# ALGORITMOS E LABORATÓRIO DE PROGRAMAÇÃO

**Prof. Lázaro Oliveira.**

#ConhecimentoTransforma

# APRESENTAÇÃO PROFESSOR/ALUNO

**Prof. Lázaro Oliveira.**

#ConhecimentoTransforma

# APRESENTAÇÃO

## Professor Lázaro Oliveira.

# EMENTA DA DISCIPLINA

**Prof. Lázaro Oliveira.**

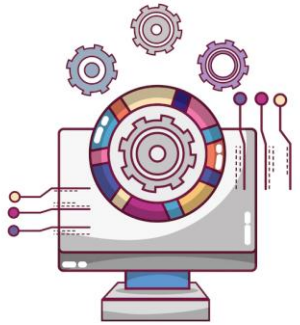
#ConhecimentoTransforma

# EMENTA:

- Revisão do conceito de algoritmo.
- Vetores e Matrizes.
- Estruturas definidas pelo programador (unions, Structs, Typedef e Enum)
- Bibliotecas padrão da linguagem.
- Strings e manipulação de strings.
- Manipulação de Arquivos.
- Algoritmos de ordenação.
- Algoritmos de busca.
- Implementação dos algoritmos em C.

# COMPETÊNCIAS E HABILIDADES

**As competências gerais desenvolvidas ao longo do curso envolvem capacidade de criação de algoritmos complexos em C para aplicações clássicas como manipulação de arquivos e strings, busca e ordenação.**



# OBJETIVO DA DISCIPLINA

**Ao final desta Disciplina, espera-se que o aluno tenha ampliado sua capacidade de desenvolvimento de programas e algoritmos na Linguagem C.**

# CONTEÚDOS PROGRAMÁTICOS

## UNIDADE 1: ALGORITMOS, VETORES, MATRIZES, ESTRUTURAS DEFINIDAS PELO PROGRAMADOR E STRINGS

**OBJETIVO ESPECÍFICO (HABILIDADES):**

- Rever o conceito de algoritmos e aprofundar os conhecimentos de vetores, matrizes e Strings.
- Aplicar o conceito de estruturas de dados definidas pelo programador.

**Conteúdo:**

1.1 - Revisão de Algoritmos e bases de programação.

1.2 - Estruturas vetores, matrizes, structs, Unions, typedef e Enum.

1.3 - Manipulação de Strings e Passagem de estruturas para funções.

# CONTEÚDOS PROGRAMÁTICOS

## UNIDADE 2: MANIPULAÇÃO DE ARQUIVOS

**OBJETIVO ESPECÍFICO (HABILIDADES):**

- **Desenvolver programas capazes de trabalhar com arquivos de texto e binários para leitura, escrita e atualização de dados usando a Linguagem C. Armazenamento de estruturas de dados em arquivos.**

**Conteúdo:**

**2.1 – Arquivo Texto: Criação, Leitura e Gravação.**

**2.2 – Arquivo Binário: Criação, Leitura, Atualização de arquivos.**

**2.3 – Manipulação com Arquivos Textos e Binários. .**

# CONTEÚDOS PROGRAMÁTICOS

## UNIDADE 3: ALGORITMOS DE ORDENAÇÃO

**OBJETIVO ESPECÍFICO (HABILIDADES):**

- **Compreender e aplicar os algoritmos de ordenação Bubble Sort, Selection Sort, Insert Sort, Merge Sort, Quick Sort, Heap Sort e Conting Sort na Linguagem C. Ordenação de um array de struct.**

**Conteúdo:**

**3.1 – Ordenação de dados usando métodos simples (Bubble Sort, Selection Sort e Insertion Sort e outros)**

**3.2 - Ordenação de dados usando métodos eficientes.**

**3.3 - Ordenação de estruturas definidas pelo programador.**

# CONTEÚDOS PROGRAMÁTICOS

## UNIDADE 4: ALGORITMOS DE BUSCA

**OBJETIVO ESPECÍFICO (HABILIDADES):**

- **Compreender e aplicar os algoritmos de busca linear, binária, ternária, entre outras na  Linguagem C.**

**Conteúdo:**

**4.1 - A importância dos algoritmos de busca.**

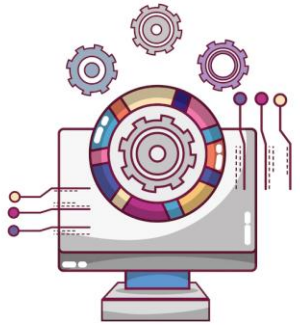
**4.2 – Implementação de algoritmos de busca.**

**4.3 - Comparação entre algoritmos de busca.**

# CONTEÚDOS PROGRAMÁTICOS

## METODOLOGIA

- Aulas expositivas, práticas e dialogadas, podendo contar com o apoio de projeções, além do  desenvolvimento de trabalhos, individuais e/ou em grupo, visando ao preparo dos alunos para o mercado de trabalho profissional. Para isso, as atividades propostas favorecem a autonomia do aluno e a construção do conhecimento.



