1º quesito: Um centro materno-infantil deseja criar um programa para recomendar aos médicos sobre o tipo de parto a ser adotado. O mecanismo de recomendação utiliza o peso do feto e quantidade de semanas de gestação para sugerir o tipo de parto mais indicado. Desenvolva um programa na linguagem C, o qual deverá:

- Ler o peso do feto em gramas e a quantidade de semanas da gestação. Caso o peso do feto seja inferior que 100 gramas ou a quantidade de semanas menor que 28, o programa deverá exibir a mensagem "Parto não deverá ser realizado, reavaliar clinicamente" e encerrar a execução.

- Caso contrário, o programa deverá calcular a quantidade de meses (considerar 4 semanas para cada mês) do feto e exibir uma das recomendações abaixo:
  1. Peso superior a 2.500 gramas e com mais de 7 meses: "Parto normal";
  2. Peso superior a 2.500 gramas e abaixo ou com 7 meses: "Parto Cesariana";
  3. Entre 2.000 gramas e 1.500 gramas e acima de 9 meses: "Parto normal";
  4. Qualquer outra combinação, "Parto Cesariana".

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
	float peso_gestacao;
	int qtd_semanas;
	int meses;

	printf("Quantos gramas o feto tem na gestacao: ");
	scanf("%f", &peso_gestacao);

	printf("Quantas semanas tem a gestaçao: ");
	scanf("%d", &qtd_semanas);

	if (peso_gestacao < 100 || qtd_semanas < 28) {
		printf("O parto nao devera ser ralizado, reavaliar clinicamente!!");
	}

	meses = qtd_semanas / 4;

	if (peso_gestacao > 2500 && meses > 7) {
		printf("Parto Normal!!");
	}
	if (peso_gestacao > 2500 && meses <= 7) {
		printf("Parto Cesariana!!");
	}
	if (peso_gestacao >= 1500 && peso_gestacao <= 2000 && meses > 9) {
		printf("Parto Normal!!");
	}
	else {
		printf("Parto Cesariana!!");
	}

