Clínica-De-Bugs(POO: Programação Orientada a Objetos).

Implementação contendo erros, fornecida pelo professor para ser corrigida.

Atividade 10: Procurar_Numero_Exibir_Posicao.

CSharp

```
int[,] matriz = new int[3];
for (int i = 0; i <= 3; i++)
{
    for (int j = 0; j <= 3; j++)
    {
        Console.Write($"Valor [{i},{j}]: ");
        matriz[i, j] = int.Parse(Console.ReadLine());
    }
}

Console.Write("Número a buscar: ");
int buscado = int.Parse(Console.ReadLine())

bool encontrado = null;
int linha = 0, coluna = 0;

for (int i = 0; i <= 3; i++)
{
    for (int j = 0; j <= 3; j++)
    {
        if (matriz[i, j] = buscado)
        {
            encontrado == true;
            linha = j;
            coluna = i;
            break;
        }
    }
    if (encontrado) break;
}

if (encontrado = true)
{
    Console.writeline($"Encontrado em ({linha + 1}, {coluna + 1})")
}
else
{
    Console.WriteLine("Número não encontrado: {buscado}");
}
```

Implementação corrigida, contendo comentários para marcar erros e correções

CSharp

```
int[,] matriz = new int[3, 3]; // Corrigindo a declaração da matriz
string? input = null;
for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        Console.Write($"Valor [{i},{j}]: ");
        input = Console.ReadLine();
        if (input is null) // Verifica se a entrada não é nula antes de
tentar usá-la.
        {
            Console.WriteLine("Entrada inválida.");
            return;
        }
        matriz[i, j] = int.Parse(input);
    }
}

Console.Write("Número a buscar: ");
input = Console.ReadLine();
if (input is null) // Verifica se a entrada não é nula antes de tentar
usá-la.
{
    Console.WriteLine("Entrada inválida.");
    return;
}
int buscado = int.Parse(input); // Adicionando ;

bool encontrado = false; // Boolean não deve ser inicializado com null
int linha = 0, coluna = 0;

for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        if (matriz[i, j] == buscado) // Corrigindo o operador de comparação
        {
            encontrado = true; // Corrigindo a atribuição
            linha = i;
            coluna = j; // Linha é i e coluna é j
            break;
```

```
        }
    }
    if (encontrado) break;
}

if (encontrado == true) // Comparação explícita com true
{
    Console.WriteLine($"Encontrado em ({linha + 1}, {coluna + 1})"); //
Adicionando ; e modificando writeline para WriteLine
}
else
{
    Console.WriteLine($"Número não encontrado: {buscado}"); // Deve-se
utilizar o símbolo $ para interpolação de strings
}
```

Ficha de Erros						
Problema nº	**ERR - 001**	**ERR - 002**	**ERR - 003**	**ERR - 004**	**ERR - 005**	**ERR - 006**
Projeto/Arquivo:	Program.cs	Program.cs	Program.cs	Program.cs	Program.cs	Program.cs
Linha(s) Afetada(s):	Linha 12, Col 44	Linha 34, Col 68	Linha 01, Col 17	Linha 14, Col 19	Linha 21, Col 13	Linha 34, Col 13
Tipo:	Sintático	Sintático	Sintático	Sintático	Sintático	Sintático
Mensagem do compilador /exceção:	error CS1003: Erro de sintaxe, "," esperado	error CS1002: ; esperado	error CS0029: Não é possível converter implicitamente tipo "int[]" em "int[*,*]"	error CS0037: Não é possível converter o valor nulo em 'bool' porque ele não é um tipo de valor não anulável	error CS0029: Não é possível converter implicitamente tipo "int" em "bool"	error CS0117: "Console" não contém uma definição para "writeline"
Hipótese (por que ocorre?):	O compilador detectou um ponto e vírgula ';' ausente.	O compilador detectou um ponto e vírgula ';' ausente.	A declaração do vetor está incorreta.	O valor da variável booleana não deve ser nula.	O operador de comparação do bloco deve ser == e não =.	Não existe a função “writeline”

Experimento (o que testei?):	Adicionado um ponto e vírgula (;) após o final da declaração	Adicionado um ponto e vírgula (;) após o final da declaração	Trocado int[] por int[,].	Trocado null por false.	Trocado = por ==.	Trocado writeline por WriteLine.
Descrição da correção:	Um ponto e vírgula é necessário no final de cada instrução em C#	Um ponto e vírgula é necessário no final de cada instrução em C#	Deve-se declarar a matriz como um vetor de vetores, declarando corretamente.	A inicialização da variável booleana deve ser true ou false.	A comparação deve ser feita com o operador correto.	O compilador é case sensitive.
Teste de regressão (entradas/saídas esperadas):						
Regra/checklist:	Cuidar com o ponto e vírgula no final de cada declaração	Cuidar com o ponto e vírgula no final de cada declaração	Verificar se as variáveis estão sendo corretamente declaradas.	Verificar se as variáveis estão sendo corretamente declaradas.	Verificar as condições corretas.	Verificar se as funções estão sendo chamadas corretamente.