# CONTEÚDOS PROGRAMÁTICOS

## AVALIAÇÃO

- **Atividades avaliativas individuais ou em grupo compreendendo o conteúdo da disciplina.**

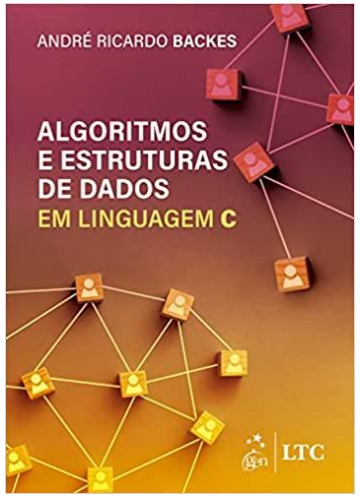
# REFERÊNCIAS BIBLIOGRÁFICAS

**Prof. Lázaro Oliveira.**

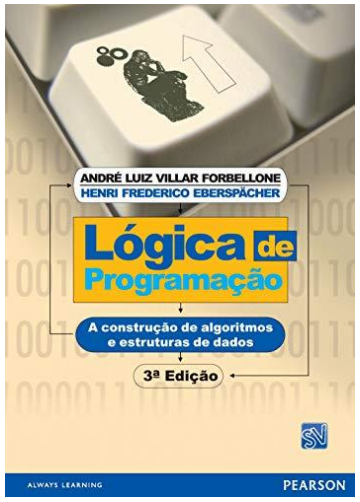
#ConhecimentoTransforma

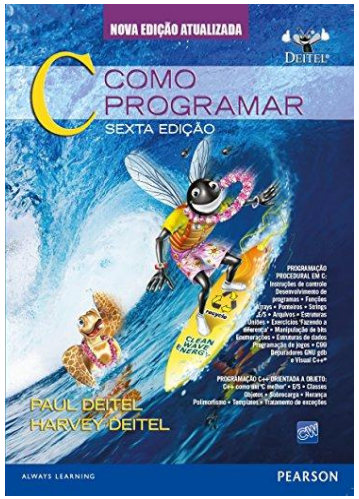
# REFERÊNCIAS BIBLIOGRÁFICAS

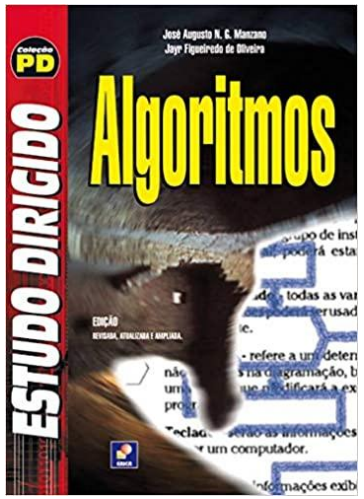


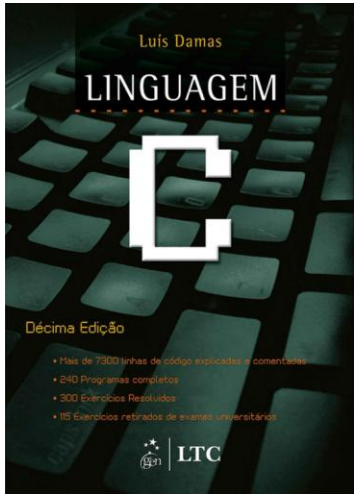




# PLANEJAMENTO DAS AULAS

**Prof. Lázaro Oliveira.**

#ConhecimentoTransforma

# PLANEJAMENTO DAS AULAS: (AJUSTAR)

| SEMANA | CONTEÚDO |
|---|---|
| 1 | Apresentação da Disciplina e do Plano de Aula.<br>Revisão de Algoritmos: Bases Prog., Vetores e Matrizes |
| 2 | Revisão de Algoritmos: Bases Prog., Estruturas basicas e manipulação de Strings. |
| 3 | Manipulação de Arquito Texto |
| 4 | Manipulação de Arquivo Binário |
| 5 | Algoritmos de Ordenação: Bubble sort, selection sort, insertion sort, merge sort (Parte 1) |
| 6 | Algoritmos de ordenação: Heap sort, quick sort, counting sort (Parte 2) |
| 7 | Algoritmos de Busca: Busca linear, busca binária e busca ternária |
| 8 | Aprendizagem baseada em problemas: Método de Ordenação, Método de Busca e Arquivos |
| **9** | **Aplicação Prática Avaliativa.** |
| **10** | **Encerramento do Trimestre Letivo.** |