	return 0;
```

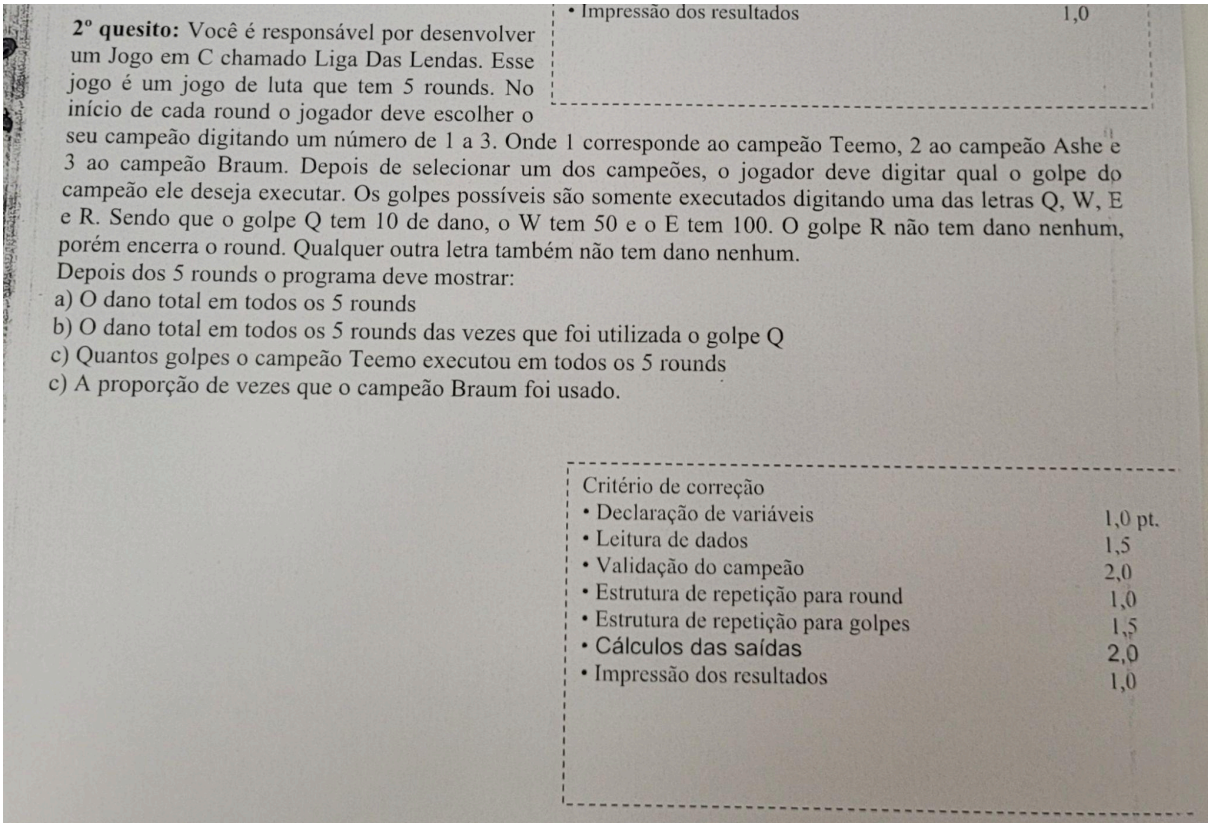
}

2º quesito: Um número perfeito é um número inteiro para o qual a soma de todos os seus divisores positivos próprios (excluindo ele mesmo) é igual ao próprio número. Por exemplo, o número 6 é um número perfeito, pois: 6 = 1 + 2 + 3. O próximo número perfeito é o 28, pois: 28 = 1 + 2 + 4 + 7 + 14.
A matemática ainda não sabe se a quantidade de números perfeitos pares é ou não finita. Não se sabe também se existem números perfeitos ímpares. Escreva um programa em C que realize as seguintes operações:

a) Leia um número inteiro e verifique se ele é par, caso seja impar obrigue o usuário a digitar outro número até que um número par seja digitado;
b) Verifique se o número digitado é perfeito e imprima uma mensagem tela indicando se o número digitado é perfeito ou não.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int main(int argc, char *argv[]) {
	int n, divisores = 100, verificar;
	int i;

	printf("Entre com um inteiro n: ");
	scanf("%d", &n);

	if (n % 2 == 0) {
	for (i = 1; i <= divisores; i++) {

	if (i != 0 && n % i == 0 && n != i) { //O divisor ou denominar da divisão que no caso eh 'i' tem que ser diferente de 0 para dar indeterminação, e o resto da divisao tem que ser igual 0;
					printf("\nOs divisores de %d sao: %d", n, i);
					verificar += i; //somar todos os divisores
			}
		}
		if (n == verificar) {
			printf("\n\n%d eh um numero perfeito!", n);
		}
		else {
			printf("\n\n%d nao eh um numero perfeito!!", n);
		}
	}
	else {
		while (n % 2 != 0) {
			printf("Entre com um inteiro n: ");
			scanf("%d", &n);
		}
	}
```

```
        return 0;
}
```



