**Introdução**

Este trabalho de Introdução a Programação tem como objetivo implementar um jogo de batalha naval em Java, com interface via console. O programa simula uma partida entre dois jogadores, usando um tabuleiro para ataque e defesa. O usuário interage com o sistema através de um menu, onde pode iniciar partidas, posicionar embarcações, atacar o oponente e verificar o resultado.

**Implementação**

A implementação foi feita usando Java, com foco em aplicação de conceitos básicos, como vetores bidimensionais, estruturas de repetição, matriz, condicionais e modularização do código usando métodos. A interface e baseada em um console, com uso da Classe Scanner para entrada de dados e método impressão das matrizes.

**Estrutura de dados**

- Quatro matrizes 9x9 do tipo String são usadas: duas para registrar os ataques (ataque1 e ataque2) e duas para registrar as defesas (defesa1 e defesa2).

- Cada ponto representa uma posição no tabuleiro, com símbolos indicando água (~), navio (#) ou acerto (X).

**Batalha Naval (Classe Principal)**

*Atributos:*

- ataque1: String [] []

- ataque2: String [] []

- defesa1: String [] []

- defesa2: String [] []

*Métodos:*

+ main()

+ menu()

+ posicionar()

+ gerarCampo()

+ verTabuleiro()

+ modificaTabuleiro()

+ escolhaAtaque()

+ limpartela()

**String [] [] (Matriz 9x9)**

**Representa um tabuleiro de jogo (Ataque ou Defesa).**

*Conteúdo das Células:*

"~": Água não atingida (estado inicial).

"#": Parte de navio.

"X": Acerto em navio.

"O": Tiro na água (erro).

Linha 0: Letras (A-H)

Coluna 0: Números (1-8)

**Relações entre a classe App e as matrizes String [] []:**

Inicializa/Cria: gerarCampo()

Lê/Acessa: verTabuleiro(), escolhaAtaque()

Modifica/Atualiza: posicionar(), modificaTabuleiro()

***Principais métodos***

* main(): Orquestra todo o jogo, inicializando variáveis, controlando os laços de repetição das rodadas e declarando o vencedor.
* gerarCampo(String[][] navios): Inicializa as matrizes do tabuleiro, preenchendo as bordas com coordenadas e o campo de jogo com o símbolo de água ("~").
* posicionar(String[][] navios): Posiciona os navios de forma aleatória e automática no tabuleiro de defesa de um jogador. Utiliza a classe Random para definir as posições.
* escolhaAtaque(String[][] ataque, String[][] defesa): Solicita as coordenadas de ataque ao jogador, verifica se a posição já foi atacada e determina se o tiro acertou a água ("O") ou um navio ("X").
* modificaTabuleiro(String posicao, String[][] tabuleiro): Atualiza a matriz do tabuleiro (seja de ataque ou defesa) com o resultado de uma jogada ("X" ou "O").
* menu(int rodadas): Exibe o menu de opções e captura a escolha do jogador.

**Entrada e Saida:**

- Entradas: Scanner (ex: coordenadas para posicionar ou atacar).

- Saídas: exibição do campo, resultados dos ataques, mensagens de vitória ou empate.

Decisões de implementação:

- O jogo foi limitado a 20 rodadas por jogador (40 no total), para evitar partidas infinitas.

- O posicionamento dos navios é feito de forma automática e aleatória pelo método posicionar().

**Diagrama Ilustrativo da Estrutura de Dados**

**EXISTE UM ARQUIVO PNG COM O FLUXOGRAMA**

Por se tratar de uma imagem grande não foi possível anexar junto ao documento, mas o mesmo se encontra junto a documentação enviada

**Descrição do Fluxo**

1. **Início:** O programa começa.
2. **Loop 'reiniciar'**: O jogo está dentro de um grande laço que permite reiniciar uma nova partida assim que a anterior termina.
3. **Inicialização**: No início de cada partida, as variáveis são preparadas: as rodadas são definidas como 40, os placares de acerto são zerados e os quatro tabuleiros (ataque e defesa para cada jogador) são criados com o método gerarCampo().
4. **Loop 'rodadas'**: Este é o laço principal do jogo, que continua enquanto houver rodadas disponíveis.
5. **Menu e Switch**: A cada turno, o menu é exibido e o programa aguarda a escolha do jogador, direcionando o fluxo de acordo com a opção (de 1 a 5).
   * **Opção 1 (Posicionar)**: Verifica se a partida já começou (iniciado == false). Se não, posiciona os navios aleatoriamente para ambos os jogadores.
   * **Opção 2 (Atacar)**: Determina de qual jogador é o turno. Pede as coordenadas do ataque, atualiza os tabuleiros de ataque do jogador atual e de defesa do oponente, verifica se foi um acerto e, em caso positivo, incrementa o placar. Ao final, decrementa o número de rodadas.
   * **Opções 3 e 4**: Apenas exibem informações (tabuleiros de ataque ou rodadas restantes) e o laço continua.
   * **Opção 5 (Sair)**: Força o fim da partida e do programa, ajustando as variáveis de controle dos laços.
6. **Verificação de Vitória Antecipada**: Após um ataque, o código verifica se algum jogador atingiu 10 acertos. Se sim, as rodadas são zeradas para encerrar a partida.
7. **Fim de Jogo**: Quando o laço de rodadas termina (seja por acabar o número de rodadas ou por vitória antecipada), o programa avalia quem é o vencedor com base no número de acertos.
8. **Exibição de Resultados**: O resultado (Vitória do Jogador 1, Vitória do Jogador 2 ou Empate) é exibido na tela, junto com os placares finais e a visualização de todos os quatro tabuleiros.
9. **Reiniciar ou Sair**: Após mostrar os resultados, o fluxo volta ao início do primeiro laço. Se o jogador não escolheu a opção "Sair" (5) durante o jogo, uma nova partida será iniciada. Caso contrário, o programa termina.