# CRITÉRIOS: AVALIAÇÃO E APROVAÇÃO

**Prof. MSc. Lázaro Oliveira.**

#ConhecimentoTransforma

# CRITÉRIOS DE AVALIAÇÃO E APROVAÇÃO

**A4**: Atividade Avaliativa ..... **10,0**
Entregar até ??????????????

**CRITÉRIOS DE APROVAÇÃO:**

**A4** ≥ 6,0 e mínimo de **75%** de presença
Então, o aluno estará APROVADO

Observação:
Não existe 2ª chamada para a Atividade Avaliativa (A4)

# ATIVIDADE AVALIATIVA

## Aprendizagem Baseada em Problemas

### Situação-Problema:

A atividade de implementação e desenvolvimento de um sistema envolve diversas estruturas de programação, independentemente da linguagem de programação escolhida. Neste viés, você deverá desenvolver um PROGRAMA EM LIGUAGEM C, que permita **registrar as vendas diárias de uma loja de roupas,** imprimindo no final do dia, os seguintes relatórios:

- ✓ Quantidade total de itens vendidos no dia, no ato do registro da venda, ou seja, assim que finalizar aquela venda específica;
- ✓ Listar todas as vendas realizadas no dia, em ordem decrescente, ou seja, considerar a venda de maior valor prioritariamente, e assim por diante, até que todas sejam listadas. O usuário informará a data da venda;
- ✓ Faturamento bruto diário sob as vendas (o usuário digitará a data);
- ✓ Quantidade de clientes que realizaram compras naquele dia (o usuário digitará a data);
- ✓ Item mais vendido em uma determinada data informada pelo usuário;
- ✓ Item menos vendido em uma determinada data informada pelo usuário.

# ATVIDADE AVALIATIVA

## Aprendizagem Baseada em Problemas

### Metodologia:

Escreva um programa em Linguagem C que obedeça as seguintes diretivas:

- ✓ Cadastrar os seguintes dados por venda realizada: **código** do item, **nome** do item, **marca** do item, **quantidade** de itens e **preço unitário** do item;
- ✓ Após cada entrada de novo item, o programa deverá **chamar uma função para calcular automaticamente o preço pago na venda** realizada para cada item registrado;
- ✓ O programa deverá atribuir um **desconto de 10%** do valor total da venda realizada para cada item, sempre que a quantidade de itens vendidos for **maior ou igual a três unidades**;
- ✓ O programa também deverá calcular automaticamente a **quantidade de clientes que realizaram compras naquele dia**.

# ATVIDADE AVALIATIVA

## Aprendizagem Baseada em Problemas

### Metodologia:

No final do dia, ou seja, quando não tiverem novos clientes a serem registrados, o programa deverá ser finalizado, gerando os seguintes **Relatórios Gerenciais**:

- ✓ Quantidade total de itens vendidos no dia, no ato do registro da venda, ou seja, assim que finalizar aquela venda específica;
- ✓ Listar todas as vendas realizadas no dia, em ordem decrescente, ou seja, considerar a venda de maior valor prioritariamente, e assim por diante, até que todas sejam listadas. O usuário informará a data da venda;
- ✓ Faturamento bruto diário sob as vendas (o usuário digitará a data);
- ✓ Quantidade de clientes que realizaram compras naquele dia (o usuário digitará a data);
- ✓ Item mais vendido em uma determinada data informada pelo usuário;
- ✓ Item menos vendido em uma determinada data informada pelo usuário.

# ATVIDADE AVALIATIVA

## Aprendizagem Baseada em Problemas

### Metodologia:

Você deverá obedecer, obrigatoriamente, os seguintes critérios:

1) Utilizar **struct, array, algoritmo de ordenação, funções, modularização**.
2) Gravar os dados no arquivo “loja_roupa.dat” ou “loja_roupa.txt”.
3) Implementar a solução algorítmica em **linguagem C**.
4) Apresentar os testes realizados, ou seja, inserir os valores de entrada e mostrar os resultados obtidos (saída).
5) Para que seja validado o critério 4, será necessário realizar, ao menos três testes com entrada de dados distintas para cada um deles.