Ficha de Erros						
Problema nº	**ERR - 007**	**ERR - 008**	**ERR - 009**	**ERR - 010**	**ERR - 011**	**ERR - 012**
Projeto/Arquivo:	Program.cs	Program.cs	Program.cs	Program.cs	Program.cs	Program.cs
Linha(s) Afetada(s):	Linha 23, Col 13	Linha 34, Col 68	Linha 38, Col 23	Linha 14, Col 19	Linha 08, Col 18	Linha 17, Col 20 e Linha 19, Col 24
Tipo:	Sintático	Sintático	Semântico	Execução	Execução	Execução
Mensagem do compilador /exceção:	error CS0201: Somente as expressões de atribuição, chamada, incremento , diminuição, espera e novo objeto podem ser utilizadas como uma instrução	error CS1002: ; esperado		warning CS8604: Possível argumento de referência nula para o parâmetro 's' em 'int int.Parse(string s)'.	warning CS8604: Possível argumento de referência nula para o parâmetro 's' em 'int int.Parse(string s)'.	
Hipótese (por que ocorre?):	A atribuição é feita com o operador = e não ==.	O compilador detectou um ponto e vírgula ';' ausente.	É necessário o símbolo “$” para apresentar o valor da variável *total*.	Porque existe a possibilidade da variável ser nula em tempo de execução.	Porque existe a possibilidade da variável ser nula em tempo de execução.	A condição do loop estoura o tamanho do vetor

Experimento (o que testei?):	Alterado o operador == por =.	Adicionado um ponto e vírgula (;) após o final da declaração	Adição do símbolo “$”.	Existem várias formas de solucionar este erro. Foi adicionado uma condição if-else para verificação do estado da variável.	Existem várias formas de solucionar este erro. Foi adicionado uma condição if-else para verificação do estado da variável.	Modificado <= para <.
Descrição da correção:	É necessário fazer a comparação com o operador correto.	Um ponto e vírgula é necessário no final de cada instrução em C#	Garante que o valor apresentado ao usuário será o correto.	Verifica o estado da variável inserida pelo usuário e apenas executa a lógica do código caso a entrada não seja nula.	Verifica o estado da variável inserida pelo usuário e apenas executa a lógica do código caso a entrada não seja nula.	Modificado <= para <, assim interrompendo o loop antes de estourar o tamanho do vetor.
Teste de regressão (entradas/saídas esperadas):						
Regra/checklist:	Sempre verificar as condições.	Cuidar com o ponto e vírgula no final de cada declaração	Garantir que o código está funcionando da maneira correta.	Sempre verificar o estado de variáveis possivelmente nulas.	Sempre verificar o estado de variáveis possivelmente nulas.	Sempre se atentar ao tamanho do vetor e à lógica do loop.

MAPA RÁPIDO DE CÓDIGOS ÚTEIS (PARA CONSULTA)	
CS1002 faltando ;	**CS0103** variável inexistente
CS0029 tipagem errada	**CS8602** variável pode ser nula

TABELA DE TESTES			
Caso	Entradas simuladas	O que observar	Saída esperada
1	‘palavra’	Tipagem correta de variáveis	The input string 'palavra' was not in a correct format.
2	50	Limite estabelecido no loop while.	
3	49	Funcionamento do loop.	

Resumo de Aprendizagem

A atividade permitiu exercitar atenção e cuidado com erros facilmente evitáveis, como a falta de ponto e vírgula no final de declarações e a lógica de tamanho de vetores, sendo assim possível visualizar com clareza os problemas que podem advir da declaração incorreta de loops condicionais e o acesso indevido fora dos limites de um vetor. Também permitiu o reforço do conhecimento em relação à declaração de variáveis possivelmente nuláveis.

Registro de uso de IA

A I.A. foi utilizada principalmente para transcrever o texto do pdf com os exercícios, pois não é possível copiar diretamente do arquivo disponibilizado no GitHub. Ex. de prompt: “copia e cola exatamente, sem mudar nada e sem corrigir”.