Universidade Federal do Rio Grande do Norte
Departamento de Informática e Matemática Aplicada

João Victor Alves
Paulo Victor das Flores Cabral de Medeiros
Maria Clara da Silva
Rafael De Melo Santos Leite

Relatório de produção do jogo Campo Minado

Natal – RN
Novembro – 2023

T03 / G4
João Victor Alves – 20200132615
Paulo Victor das Flores Cabral de Medeiros – 20180039297
Rafael De Melo Santos Leite - 20220055818
Maria Clara da Silva - 20230032030

Relatório de produção do jogo Campo Minado

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 Umberto Souza da Costa.

Natal – RN
Novembro - 2023

SUMÁRIO

1. Introdução

O desenvolvimento deste trabalho está sendo conduzido pelo grupo visando concluir a disciplina de Algoritmos e Programação. Este documento visa relatar a implementação do jogo Campo Minado em linguagem C, proporcionando uma visão detalhada da jornada do grupo durante o desenvolvimento do algoritmo.

2. Planejamento do Projeto

Para o planejamento deste projeto, realizamos inicialmente uma reunião para compreender o funcionamento do jogo e sua lógica, onde separamos os principais tópicos para a implementação, a fim de atribuir responsabilidades a cada integrante em um primeiro momento. Após a listagem dos principais tópicos, ficou acordado o que desenvolveria cada um. A divisão pode ser visualizada na tabela abaixo.

INTEGRANTE	FUNÇÃO
Paulo Victor	Calcular números de pistas.
João Victor	Tratar eventos e atualizar tabuleiro.
Maria Clara	Definir a estrutura dos tabuleiros e menu.
Rafael de Melo	Gerar minas aleatórias.

Vale ressaltar que, apesar das distribuições de funções, todos os integrantes participaram da implementação do código em geral, utilizando reuniões no Meet, conversas no WhatsApp e horários de aula para a discussão e explicação dos códigos elaborados. O intuito deste relatório é registrar as estratégias escolhidas pelo grupo, assim como explicar de maneira concisa as principais funções presentes no algoritmo. Este pode ser dividido da seguinte forma:

2.1 Struct.

A construção do algoritmo iniciou com a criação de uma estrutura (struct) denominada "Celula", destinada a armazenar quatro tipos de informações. Essas informações foram definidas como variáveis do tipo inteiro, cada uma desempenhando funções específicas: "temBomba" (responsável por verificar se uma célula contém uma bomba), "aberto" (responsável por rastrear as células abertas no jogo), "vizinhos" (responsável por verificar as células ao redor da célula selecionada) e "flag" (marca uma célula que contém uma bomba).

2.2 Matriz.

Em seguida, criamos uma matriz bidimensional chamada "jogo" como uma variável global, com um tamanho predefinido de 20x20, visando contemplar diferentes dimensões de tabuleiro. Para auxiliar a implementação, declaramos uma variável do tipo inteiro, chamada "tam", e a inicializamos com o valor 8, representando o tamanho mínimo permitido para o tabuleiro.

2.3 Void inicioJogo.

A função “inicioJogo” é a responsável por inicializar o tabuleiro, e traz consigo os parâmetros temBomba, aberto, vizinhos e flag. Como não ocorreram interações no jogo, todas as variáveis retornam 0.

2.4 Void sortearBombas.

A função “sortearBombas” vai utilizar da função srand(time(NULL)) para sortear bombas de forma aleatória, utilizando loops e condicionais para analisar cada célula e verificar se ela já possui uma bomba ou não.

2.5 Validação Coordenadas.

A função “Validação Coordenadas” foi feita para retornar verdadeiro (1) se as coordenadas estiverem dentro dos limites validos, e falso (0) caso ocorra o contrário. A condicional estabelecida garantirá que as variáveis "linha" e "coluna" sejam maiores ou iguais a 0, ao mesmo tempo, em que ambas são menores que a variável "tam".

2.6 Void bombasVizinhas.

A função Bombas Vizinhas vai verificar se as coordenadas são validas e se elas possuem bombas, essa análise é feita em oito posições diferentes, como demonstrado na figura a seguir. Porém, se as duas condições forem atendidas, a variável “quantidade” será incrementada com o valor de 1, e impressa ao final do laço.

2.7 Void contarBombas.

A função "Contar Bombas" percorre todas as células do tabuleiro. Para cada célula, ela calcula a quantidade de bombas vizinhas usando a função "Bombas Vizinhas" e armazena esse valor na variável "vizinhos", que está contida na estrutura "Celula".