# FERRAMENTAS DE APOIO

**ATENÇÃO**:
Recomenda-se o uso das seguintes ferramentas para apoio ao aprendizado prático desta disciplina.

IDE DEV C++

IDE VS Code Studio

# REVISÃO DE BASES: VETORES E MATRIZES

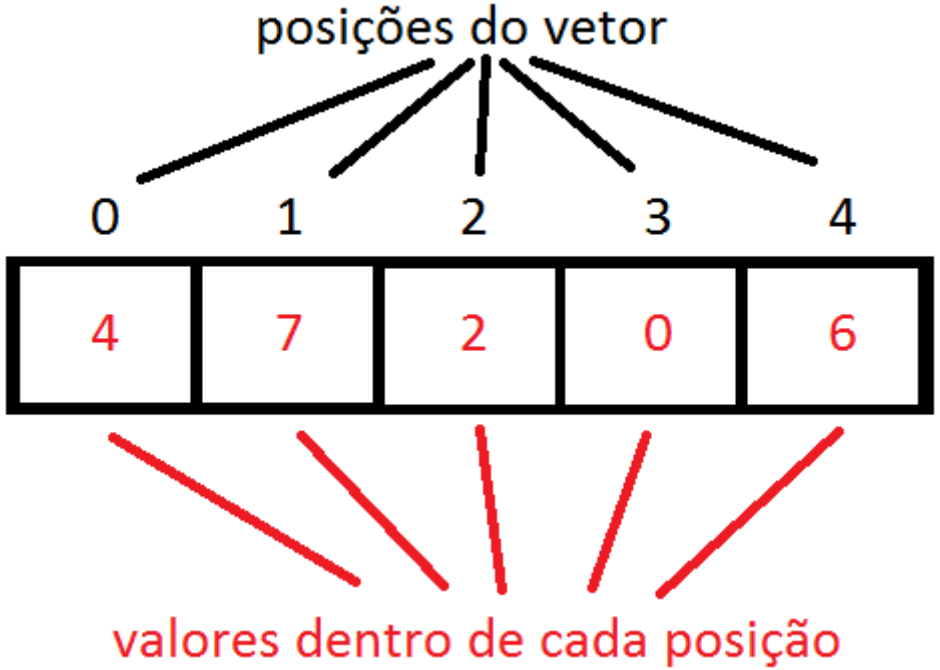
**Prof. Lázaro Oliveira.**

#ConhecimentoTransforma

**Semana 01**

# VETORES & MATRIZES

**ARRAY**:
Estrutura, ARRAY, que armazena temporariamente dados de um programa na memória real.




Vetor Unidimensional

# VETORES & MATRIZES

**ARRAY**:
Estrutura, ARRAY, que armazena temporariamente dados de um programa na memória real.

| | Coluna 0 | Coluna 1 | Coluna 2 | Coluna 3 |
|---|---|---|---|---|
| Linha 0 | P[0][0] | P[0][1] | P[0][2] | P[0][3] |
| Linha 1 | P[1][0] | P[1][1] | P[1][2] | P[1][3] |
| Linha 2 | P[2][0] | P[2][1] | P[2][2] | P[2][3] |

Vetor Multidimensional = Matriz

# VETORES

```c
/* Programa 01: Vetor de notas */
#include <stdio.h>
#include <conio.h>

int main(void)
{
	float notas[5] = {7, 8, 9.5, 9.9, 5.2};
	// declarando e inicializando o vetor notas

	printf("Exibindo os Valores do Vetor \n\n");
	printf("notas[0] = %.1f\n", notas[0]);
	printf("notas[1] = %.1f\n", notas[1]);
	printf("notas[2] = %.1f\n", notas[2]);
	printf("notas[3] = %.1f\n", notas[3]);
	printf("notas[4] = %.1f\n", notas[4]);

	getch();  //pausa até o pressionamento de tecla
	return 0;
}
```

**Resultado obtido:**

```
Exibindo os Valores do Vetor

notas[0] = 7.0
notas[1] = 8.0
notas[2] = 9.5
notas[3] = 9.9
notas[4] = 5.2
```

# VETORES

Prog02.c

```c
/* Programa 02: Vetor de notas - Loop */

#include<stdio.h>
#include<conio.h>

int main(void)
{
  int i;
  float notas[5] = {7, 8, 9.5, 9.9, 5.2};
  // declarando e inicializando o vetor notas

  printf("Exibindo os Valores do Vetor \n\n");

  for( i = 0 ; i <= 4; i++)
  {
    printf("notas[%d] = %.1f\n",i, notas[i]);
  }

  getch();  // pausa até o pressionamento de tecla
  return 0;
}

