um novo jogo. Como mostra na figura5.

 **figura5**

Caso o jogador suspeite de que um determinado campo possui uma bomba, ele poderá clicar com o botão direito do mouse em cima do campo e ficara uma bandeira, indicando que ali há uma bomba (figura6). Para desmarcar basta clicar novamente com o botão direito do mouse em cima do campo.

 **figura6**

E ao clicar no botão “Sobre” na barra de menu, abre uma janela contendo informações, como mostra a figura7.

 **figura7**

**8. Bibliografia**

**Windows** – Como Jogar Campo Minado (<http://windows.microsoft.com/pt-BR/windows7/Minesweeper-how-to-play>)

**Wikipédia** – Campo Minado (<http://pt.wikipedia.org/wiki/Campo_minado>)

**Nova Web Cheats** – Tutorial Campo Minado (<http://www.webcheats.com.br/forum/info-zone-lixao/55064-tutorial-jogando-campo-minado.html>)

**GUJ** – Campo Minado (<http://www.guj.com.br/java/257546-jogo-campo-minado>)

**GUJ** - Configuração NetBeans JFrame (<http://www.guj.com.br/java/90528-configuracao-netbeans-jframe>)

**iMasters** – Dicas rápidas jGrid (<http://imasters.com.br/artigo/22309/desenvolvimento/dicas-rapidas-de-jqgrid>)

**Java Free** – Duvida com ActionEvent (<http://javafree.uol.com.br/topic-881171-Duvida-com-ActionEvent.html>)

**Java Free** – Posicionamento por setBounds(int, int, int, int) (<http://javafree.uol.com.br/topic-5754-Posicionamento-por-setBoundsintintintint.html>)

**Java Free** – If com operadores lógicos (<http://javafree.uol.com.br/topic-862779-IF-com-operadores-logicos-AND-and-OR.html>)

**Forum PCs** – Algoritmo em português estruturado (<http://www.forumpcs.com.br/comunidade/viewtopic.php?t=168806>)

**About.com Java -** Add a JLabel to the JFrame (<http://java.about.com/od/creatinguserinterfaces/ss/simplewindow_4.htm>)