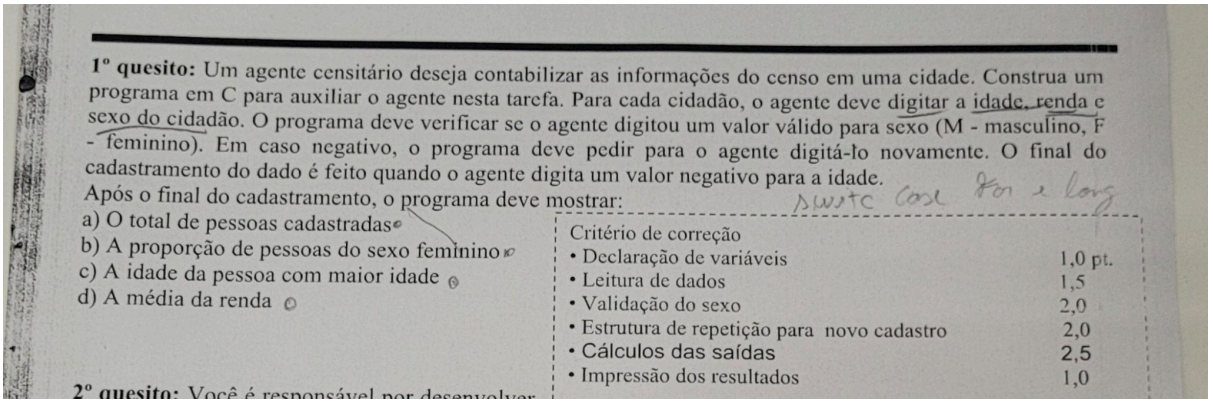
2º quesito: Você é responsável por desenvolver um Jogo em C chamado Liga Das Lendas. Esse jogo é um jogo de luta que tem 5 rounds. No início de cada round o jogador deve escolher o seu campeão digitando um número de 1 a 3. Onde 1 corresponde ao campeão Teemo, 2 ao campeão Ashe e 3 ao campeão Braum. Depois de selecionar um dos campeões, o jogador deve digitar qual o golpe do campeão ele deseja executar. Os golpes possíveis são somente executados digitando uma das letras Q, W, E e R. Sendo que o golpe Q tem 10 de dano, o W tem 50 e o E tem 100. O golpe R não tem dano nenhum, porém encerra o round. Qualquer outra letra também não tem dano nenhum.
Depois dos 5 rounds o programa deve mostrar:
a) O dano total em todos os 5 rounds
b) O dano total em todos os 5 rounds das vezes que foi utilizada o golpe Q
c) Quantos golpes o campeão Teemo executou em todos os 5 rounds
c) A proporção de vezes que o campeão Braum foi usado.

Critério de correção	
• Declaração de variáveis	1,0 pt.
• Leitura de dados	1,5
• Validação do campeão	2,0
• Estrutura de repetição para round	1,0
• Estrutura de repetição para golpes	1,5
• Cálculos das saídas	2,0
• Impressão dos resultados	1,0

```
#include <stdio.h>

int main() {

    int campeao;
    char golpe;
    int dano_total = 0;
    int golpes_q = 0;
    int usos_teemo = 0;
    int usos_braum = 0;
    int round;

    printf("JOGO LIGA DAS LENDAS");


    for (round = 1; round <= 5; round++) {
        printf("ROUND %d\n", round);


        printf("Escolha seu campeão:\n");
```

```
printf("1 - Teemo\n");
printf("2 - Ashe\n");
printf("3 - Braum\n");
printf("Digite (1-3): ");
scanf("%d", &campeao);

if (campeao < 1 || campeao > 3) {
    printf("Campeão inválido! Usando Teemo.\n");
    campeao = 1;
}

if (campeao == 1) {
    usos_teemo++;
}
else if (campeao == 3) {
    usos_braum++;
}

printf("Digite o golpe (Q, W, E ou R): ");
scanf(" %c", &golpe);

if (golpe >= 'a' && golpe <= 'z') {
    golpe = golpe - 32;
}


int dano = 0;

if (golpe == 'Q') {
    dano = 10;
    golpes_q++;
}
else if (golpe == 'W') {
    dano = 50;
}
else if (golpe == 'E') {
    dano = 100;
}
else if (golpe == 'R') {
    printf("Golpe R encerra o round sem dano.\n");
}
else {
    printf("Golpe inválido! Sem dano.\n");
}


dano_total += dano;
```

```
        printf("Dano neste round: %d\n\n", dano);
    }


    float proporcao_braum = (float)usos_braum / 5;


    int dano_golpe_q = golpes_q * 10;


    printf("RESULTADOS FINAIS:\n");
    printf("a) Dano total: %d\n", dano_total);
    printf("b) Dano total do golpe Q: %d\n", dano_golpe_q);
    printf("c) Golpes do campeão Teemo: %d\n", usos_teemo);
    printf("d) Proporção de uso do Braum: %.2f\n", proporcao_braum);

    return 0;
}
```