```

**Resultado obtido:**

```
Exibindo os Valores do Vetor

notas[0] = 7.0
notas[1] = 8.0
notas[2] = 9.5
notas[3] = 9.9
notas[4] = 5.2
```

# VETORES

#include <stdlib.h>  // necessário para limpar a tela (CLS)

[*] Prog03.c

```c
/* Programa 03: Vetor com Modularização */

#include<stdio.h>
#include<conio.h>
#include <locale.h> //necessário para usar setlocale

// Declaração de constante - dimensão máxima do vetor
#define MAX  10

// Função para imprimir elementos de vetor
void imprime_dados_vetor (int vet[], int tam)
{
  int i;
  for (i=0; i<tam; i++)
    printf("%d ", vet[i]);
  printf("\n"); // ao final new line
  }

// Função que define os valores para o vetor (passagem por referência/endereço)
void definirVetor (int v[], int tam)
{
  int i;
  for (i=0; i<tam; i++)
    v[i] = 10+i;
  }

// Função principal
int main (void)
{
  int n=10, vet[MAX];     // alocar MAX posições para o vetor

  definirVetor(vet, n);

  // Imprime o vetor de dados
  system("CLS"); // limpa a tela

  setlocale(LC_ALL, "Portuguese"); // Acentuação gráfica
  printf("\nExibição dos dados do vetor: ");

  imprime_dados_vetor(vet, n);

  getch();  // pausa até o pressionamento de tecla
  return 0;
  }

```

Resultado obtido:

```
Exibição dos dados do vetor: 10 11 12 13 14 15 16 17 18 19
```

# VETORES

Prog04.c

```c
/* Programa 04: Vetor de String */

#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include <string.h> // para usar a função 'strlen(...)'
#include <locale.h> // necessário para usar setlocale

// Função principal
int main (void)
{
    int i;
    char string[] = "0123456789abcdefghij"; // definição da "string"

    // Imprimir cada caractere da string com seu codigo ASCII

    system("CLS"); // limpa a tela

    setlocale(LC_ALL, "Portuguese"); // Acentuação gráfica
    printf("Imprime caracteres e seus códigos ASCII na forma de tabela\n");
    printf("Coluna 1 => caractere | coluna 2 => seu código ASCII\n");

    for (i=0; i<strlen(string); i++)
        printf("%21c | %3d\n", string[i], string[i]);

    printf("\n\nFrase: %s\n", string);
    printf("A string tem 20 caracteres, testando os caracteres...\n");
    printf("carac[0] é '%c' e carac[19] é '%c'\n", string[0], string[19]);

    system("PAUSE");  // pausa temporária até o pressionamento de tecla
    return 0;
}
```

## Resultado obtido:

```
Imprime caracteres e seus códigos ASCII na forma de tabela
Coluna 1 => caractere | coluna 2 => seu código ASCII
                    0 |  48
                    1 |  49
                    2 |  50
                    3 |  51
                    4 |  52
                    5 |  53
                    6 |  54
                    7 |  55
                    8 |  56
                    9 |  57
                    a |  97
                    b |  98
                    c |  99
                    d | 100
                    e | 101
                    f | 102
                    g | 103
                    h | 104
                    i | 105
                    j | 106


Frase: 0123456789abcdefghij
A string tem 20 caracteres, testando os caracteres...
carac[0] é '0' e carac[19] é 'j'
Pressione qualquer tecla para continuar. . .
```

# VETORES

[*] Prog05.c

```c
/* Programa 05: Vetor Modular - Apontador ou Ponteiro de String */

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

// Função ImprimeVetor
void imprimeVetor (int *apont, int n)
{
    int i;
    printf("Vetor:");
    for (i=0; i<n; i++)
        printf(" %d", *(apont+i)); // pega elemento apontado no vetor
    printf("\n");
}

// Função principal
int main (void)
{
    system("CLS"); // Limpa a tela
    int vet1[] = { 5, 4, 3, 2, 1 }; // define vetor com 5 posicoes
    int *apont;
    int N = 5;
    imprimeVetor(vet1, N); // passa para ponteiro "apont" o endereco inicial do vetor
    system("PAUSE");  // pausa temporária até o pressionamento de tecla
    return 0;
}

```

Resultado obtido:

# VETORES

Prog06.c

```c
/* Programa 06: Entrada de Dados no Vetor - Formatação casas decimais */

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

// Função principal
int main (void)
{
    int idade;
    float peso, altura;

    system("CLS"); // Limpa a tela

    // Entrada de dados
    printf("Digite a idade, a altura e o peso, separados por um espaço em branco.\n");
    scanf("%d%f%f", &idade, &altura, &peso);

    // Saída de informações
    printf("Idade: %d\n", idade);
    printf("Altura: %1.2f\n", altura); // Formatação casas decimais
    printf("Peso: %3.3f\n", peso); // Formatação de casas decimais

    system("PAUSE");  // pausa temporária até o pressionamento de tecla
    return 0;
}