**CONCLUSAO**

O desenvolvimento deste trabalho ajudou de forma crítica a melhorar a compreensão de fundamentos essenciais para a programação, especialmente no uso de Arrays bidimensionais, controle de fluxo e modularização. A principal dificuldade foi gerenciar o estado de múltiplas matrizes e lopping’s, e garantir a correta atualização visual do tabuleiro após cada ação. A experiencia foi complexa. No início de difícil compreensão, em organização e implementação de todos os métodos, mas ao decorrer do projeto foi agregando mais conhecimento e aprendizado para novas aplicações.

**BIBLIOGRAFIA**

- Documentação oficial da linguagem Java: <https://docs.oracle.com/javase/8/docs/>

- Stack Overflow: <https://stackoverflow.com/>

**CODIGO FONTE DO TRABALHO EM JAVA**

import java.util.Random;

import java.util.Scanner;

public class App {

    public static *void* main(String[] *args*) {

*int* opcao, rodadas,acerto1, acerto2;

*boolean* reiniciar = true,iniciado;

        String local;

        String ataque1[][] = **new** String[9][9], ataque2[][] = **new** String[9][9];

        String defesa1[][] = **new** String[9][9], defesa2[][] = **new** String[9][9];

        do {

            iniciado=false;

            rodadas = 40;

            acerto1=0;

            acerto2=0;

            ataque1 = gerarCampo(ataque1);

            ataque2 = gerarCampo(ataque2);

            defesa1 = gerarCampo(defesa1);

            defesa2 = gerarCampo(defesa2);

            limpartela();

            do {

                opcao = App.menu(0);

                switch (opcao) {

                    case 1:

                        limpartela();

                        if (iniciado==false) {

                            defesa1=posicionar(defesa1);

                            defesa2=posicionar(defesa2);

                        } else {

                            System.out.println("O Jogo ainda está em andamento, não é possivel reposicionar.");

                        }

                        break;

                    case 2:

                        limpartela();

                        iniciado=true;

                        if (rodadas % 2 == 0) {

                            cabeca();

                            System.out.println("  Jogador Nº1  ");

                            verTabuleiro(ataque1);

                            local = escolhaAtaque(ataque1, defesa2);

                            ataque1 = modificaTabuleiro(local, ataque1);

                            defesa2 = modificaTabuleiro(local, defesa2);

                            local = String.valueOf(local.charAt(2));

                            if (local.equals("X")) {

                                System.out.println("Parabens, seu tiro acertou um navio");

                                acerto1++;

                            } else {

                                System.out.println("Você errou, tiro na agua!");

                            }

                        } else {

                            cabeca();

                            System.out.println("  Jogador Nº2  ");

                            verTabuleiro(ataque2);

                            local = escolhaAtaque(ataque1, defesa2);

                            ataque2 = modificaTabuleiro(local, ataque2);

                            defesa1 = modificaTabuleiro(local, defesa1);

                            local = String.valueOf(local.charAt(2));

                            if (local.equals("X")) {

                                System.out.println("Parabens, seu tiro acertou um navio");

                                acerto2++;

                            } else {

                                System.out.println("Você errou, tiro na agua!");

                            }

                        }

                        rodadas--;

                        break;

                    case 3:

                        limpartela();

                        System.out.println("\nTabuleiro 1");

                        verTabuleiro(ataque1);

                        System.out.println("\nTabuleiro 2");

                        verTabuleiro(ataque2);

                        break;

                    case 4:

                        limpartela();

                        System.out.println("Rodadas restantes: " + rodadas);

                        break;

                    case 5:

                        limpartela();

                        rodadas = 0;

                        reiniciar = false;

                        break;

                    default:

                        limpartela();

                        System.out.println("Opção invalida");