1º quesito: Um agente censitário deseja contabilizar as informações do censo em uma cidade. Construa um programa em C para auxiliar o agente nesta tarefa. Para cada cidadão, o agente deve digitar a idade, renda e sexo do cidadão. O programa deve verificar se o agente digitou um valor válido para sexo (M - masculino, F - feminino). Em caso negativo, o programa deve pedir para o agente digitá-lo novamente. O final do cadastramento do dado é feito quando o agente digita um valor negativo para a idade.
Após o final do cadastramento, o programa deve mostrar:
a) O total de pessoas cadastradas
b) A proporção de pessoas do sexo feminino
c) A idade da pessoa com maior idade
d) A média da renda

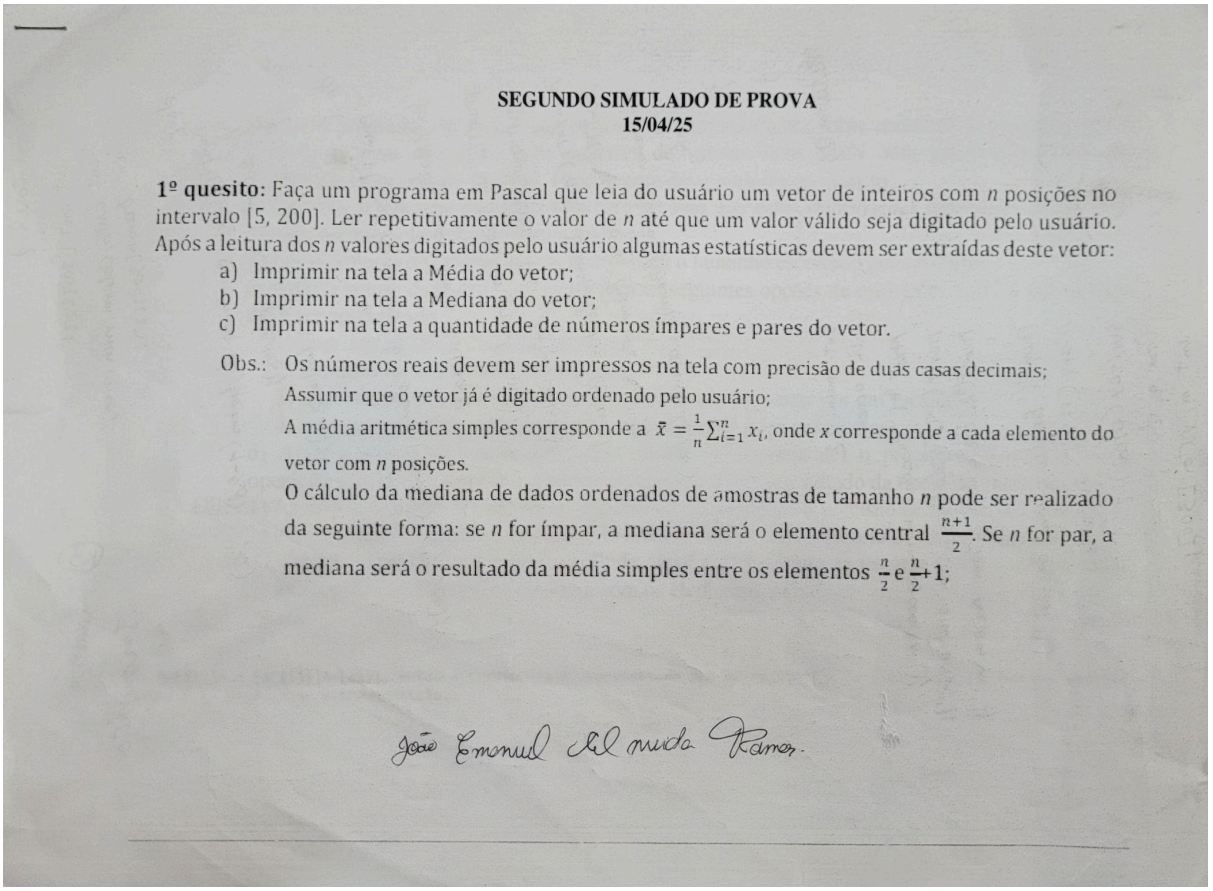
Critério de correção	
• Declaração de variáveis	1,0 pt.
• Leitura de dados	1,5
• Validação do sexo	2,0
• Estrutura de repetição para novo cadastro	2,0
• Cálculos das saídas	2,5
• Impressão dos resultados	1,0