```

**Resultado obtido:**

```
Digite a idade, a altura e o peso, separados por um espaþo em branco.
23 1.84 98.355
Idade: 23
Altura: 1.84
Peso: 98.355
Pressione qualquer tecla para continuar. . .
```

# VETORES

Prog07.c

```c
/* Programa 07: Entrada de Dados no Vetor Modular - Formatação casas decimais */

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

// Função Multiplica Números
int multiplica(int N1, int N2) //multiplica recebe N1,N2 e retorna um int
{
  int resultado;
  resultado = N1 * N2;
  return(resultado); //retornando o valor para main
}

// Função principal
int main(void)
{
  int V1, V2, resultado;

  system("CLS"); // Limpa a tela
  printf("Digite o primeiro valor:");
  scanf("%d", &V1);
  printf("Digite o segundo valor:");
  scanf("%d", &V2);

  // Chama a função e recebe o retorno
  resultado = multiplica(V1,V2);

  printf("Resultado = %d\n", resultado);

  system("PAUSE");  // pausa temporária até o pressionamento de tecla
  return 0;
}

```

## Resultado obtido:

```
Digite o primeiro valor:6
Digite o segundo valor:4
Resultado = 24
Pressione qualquer tecla para continuar. . .
```

# VETORES

Prog08.c

```c
/* Programa 08: Entrada de Dados no Vetor String */

#include<stdio.h>
#include<conio.h>
#include <string.h>  // arquivo de cabeçalho para trabalhar com strings

// Função principal
void main()
{
   char Nome[30];  // declara uma variável string

   system("CLS"); // Limpa  a tela

   // Uso da função de cópia de strings - strcpy( )
   strcpy(Nome, "Juan Gabriel");  // atribui "Juan Gabriel" para a variável Nome
   printf("O amigo %s foi armazenado\n\n", Nome);

   system("PAUSE");
   return(0);
}

```

**Resultado obtido:**

```
O amigo Juan Gabriel foi armazenado

Pressione qualquer tecla para continuar. . .
```

# Entrada de Dados em uma String . Neste exemplo, se digitar o nome completo, será impresso somente o primeiro nome

```
#include <stdio.h>
#include <conio.h>   // necessário para usar getch()
#include <stdlib.h>  // necessário para limpar a tela (CLS)
#include <locale.h>
#include <string.h>  // arquivo de cabeçalho para trabalhar com string

int main()
{
	char nome[100];
	printf ("Informe seu nome: ");
	scanf ("%s",nome); // não se usa o &
	printf ("Seu nome: %s\n",nome);
}
```

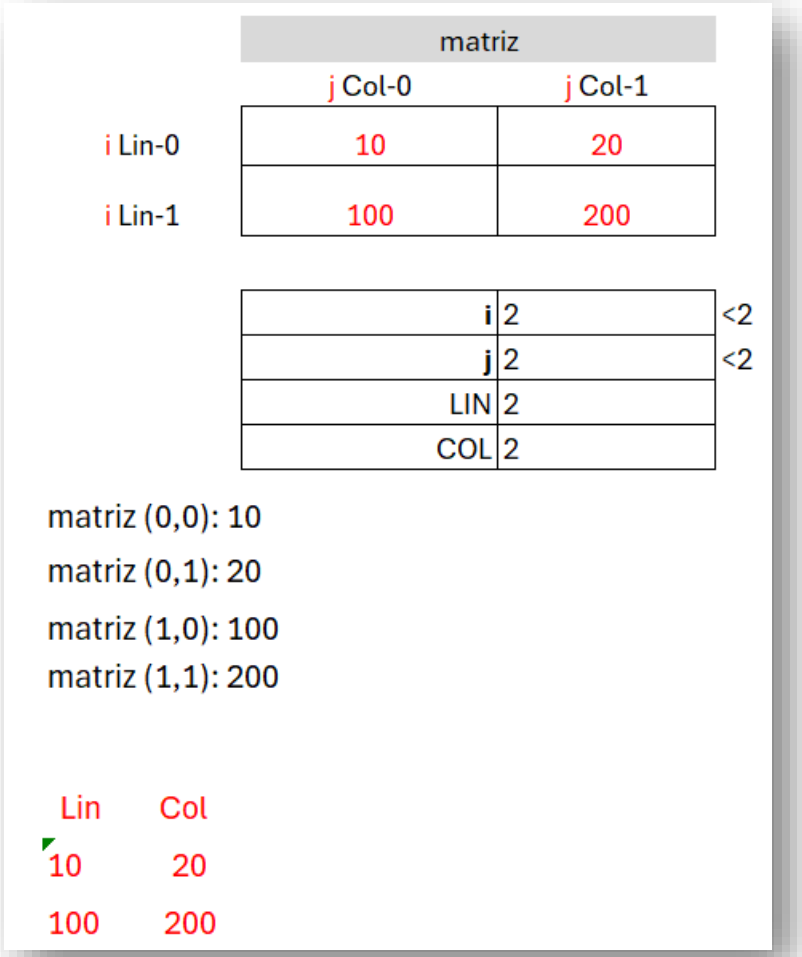
# Programa String_entrada: Entrada de Dados em uma String
# Neste exemplo, podemos digitar o nome completo.

```
#include <stdio.h>
#include <conio.h>   // necessário para usar getch()
#include <stdlib.h>  // necessário para limpar a tela (CLS)
#include <locale.h>
#include <string.h>  // arquivo de cabeçalho para trabalhar com string

int main()
{
	char nome[100];
	printf ("Informe seu nome: ");
	scanf (" %[^\n]",nome); // não se usa o &. [^\n] permite a entrada de string composta
	printf ("Seu nome: %s\n",nome);
}
```