                }

                if(acerto1==10 || acerto2==10){

                    rodadas=0;

                }

            } while (rodadas > 0);

            limpartela();

            //fim de jogo

            if(acerto1==10 || acerto1>acerto2){

                //vitoria 1

                System.out.println("Vitoria do Jogador Nº1");

                System.out.println("Total de acertos Jogador Nº 1: "+acerto1);

                System.out.println("Total de acertos Jogador Nº 2: "+acerto2);

                System.out.println("\nTabuleiro Ataque 1");

                verTabuleiro(ataque1);

                System.out.println("\nTabuleiro Ataque 2");

                verTabuleiro(ataque2);

                 System.out.println("\nTabuleiro Defesa 1");

                verTabuleiro(defesa1);

                System.out.println("\nTabuleiro Defesa 2");

                verTabuleiro(defesa2);

            }else if(acerto2==10 || acerto2>acerto1){

                //vitoria 2

                System.out.println("Vitoria do Jogador Nº2");

                System.out.println("Total de acertos Jogador Nº 1: "+acerto1);

                System.out.println("Total de acertos Jogador Nº 2: "+acerto2);

                System.out.println("\nTabuleiro Ataque 1");

                verTabuleiro(ataque1);

                System.out.println("\nTabuleiro Ataque 2");

                verTabuleiro(ataque2);

                 System.out.println("\nTabuleiro Defesa 1");

                verTabuleiro(defesa1);

                System.out.println("\nTabuleiro Defesa 2");

                verTabuleiro(defesa2);

            }else{

                //empate

                System.out.println("Empate tecnico");

                System.out.println("Total de acertos Jogador Nº 1: "+acerto1);

                System.out.println("Total de acertos Jogador Nº 2: "+acerto2);

                System.out.println("\nTabuleiro Ataque 1");

                verTabuleiro(ataque1);

                System.out.println("\nTabuleiro Ataque 2");

                verTabuleiro(ataque2);

                 System.out.println("\nTabuleiro Defesa 1");

                verTabuleiro(defesa1);

                System.out.println("\nTabuleiro Defesa 2");

                verTabuleiro(defesa2);

            }

        } while (reiniciar == true);

    }

    //CABEÇALHO

    static *void* cabeca() {

        System.out.println("==============================");

        System.out.println("         BATALHA NAVAL        ");

        System.out.println("==============================");

    }

    //EXIBIR MENU

    static *int* menu(*int* *rodadas*) {

*int* opcao;

        Scanner teclado = **new** Scanner(System.in);

        cabeca();

        System.out.println("\n======= MENU =======");

        System.out.println("1 - Posicionar navios automaticamente");

        System.out.println("2 - Atacar oponente");

        System.out.println("3 - Ver tabuleiro de ataques");

        System.out.println("4- Ver rodadas restantes");

        System.out.println("5 - Sair do jogo");

        System.out.println("====================");

        System.out.print("Digite a opção desejada: ");

        opcao=teclado.nextInt();

        return opcao;

    }

    //POSICIONAR NAVIOS

    static String[][] posicionar(String[][] *navios*) {

        Random r = **new** Random();

*int* linha,coluna;

*boolean* vertical, valido=false;

            //Navio 3 espaço

            do {

                linha=r.nextInt(6)+1;

                coluna=r.nextInt(6)+1;

                vertical=r.nextBoolean();

                if(vertical==true && linha<5){

                    valido=true;

*navios*[linha][coluna]="#";

*navios*[linha+1][coluna]="#";

*navios*[linha+2][coluna]="#";

                }else if(vertical==false && coluna<5){

                    valido=true;

*navios*[linha][coluna]="#";

*navios*[linha][coluna+1]="#";

*navios*[linha][coluna+2]="#";

                }

            } while (valido==false);

            valido=false;

            //Navios 2 espaço

            for (*int* i = 0; i < 2; i++) {

                do {

                    linha = r.nextInt(6) + 1;

                    coluna = r.nextInt(6) + 1;

                    vertical = r.nextBoolean();

                    if (vertical == true && linha<6 && *navios*[linha][coluna].equals("~") && *navios*[linha+1][coluna].equals("~")) {

                        valido=true;

*navios*[linha][coluna]="#";

*navios*[linha+1][coluna]="#";

                    } else if(vertical == false && coluna<6 && *navios*[linha][coluna].equals("~") && *navios*[linha][coluna+1].equals("~")){

                        valido=true;

*navios*[linha][coluna]="#";

*navios*[linha][coluna+1]="#";

                    }

                } while (valido == false);

                valido = false;

            }

            //Navios 1 espaço

            for (*int* i = 0; i < 3; i++) {

                do {

                    linha = r.nextInt(6) + 1;

                    coluna = r.nextInt(6) + 1;

                    if(*navios*[linha][coluna].equals("~")){

                        valido=true;