2.8 Void inicializarTabuleiro.

A função "Inicializar Tabuleiro" é a encarregada por inicializar o jogo, ela chama as funções: inicioJogo, sortearBombas, contarBombas para preparar o tabuleiro antes da primeira interação com o usuário. Ao chamar essas as funções, o void inicializarTabuleiro posiciona bombas aleatoriamente, conta as bombas vizinhas e calcula as posições necessárias para vencer.

2.9 Void impressaoTabuleiro.

A função "Impressão Tabuleiro" imprimirá a parte visual de cada tabuleiro, mostrando as letras do alfabeto como índices das colunas e números como índices das linhas. Ela ficará responsável por apresentar visualmente algumas das ações que o usuário escolher, como os caracteres de bomba, espaços vazios e bandeiras.

2.10 exibirMenu.

A função "Exibir Menu" inclui informações essenciais que o usuário precisa conhecer antes de iniciar o jogo, sendo sempre apresentada no início do código. No menu principal, o usuário seleciona o nível de dificuldade desejado por meio de uma condicional switch case. Após a escolha, o algoritmo imprime o tabuleiro correspondente.

2.11 Void abrirCelula.

A função "Abrir Célula" é acionada quando o usuário escolhe a opção de abrir uma célula no tabuleiro. Se a célula não contiver uma bomba, será marcada como aberta; no entanto, se conter uma bomba, a função "perdeuJogo" será chamada para encerrar a partida. Contudo, se o número de posições abertas for igual ao número necessário para vencer, a função "ganhouJogo" será acionada.

2.12 coordColuna.

A função "Coordenada Coluna" é responsável por converter um caractere que representa a coluna em um número inteiro, utilizando a tabela ASCII para realizar essa conversão.

2.13 Void marcarBandeira.

A função "Marcar Bandeira" visa pegar uma posição escolhida pelo usuário e marcar ela com uma bandeira, que seria um artifício presente na lógica do jogo original para solucionar o tabuleiro.

2.14 Void desmarcarBandeira.

A função "Desmarcar Bandeira" funciona de maneira contrária a anterior, dando uma opção ao usuário de desmarcar um espaço com uma bandeira previamente colocada no tabuleiro.

2.15 Void perdeuJogo.

A função "Perdeu Jogo" é chamada quando o usuário abre uma posição com bomba, assim imprimindo uma mensagem de derrota e atualizando as variáveis “derrotas” e “jogo ativo”.

2.16 Void ganhouJogo.

A função "Ganhou Jogo" é utilizada quando o usuário resolve o tabuleiro e vence o jogo, ela atualiza as variáveis “vitorias” e “jogo ativo”.

2.17 Main.

A função "Main" está sendo utilizada para gerenciar a execução do jogo, possuindo funções internas chamadas conforme as necessidades funcionais ou escolhas do usuário. Como, por exemplo, a opção de jogar novamente e/ou a impressão dos dados estatísticos de derrotas e vitorias.

3. Descrição do Jogo

3.1 Objetivo:

O campo minado é um jogo de estratégia baseado em coordenadas. O objetivo é conseguir localizar onde estão as bombas baseando-se na contagem de espaços vizinhos. No jogo podemos marcar as posições em que estão as bombas e para ganhar temos que conseguir abrir todas as coordenadas sem abrir nenhuma bomba.

3.2 Especificação dos controles do jogo:

Para o jogo é necessário digitar primeiramente a fase que iremos jogar, sendo essas “Muito Fácil,” Fácil ","Médio" e"Difícil ". Após digitar o número correspondente ao nível, digitamos a coordenada, informando o número correspondente a ação. As ações separadas são: “Abrir Célula”, “Marcar Posição com Bandeira”, “Desmarcar posição com bandeira” e “Voltar”.

Cada carácter definido no jogo tem uma função. Os “*” correspondem às bombas, “#” aos espaços vazios e “!” corresponde às bandeiras.

Após selecionar o posicionamento, poderemos abrir aquela célula, marcar uma bandeira, desmarcar uma bandeira ou voltar.

3.2.1 Regras:

A principal regra e condição de derrota do campo minado é clicar em uma coordenada que tenha uma bomba. O objetivo e condição de vitória dar-se pela abertura de todas as coordenadas que não tenham bomba.