2º quesito: Você é responsável por desenvolver

```
#include <stdio.h>

int main() {

    int idade;
    float renda;
    char sexo;


    int total = 0;
    int mulheres = 0;
    int maior_idade = 0;
    float soma_renda = 0;


    int idade_maxima = 60;
```

```
float proporcao_mulheres;
float media_renda;

printf("SISTEMA DE CENSO\n");
printf("Digite idade negativa para encerrar\n\n");


while (1) {
    printf("Pessoa %d\n", total + 1);

    printf("Digite a idade: ");
    scanf("%d", &idade);


    if (idade < 0) {
        break;
    }


    if (idade > idade_maxima) {
        printf("Idade muito alta! Máximo: %d\n", idade_maxima);
        continue;
    }


    printf("Digite a renda: ");
    scanf("%f", &renda);


    printf("Digite o sexo (M ou F): ");
    scanf(" %c", &sexo);


    if (sexo == 'm') sexo = 'M';
    if (sexo == 'f') sexo = 'F';


    if (sexo != 'M' && sexo != 'F') {
        printf("Sexo inválido! Digite novamente.\n");
        continue;
    }


    total++;

    if (sexo == 'F') {
```

```
            mulheres++;
        }

        if (idade > maior_idade) {
            maior_idade = idade;
        }

        soma_renda += renda;

        printf("Pessoa cadastrada!\n\n");
    }


    proporcao_mulheres = 0;
    if (total > 0) {
        proporcao_mulheres = (float)mulheres / total;
    }

    /* Calcula a média da renda */
    media_renda = 0;
    if (total > 0) {
        media_renda = soma_renda / total;
    }

    /* Mostra os resultados DAS PESSOAS*/
    printf("\nRESULTADOS:\n");
    printf("a) Total de pessoas: %d\n", total);
    printf("b) Proporção de mulheres: %.2f\n", proporcao_mulheres);
    printf("c) Maior idade: %d\n", maior_idade);
    printf("d) Média da renda: R$ %.2f\n", media_renda);

    return 0;
}
```



SEGUNDO SIMULADO DE PROVA
15/04/25

1º quesito: Faça um programa em Pascal que leia do usuário um vetor de inteiros com *n* posições no intervalo [5, 200]. Ler repetitivamente o valor de *n* até que um valor válido seja digitado pelo usuário. Após a leitura dos *n* valores digitados pelo usuário algumas estatísticas devem ser extraídas deste vetor:

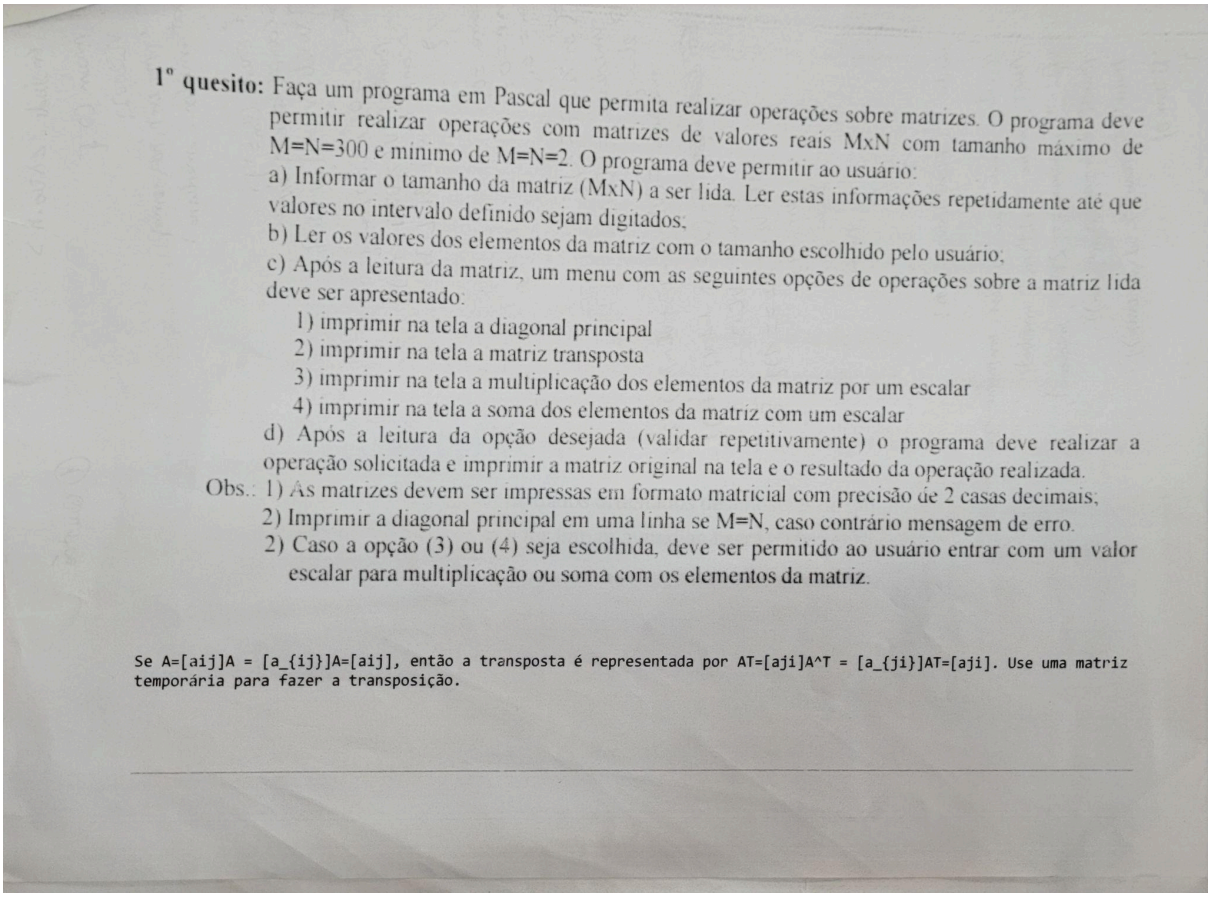
a) Imprimir na tela a Média do vetor;
b) Imprimir na tela a Mediana do vetor;
c) Imprimir na tela a quantidade de números ímpares e pares do vetor.