*navios*[linha][coluna]="#";

                    }

                } while (valido == false);

                valido=false;

            }

        return *navios*;

    }

    //GERA UM NOVO CAMPO

    static String[][] gerarCampo(String[][] *navios*) {

        for (*int* i = 0; i < *navios*.length; i++) {

            for (*int* j = 0; j < *navios*[i].length; j++) {

                if (i == 0) {

                    switch (j) {

                        case 0:

*navios*[i][j] = " ";

                            break;

                        case 1:

*navios*[i][j] = "A";

                            break;

                        case 2:

*navios*[i][j] = "B";

                            break;

                        case 3:

*navios*[i][j] = "C";

                            break;

                        case 4:

*navios*[i][j] = "D";

                            break;

                        case 5:

*navios*[i][j] = "E";

                            break;

                        case 6:

*navios*[i][j] = "F";

                            break;

                        case 7:

*navios*[i][j] = "G";

                            break;

                        case 8:

*navios*[i][j] = "H";

                            break;

                        default:

                            throw **new** AssertionError();

                    }

                } else if (j == 0) {

                    switch (i) {

                        case 0:

*navios*[i][j] = " ";

                            break;

                        case 1:

*navios*[i][j] = "1";

                            break;

                        case 2:

*navios*[i][j] = "2";

                            break;

                        case 3:

*navios*[i][j] = "3";

                            break;

                        case 4:

*navios*[i][j] = "4";

                            break;

                        case 5:

*navios*[i][j] = "5";

                            break;

                        case 6:

*navios*[i][j] = "6";

                            break;

                        case 7:

*navios*[i][j] = "7";

                            break;

                        case 8:

*navios*[i][j] = "8";

                            break;

                        default:

                            throw **new** AssertionError();

                    }

                } else {

*navios*[i][j] = "~";

                }

            }

        }

        return *navios*;

    }

    //EXITE TABULEIRO

    static *void* verTabuleiro(String[][] *navios*){

        for(*int* linha=0; linha < *navios*.length; linha++){

            for(*int* coluna=0; coluna < *navios*[linha].length; coluna++){

                System.out.print(*navios*[linha][coluna]);

            }

            System.out.print("\n");

        }

    }

    //MODIFICA TABULEIRO

    static String[][] modificaTabuleiro(String *posicao*, String[][] *tabuleiro*){

*int* coluna, linha;

        String caracter;

        linha=*posicao*.charAt(0)-'0';

        coluna=*posicao*.charAt(1)-'0';

        caracter = String.valueOf(*posicao*.charAt(2));

*tabuleiro*[linha][coluna]=caracter;

        return *tabuleiro*;

    }

    //CAPTURAR ATAQUE

    static String escolhaAtaque(String[][] *ataque*, String *defesa*[][]) {

        Scanner teclado = **new** Scanner(System.in);

*boolean* valido = false;

        String entrada,saida;

*int* coluna=0, linha=0;

        do {

            //loop para converter letras em numero e verificar se está dentro

            do {

                valido=true;

                System.out.print("Informe a letra da coluna que deseja atacar: ");

                entrada = teclado.next();

                if (entrada.equals("a") || entrada.equals("A")) {

                    coluna = 1;

                } else if (entrada.equals("b") || entrada.equals("B")) {

                    coluna = 2;

                } else if (entrada.equals("c") || entrada.equals("C")) {

                    coluna = 3;

                } else if (entrada.equals("d") || entrada.equals("D")) {

                    coluna = 4;

                } else if (entrada.equals("e") || entrada.equals("E")) {

                    coluna = 5;

                } else if (entrada.equals("f") || entrada.equals("F")) {

                    coluna = 6;

                } else if (entrada.equals("g") || entrada.equals("G")) {

                    coluna = 7;

                } else if (entrada.equals("h") || entrada.equals("H")) {

                    coluna = 8;

                } else {

                    System.out.println("Letra invalida!");

                    valido = false;

                }

                System.out.print("Informe o numero da linha que deseja atacar: ");

                linha = teclado.nextInt();

                if (linha < 0 || linha > 8) {

                    System.out.println("Numero invalido!");

                    valido = false;

                }

            } while (valido == false);

            valido = false;

            if(*ataque*[linha][coluna].equals("~")){

                valido=true;

            }else{

                System.out.println("Local invalido para ataque");

            }

        } while (valido == false);

        if(*defesa*[linha][coluna].equals("~")){

            //erro

            saida=String.valueOf(linha)+String.valueOf(coluna)+"O";

        }else{

            //acerto

            saida=String.valueOf(linha)+String.valueOf(coluna)+"X";

        }

        return saida;

    }

    //LIMPAR TELA

    static *void* limpartela(){

        for(*int* i=0; i<40; i++){

            System.out.print("\n");

        }

    }

}