Os números que aparecem ao abrir células estão relacionados ao quanto de bombas que tem ao redor daquela coordenada, isto é, em cima, em baixo, na esquerda, direita, e os cantos diagonais referentes àquela posição.

Ao clicar em um local que tenha bomba, o jogo irá parar e aparecer a mensagem de derrota, e perguntar se quer jogar novamente.

Caso sejam abertos todas as coordenadas sem bombas, ou consiga marcar todas as coordenadas com bandeira, o jogo finaliza também, aparecer as parabenizações e contabiliza uma vitória.

3.2.2 Fluxos:

O jogo segue um fluxo simples e intuitivo. A primeiro momento, vai ser solicitado qual nível o usuário quer, depois disso, já podemos visualizar o tabuleiro. No segundo passo, temos que escolher a coordenada que iremos executar uma ação, sendo essa abrir célula, colocar bandeira ou retirar uma bandeira (no jogo também foi implementado a opção de voltar, caso o usuário queira inserir outra coordenada). O jogo só finaliza caso o usuário abra uma posição que tenha bomba, coloque bandeiras em todas as bombas, ou visualize todas as células que não tem bomba.

Se o usuário digitar uma posição que tem bomba, vai aparecer a mensagem correspondente à derrota. Nesta mensagem terão as opções de jogar novamente. Caso o usuário digite para jogar novamente, será feito novamente o começo do fluxo. Se o usuário digitar para sair, vai aparecer a mensagem de finalização com os créditos dos criadores do jogo.

Se o conseguir obedecer às condições de vitória, isto é, consiga marcar todas as bombas com bandeiras ou abrir todas as células que não tem bomba, ele será direcionado para uma tela de vitória, em que poderá visualizar a opção de jogar novamente ou sair, sendo essa parte, especificamente, semelhante ao fluxo final de

derrota também, em que o usuário pode voltar para o começo do fluxo ou finalizar, aparecendo assim os créditos finais.

4. Funcionalidades não concluídas

A funcionalidade de visualização automática dos campos vazios de uma determinada região:

- Foi gasto muito tempo na implementação das funções responsáveis pela atualização do tabuleiro conforme as ações do usuário
- A complexidade do código

A implementação de um cronômetro:

- Gerenciamento de tempo ao projeto
- Ordem de prioridade, por se tratar de uma funcionalidade opcional foi decidido focar na parte principal do jogo.

A interface gráfica do jogo:

- Gerenciamento de tempo ao projeto
- Ordem de prioridade
- Dificuldades na implementação de bibliotecas

5. Cronograma das atividades realizadas

Data	Autor	Descrição da atividade
18/11	Todos	Reunião do Grupo via Discord para organização do projeto.
23/11	Maria	Implementação de código que gera o tabuleiro.
28/11	Maria	Implementação do menu do jogo e impressão dos tabuleiros.
28/11	Paulo	Implementação do código para analisar bombas vizinhas.
28/11	Todos	Reunião para testes de código, dúvidas e sugestões para o projeto.
30/11	Rafael	Implementação do código referente a geração das bombas.
30/11	Todos	Discussão em horário de aula sobre o código e tira-dúvidas com o Professor.
02/12	Todos	Reunião Assíncrono para organização do projeto.
03/12	Maria	Modificação no menu e adição de legendas.
05/12	Todos	Discussão em horário de aula sobre o código e tira-dúvidas com o Professor.
06/12	João	Implementação do código referente às ações do usuário (input) e reorganização das funções antigas com as novas.
08/12	Todos	Reunião para testes de código e dúvidas.
10/12	João	Finalização do código referente às ações do usuário (input).
11/12	Todos	Testagem do jogo.

6. Conclusão

Ao final do projeto, concluímos, que foi um trabalho complexo, mas também muito interessante e interativo em relação a uma prova convencional. E com a ajuda de todos do grupo, deixamos o projeto mais leve e dinâmico, sem sobrecarregar nenhum componente.

O projeto tem um grau de dificuldade avançado, mas com o acompanhamento do professor e os conhecimentos obtidos durante as aulas foi possível finalizar com excelência o trabalho.

O professor se manteve sempre muito prestativo em relação a todas as dúvidas que surgiram ao longo do projeto e nos ajudou da melhor maneira possível mostrando o caminho mais fácil ou nos mostrando os ajustes necessários a fazer.

Por fim, o grupo se mostrou muito empolgado e dedicado para realização do projeto e a partir disso conseguimos finalizar o trabalho de maneira positiva e sempre nos ajudando a todo o momento.