Obs.: Os números reais devem ser impressos na tela com precisão de duas casas decimais;
Assumir que o vetor já é digitado ordenado pelo usuário;
A média aritmética simples corresponde a $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, onde *x* corresponde a cada elemento do vetor com *n* posições.
O cálculo da mediana de dados ordenados de amostras de tamanho *n* pode ser realizado da seguinte forma: se *n* for ímpar, a mediana será o elemento central $\frac{n+1}{2}$. Se *n* for par, a mediana será o resultado da média simples entre os elementos $\frac{n}{2}$ e $\frac{n}{2}+1$;

João Emanuel Almeida Ramos.

```
#include <stdio.h>
#include <stdlib.h>

#define TAMANHO_MIN 5
#define TAMANHO_MAX 200

int main() {
   int vetor[TAMANHO_MAX];
   int n;
   int cont_pares = 0, cont_impares = 0;
   float media = 0.0, mediana = 0.0;

   // Obter tamanho do vetor
   do {
      printf("Digite o tamanho do vetor (entre %d e %d): ", TAMANHO_MIN,
TAMANHO_MAX);
      scanf("%d", &n);
   } while (n < TAMANHO_MIN || n > TAMANHO_MAX);

   // Ler elementos do vetor
   printf("Digite %d inteiros (assume-se que o vetor já está ordenado):\n", n);
   for (int i = 0; i < n; i++) {
      printf("vetor[%d]: ", i);
```

```
        scanf("%d", &vetor[i]);

        // Calcular soma para média
        media += vetor[i];

        // Contar números pares e ímpares
        if (vetor[i] % 2 == 0) {
            cont_pares++;
        } else {
            cont_impares++;
        }
    }

    // Calcular média
    media /= n;

    // Calcular mediana
    if (n % 2 == 1) {
        // Número ímpar de elementos
        mediana = vetor[n / 2];
    } else {
        // Número par de elementos
        mediana = (vetor[n / 2 - 1] + vetor[n / 2]) / 2.0;
    }

    // Exibir resultados
    printf("\nEstatísticas do Vetor:\n");
    printf("a) Média: %.2f\n", media);
    printf("b) Mediana: %.2f\n", mediana);
    printf("c) Números pares: %d\n", cont_pares);
    printf("   Números ímpares: %d\n", cont_impares);

    return 0;
}
```



1º quesito: Faça um programa em Pascal que permita realizar operações sobre matrizes. O programa deve permitir realizar operações com matrizes de valores reais MxN com tamanho máximo de M=N=300 e mínimo de M=N=2. O programa deve permitir ao usuário:

a) Informar o tamanho da matriz (MxN) a ser lida. Ler estas informações repetidamente até que valores no intervalo definido sejam digitados;

b) Ler os valores dos elementos da matriz com o tamanho escolhido pelo usuário;

c) Após a leitura da matriz, um menu com as seguintes opções de operações sobre a matriz lida deve ser apresentado:

1) imprimir na tela a diagonal principal
2) imprimir na tela a matriz transposta
3) imprimir na tela a multiplicação dos elementos da matriz por um escalar
4) imprimir na tela a soma dos elementos da matriz com um escalar

d) Após a leitura da opção desejada (validar repetitivamente) o programa deve realizar a operação solicitada e imprimir a matriz original na tela e o resultado da operação realizada.

Obs.: 1) As matrizes devem ser impressas em formato matricial com precisão de 2 casas decimais;
2) Imprimir a diagonal principal em uma linha se M=N, caso contrário mensagem de erro.
2) Caso a opção (3) ou (4) seja escolhida, deve ser permitido ao usuário entrar com um valor escalar para multiplicação ou soma com os elementos da matriz.

