UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE CIÊNCIAS EXATAS E DA TERRA
DEPARTAMENTO DE INFORMÁTICA E MATEMÁTICA APLICADA DIM0320 - ALGORITMO E PROGRAMAÇÃO DE COMPUTADORES

Flawbert Lorran da Silva Costa (20230031776)
Murilo Antonio Lima da Costa (20230048906)

Projeto - Jogo de Campo Minado na linguagem de programação C

Natal/RN

Junho - 2024

Flawbert Lorran da Silva Costa (20230031776)
Murilo Antonio Lima da Costa (20230048906)

Projeto - Jogo de Campo Minado na linguagem de programação C

Relatório apresentado à Universidade Federal do Rio Grande do Norte, na disciplina de Algoritmo e Programação de Computadores, como requisito para avaliação na 3° unidade, solicitado pelo professor Lucas Rodrigues da Silva.

Natal/RN

Junho - 2024

SUMÁRIO

SUMÁRIO..... 3
1. Introdução..... 4
2. Planejamento do Projeto..... 5
3. Descrição do Jogo..... 7
3.2 Especificação dos controles do jogo..... 7
3.2.1 Regras..... 7
3.2.2 Fluxos..... 7
4. Funcionalidades não concluídas..... 9
5. Cronograma das atividades realizadas..... 10
6. Conclusão..... 11

1. Introdução

Esse relatório consiste em descrever o desenvolvimento de um jogo de campo minado desenvolvido na linguagem de programação C, utilizando os conhecimentos adquiridos durante a disciplina de Algoritmo e Programação de Computadores, tais como laços de repetição, funções, condicionais, matrizes e vetores.

O campo minado é um jogo que desafia o usuário a revelar cédulas de um tabuleiro sem explodir nenhuma mina. A dinâmica inclui habilidades de dedução, lógica e planejamento de riscos.

Nesse relatório abordaremos o planejamento do projetor, a descrição e as regras do jogo, fluxos, funcionalidades não concluídas e cronograma das atividades realizadas pelos componentes e desenvolvedores do projeto.

Sendo assim, esse relatório visa não apenas documentar o projeto final, mas apresentar as etapas de desenvolvimento que foram realizadas para que se chegasse a uma conclusão.

2. Planejamento do Projeto

Para o desenvolvimento do jogo, o grupo pretende procurar sobre as regras do campo minado, tais como: como se deve jogar, a quantidade de minas por tamanho de tabuleiro, o que são considerados níveis fácil, médio e difícil. Os componentes jogaram para se familiarizar com as técnicas e lógica do jogo.

Como já existem projetos de jogo de campo minado, o grupo procurou soluções similares para desenvolver e definir quais funções e técnicas de programação serão utilizadas para desenvolver uma solução eficaz e de fácil entendimento tanto para os programadores quanto para o avaliador do projeto.

Para o funcionamento do jogo, serão utilizadas as seguintes funções:

Para a impressão do campo minado na tela para o usuário, além do menu e instruções para jogar, serão utilizadas as seguintes funções:

- **inicializaCampo**: A função percorre cada célula da matriz e a preenche de acordo com a célula
- **inicializaMenu**: Mostra o menu que permite o usuário a escolher a dificuldade do jogo, mostrar a sua pontuação ou instruções ou sair do campo minado.
- **printCampo**: Imprime uma matriz por meio da estrutura de repetição for, que é o tabuleiro do campo minado.
- printaBomba: Imprime o desenho de uma bomba.
- **mostraInstrucoes**: Explica ao usuário quais são as instruções do jogo campo minado.

Na parte do jogo, serão utilizadas as seguintes funções para que o funcionamento seja correto e o usuário possa jogar de acordo com as regras:

- **inicializaBombas**: Essa função coloca ‘n’ bombas em posições aleatórias na matriz ‘campo’ e garante que não haja mais de uma bomba na mesma posição.
- **sensorBombas**: A função analisa a matriz ‘campo’ para detectar a presença de bombas e atualiza a outra matriz ‘campoClone’ com o número de bombas adjacentes a cada célula.
- **verificaCampo**: A função mostra células em uma matriz de jogo que começa nas coordenadas x e y. Ela revela células próximas que não contém bombas. Quando encontra uma célula com o número '0', ela chama a si mesma repetidamente para revelar as células próximas a essa célula com o número '0', criando o “efeito cascata” que ocorre no jogo campo minado.
- **flagPosition**: A função posiciona uma bandeira (‘flag’) dentro da matriz em formato de caractere ‘>’, sinalizando poder haver uma bomba no local.
- **flagSuggestion**: Essa função faz o print da sugestão do usuário inserir uma bandeira ou não, colhendo a informação do usuário. Caso o usuário queira, a função flagPosition() é chamada, caso não, seguirá o jogo.
- **verificaVitoria**: Faz a verificação e comparação do número de campos livres com os ocupados, pois caso bata com a quantidade de bombas da dificuldade, ela retorna o número 1, ou 0 se não for o caso.

- **printfVitoria**: Se a função verificaVitoria() retornar 1, ela é chamada.
- **jogaJogo**: A função controla o fluxo do jogo, solicitando que o jogador revele células no campo. Ela atualiza o campo de jogo de acordo se o jogador encontrar uma bomba, sair do jogo ou fazer uma escolha errada. O jogo termina quando o jogador decide sair livremente ou uma bomba explode.