# MATRIZES

## M09 - MATRIZ QUADRADA:

Programa para receber entrada em uma matriz com 2 linhas e duas colunas. Ao final, escrever os valores digitados em cada linha/coluna da matriz.



matriz

| | j Col-0 | j Col-1 |
|---|---|---|
| i Lin-0 | 10 | 20 |
| i Lin-1 | 100 | 200 |

| | | |
|---|---|---|
| i | 2 | <2 |
| j | 2 | <2 |
| LIN | 2 | |
| COL | 2 | |

matriz (0,0): 10

matriz (0,1): 20

matriz (1,0): 100

matriz (1,1): 200

| Lin | Col |
|---|---|
| 10 | 20 |
| 100 | 200 |

## Resultado obtido:

```
matriz(0,0): 10
matriz(0,1): 20
matriz(1,0): 100
matriz(1,1): 200

Lin   Col
  10   20
 100  200
```

```c
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
/* Definição de Constantes */
#define LIN 2
#define COL 2
int main(void)
{
	int matriz[LIN] [COL], i, j;

	//entrada de dados
	for(i=0; i<LIN; i++){
		for(j=0; j<COL; j++)
		{
			printf("matriz(%d,%d): ", i, j);
			scanf("%d", &matriz[i][j]);
		}
	}
	//saída de dados
	printf("\nLin   Col\n");
	for(i=0; i<LIN; i++)
	{
		for(j=0; j<COL; j++)
			printf("%4d", matriz[i][j]);
		printf("\n");
	}
	system("PAUSE");
	return(0);
}
```

# MATRIZES

**M10 - MATRIZ NÃO QUADRADA:**

Programa para receber entrada em uma matriz com 3 linhas e duas colunas. Ao final, escrever os valores digitados em cada linha/coluna da matriz.

| | | |
|---|---|---|
| i | 2 | <2 |
| j | 2 | <2 |
| LIN | 2 | |
| COL | 2 | |

matriz (0,0): 10
matriz (0,1): 20
matriz (1,0): 100
matriz (1,1): 200
matriz (2,0): 1000
matriz (2,1): 2000

| Lin | Col |
|---|---|
| 10 | 20 |
| 100 | 200 |
| 1000 | 2000 |

## Resultado obtido:

```
*** MATRIZ ***
matriz(0,0): 10
matriz(0,1): 20
matriz(1,0): 100
matriz(1,1): 200
matriz(2,0): 1000
matriz(2,1): 2000

Lin   Col
  10   20
 100  200
10002000
```

```c
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#define LIN 3   //3 linhas
#define COL 2   //2colunas
int main(void)
{
	int matriz[LIN][COL], i, j;
	printf("*** MATRIZ ***\n");
	//entrada de dados
	for(i=0; i<LIN; i++)
		for(j=0; j<COL; j++)
		{
			printf("matriz(%d,%d): ", i, j);
			scanf("%d", &matriz[i][j]);
		}
	//saída de dados
	printf("\nLin   Col\n");
	for(i=0; i<LIN; i++)
	{
		for(j=0; j<COL; j++)
		{
			printf("%4d", matriz[i][j]);
		}
		printf("\n");
	}
	system("PAUSE");
	return(0);
}
```

# VETORES

## V13 – VETOR – CÁLCULO IMCRADA:

Programa para ler idade, altura e peso de 3 pessoas.