Se A=[aij]A = [a_{ij}]A=[aij], então a transposta é representada por AT=[aji]A^T = [a_{ji}]AT=[aji]. Use uma matriz temporária para fazer a transposição.

```
#include <stdio.h>
#include <stdlib.h>

#define TAMANHO_MAX 300
#define TAMANHO_MIN 2

int main() {
    float matriz[TAMANHO_MAX][TAMANHO_MAX];
    float resultado[TAMANHO_MAX][TAMANHO_MAX];
    int linhas, colunas;
    int opcao;
    float escalar;

    // Obter dimensões da matriz
    do {
        printf("Digite o número de linhas (entre %d e %d): ", TAMANHO_MIN,
TAMANHO_MAX);
        scanf("%d", &linhas);
    } while (linhas < TAMANHO_MIN || linhas > TAMANHO_MAX);

    do {
        printf("Digite o número de colunas (entre %d e %d): ", TAMANHO_MIN,
TAMANHO_MAX);
```

```
    scanf("%d", &colunas);
} while (colunas < TAMANHO_MIN || colunas > TAMANHO_MAX);

// Ler elementos da matriz
printf("Digite os elementos da matriz:\n");
for (int i = 0; i < linhas; i++) {
    for (int j = 0; j < colunas; j++) {
        printf("matriz[%d][%d]: ", i, j);
        scanf("%f", &matriz[i][j]);
    }
}

// Exibir menu de operações
do {
    printf("\nMENU DE OPERAÇÕES:\n");
    printf("1) Imprimir a diagonal principal\n");
    printf("2) Imprimir a matriz transposta\n");
    printf("3) Multiplicar matriz por um escalar\n");
    printf("4) Somar um escalar à matriz\n");
    printf("0) Sair\n");
    printf("Escolha uma opção: ");
    scanf("%d", &opcao);

    if (opcao < 0 || opcao > 4) {
        printf("Opção inválida. Tente novamente.\n");
        continue;
    }

    if (opcao == 0) {
        break;
    }

    // Processar a opção selecionada
    switch (opcao) {
        case 1: // Diagonal principal
            if (linhas != colunas) {
                printf("Erro: A diagonal principal existe apenas para matrizes quadradas.\n");
            } else {
                printf("\nMatriz Original:\n");
                for (int i = 0; i < linhas; i++) {
                    for (int j = 0; j < colunas; j++) {
                        printf("%.2f\t", matriz[i][j]);
                    }
                    printf("\n");
                }

                printf("\nDiagonal Principal: ");
                for (int i = 0; i < linhas; i++) {
```

```
            printf("%.2f ", matriz[i][i]);
        }
        printf("\n");
    }
    break;

case 2: // Transposta
    printf("\nMatriz Original:\n");
    for (int i = 0; i < linhas; i++) {
        for (int j = 0; j < colunas; j++) {
            printf("%.2f\t", matriz[i][j]);
        }
        printf("\n");
    }

    printf("\nMatriz Transposta:\n");
    for (int j = 0; j < colunas; j++) {
        for (int i = 0; i < linhas; i++) {
            printf("%.2f\t", matriz[i][j]);
        }
        printf("\n");
    }
    break;

case 3: // Multiplicar por escalar
    printf("Digite o valor escalar para multiplicação: ");
    scanf("%f", &escalar);

    printf("\nMatriz Original:\n");
    for (int i = 0; i < linhas; i++) {
        for (int j = 0; j < colunas; j++) {
            printf("%.2f\t", matriz[i][j]);
        }
        printf("\n");
    }

    printf("\nMatriz após multiplicação por %.2f:\n", escalar);
    for (int i = 0; i < linhas; i++) {
        for (int j = 0; j < colunas; j++) {
            resultado[i][j] = matriz[i][j] * escalar;
            printf("%.2f\t", resultado[i][j]);
        }
        printf("\n");
    }
    break;

case 4: // Adicionar escalar
    printf("Digite o valor escalar para adicionar: ");
```

```
            scanf("%f", &escalar);

            printf("\nMatriz Original:\n");
            for (int i = 0; i < linhas; i++) {
                for (int j = 0; j < colunas; j++) {
                    printf("%.2f\t", matriz[i][j]);
                }
                printf("\n");
            }

            printf("\nMatriz após adição de %.2f:\n", escalar);
            for (int i = 0; i < linhas; i++) {
                for (int j = 0; j < colunas; j++) {
                    resultado[i][j] = matriz[i][j] + escalar;
                    printf("%.2f\t", resultado[i][j]);
                }
                printf("\n");
            }
            break;
        }
    } while (opcao != 0);

    return 0;
}
```