3. Descrição do Jogo

3.1 Objetivo

O jogador deve digitar a coordenada de quadrados e revelar seu conteúdo. Dentro dos quadrados podem haver bombas ou não. O jogador deve usar dedução e lógica para descobrir quais quadrados existem bombas ou não. O objetivo final do jogo é conseguir descobrir todos os quadrados do campo minado sem colocar a localização em nenhuma bomba. Caso o jogador consiga, ele vence.

3.2 Especificação dos controles do jogo

O usuário deve primeiro selecionar um quadrado, utilizando as suas coordenadas (a linha e a coluna) que aquele quadrado está localizado, digitando tais coordenadas pelo teclado. Caso o jogador tenha colocado a coordenada de um quadrado sem bombas, será revelada a quantidade de bombas em quadrados adjacentes ao selecionado.

3.2.1 Regras

O jogador vence quando revela todos os quadrados sem minas. Ao clicar em um quadrado com minas, ele irá perder a partida.

3.2.2 Fluxos

Ao se iniciar o programa, é apresentada uma interface no terminal, à espera da interação do usuário.

O usuário escolhe o grau de dificuldade do programa, alternando entre fácil, médio, difícil e impossível. Sendo cada um com suas respectivas características:

- Fácil: 10 bombas / 71 campos livres;
- Médio: 26 bombas / 55 campos livres;
- Difícil: 57 bombas / 24 campos livres;
- Impossível: 80 bombas / 1 campo livre;

Além disso, o usuário pode consultar as regras do jogo como também pode sair do jogo / encerrar o programa.

Então, ao usuário escolher sua dificuldade, é inicializado um campo em forma de matriz 10x10, onde sua primeira linha e primeira coluna representam os números das coordenadas X e Y do campo. Em seguida, na submatriz 9x9 que resta, ou seja, 81 elementos, é chamada uma função para dispor bombas aleatoriamente nelas. E seguindo isso, é inicializada outra matriz, tal qual uma “matriz gabarito”, onde é realizada uma varredura em todos os campos diferentes de bombas para enumerar a quantidade de bombas ao redor, sendo esse número indo de 1 a 8. Após isso, finalmente, é dado início a função jogaJogo(), com uma matriz 10x10 limpa, que será disposta ao usuário, para ele não ver o resultado, e a matriz gabarito, para ter a base de como irá ocorrer o jogo, como parâmetro. O jogo conta os pontos que o usuário fez ao selecionar um campo sem explodir nenhuma bomba. Também há a disposição

do usuário 10 bandeiras para serem colocadas em possíveis campos que contém bombas.

A função jogaJogo() dispõe uma interface para usuário no terminal, solicitando ao usuário as coordenadas do campo escolhido pelo mesmo.

Caso o usuário escolha um campo contendo uma bomba, ele perde e retorna ao menu inicial. Caso não seja uma bomba, e sim um campo livre, vai ser chamada uma função que vai varrer uma matriz 3x3, sendo o elemento escolhido o elemento central, ou seja, o elemento[1][1] da matriz, para dispor o número de bombas ao redor do campo, e então, sendo disponibilizada para o usuário uma nova matriz na tela, com o resultado dessa varredura do campo escolhido pelo usuário, segundo jogo. E por fim, caso o usuário digite x: 0 e y: 0, o usuário retorna a interface inicial.

4. Funcionalidades não concluídas

O grupo desenvolveu e implementou todas as funcionalidades necessárias para se jogar uma partida de campo minado clássica, como: posicionamento de bandeiras, contagem de pontos e indicação de campos vazios ao redor da localização escolhida pelo usuário.

5. Cronograma das atividades realizadas

Data	Autor	Descrição da atividade
28/05	Murilo Costa	Início do desenvolvimento do projeto.
30/05	Flawbert Costa e Murilo Costa	Determinação das funções disponíveis no jogo e o que iria ser igual ou diferente do clássico jogo do campo minado.
31/05	Murilo Costa	Implementação das idéias e desenvolvimento do projeto.
13/06	Murilo Costa	Vistoria da primeira etapa do projeto, e edição do primeiro relatório.
17/06	Flawbert Costa	Modificações no relatório e adição de informações em alguns tópicos.
18/06	Flawbert Costa	Revisão do código e do relatório para conferir se estão equivalentes quanto à informações e funcionalidades.
18/06	Murilo Costa	Atualização do código, implementando novas funções que faltavam.
18/06	Murilo Costa	Atualização do relatório com base na atualização do programa
21/06	Flawbert Costa	Revisão e finalização do relatório

6. Conclusão

- **O grau de dificuldade do projeto:** Em uma escala de 1 a 10, os desenvolvedores avaliaram em nota 5 quanto à dificuldade do projeto.
- **A importância dos conhecimentos obtidos:** O conhecimento mais importante a ser adquirido é o trabalho em grupo, próxima uma experiência profissional, contando com revisão de código, produção de relatório, discussão acerca de desempenho e funcionalidades, além da aplicação do conhecimento obtido em sala de aula durante todo o semestre utilizando linguagem C e a maioria das ferramentas que a linguagem tem a disposição para atender os requerimentos do programador.
- **As vantagens e desvantagens observadas em relação a uma prova convencional:** A percepção do grupo, não há desvantagens, pois apresenta um grau de familiaridade maior com uma experiência profissional do que uma prova que apenas constata o conhecimento sobre a matéria ou o que foi abordado, mas não coloca essa constatação a prova, como um projeto real, com um intuito funcional, com um tempo maior para o desenvolvimento e para pensar em diferentes formas de resolver o problema que foi apresentado.
- **Acompanhamento do professor:** Não foi necessário.