Para cada uma delas, calcular o IMC.

Esse cálculo se dará por meio de uma função (imc).

Ao final, será informado: a idade, altura, peso e o IMC.

Fórmula para o Calculo do IMC: (peso/(altura*altura)).

***Classificação do IMC:***

Massa <=18.5: Abaixo do peso;

Massa >=25 e <25: Peso normal;

Massa >=25 e <30: Sobrepeso;

Massa >=30 e <35: Obesidade grau 1;

Massa >=35 e <40: Obesidade grau 2;

Massa >=40: Obesidade grau 3.

```c
/* Programa 13: Vetor heterogênio e Cálculo IMC */
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>

#define VET 3
int i;    //Variável Global
float imc(float p, float a) { //Função cálculo do índice do IMC
      float m;         //Variável local
      m=(p/(a*a));
      return m;     //retorna o cálculo  do IMC
}
int main(void){  //Método principal
      int idade[VET];
      float peso[VET], altura[VET], massa;

      for(i=0; i<3; i++) {
            system("CLS");  //limpa a tela
            printf("Digite sua idade: ");
            scanf("%d", &idade[i]);
            printf("Digite sua altura: ");
            scanf("%f", &altura[i]);
            printf("Digite seu peso: ");
            scanf("%f", &peso[i]);
            system ("CLS");
            printf("Idade: %d\n", idade[i]);
            printf("Altura: %1.2f\n", altura[i]);
            printf("Peso: %3.3f\n", peso[i]);

            massa = imc(peso[i], altura[i]);

            //Classifica o IMC
            if(massa<18.5)
                  printf("IMC %3.3f\tAbaixo do peso\n", massa);
                  else if (massa>=18.5 && massa<25)
                        printf("IMC %3.3f\tPeso normal\n", massa);
                  else if (massa>=25 && massa<30)
                        printf("IMC %3.3f\tSobrepeso\n", massa); //\t = Tab
                  else if (massa>=30 && massa<35)
                        printf("IMC %3.3f\tObesidade\n", massa);
                  else if (massa>=35 && massa<40)
                        printf("IMC %3.3f\tObesidade grau 2\n", massa);
                  else if (massa>=35 && massa<40)
                        printf("IMC %3.3f\tObesidade grau 2\n", massa);
                  else
                        printf("IMC %3.3f\tObesidade grau 3\n", massa);
            system("PAUSE");
      }
      return 0;
}
```

# VETORES

## /* Programa 13: Vetor heterogênio e Cálculo IMC */

| idade | | | |
|---|---|---|---|
| 18 | 20 | 30 | <-- valor do índice |
| 0 | 1 | 2 | <-- índice |

<--nome do vetor

| peso | | | |
|---|---|---|---|
| 77 | 74 | 98 | <-- valor do índice |
| 0 | 1 | 2 | <-- índice |

<--nome do vetor

| altura | | | |
|---|---|---|---|
| 1,74 | 1,9 | 1,8 | <-- valor do índice |
| 0 | 1 | 2 | <-- índice |

<--nome do vetor

| i | 3 |
|---|---|

| massa | 30,24691 |
|---|---|

| Entrada dados | |
|---|---|
| Idade: | 30 |
| altura: | 1,80 |
| peso: | 98,000 |

| Função imc | |
|---|---|
| parâmetros: | |
| p | 98 |
| a | 1,8 |
| | |
| m | 30,24691 |

| Saída: | |
|---|---|
| IMC 25,433 | Sobrepeso |
| IMC 20,499 | Peso normal |
| IMC 30,247 | Obesidade |

**Scanf()**

```
Digite sua idade: 30
Digite sua altura: 1.8
Digite seu peso: 98
```

**Printf()**

```
Idade: 30
Altura: 1.80
Peso: 98.000
IMC 30.247      Obesidade
```

# MATRIZES

**M12 – TRAÇO DA MATRIZ (diagonal):**
Programa para receber números e uma matriz quadrada com 3 linhas e 3 colunas. Ao final, deverá exibir os números inseridos na diagonal da matriz, bem como a soma desses números da diagonal. A entrada dos valores se dará dentro de uma função chamada "entrada". A filtragem dos valores da diagonal será feita dentro de uma função chamada "traço. A exibição dos valores da diagonal será feito dentro da função "exibir".

| | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 10 | 20 | 30 |
| 1 | 40 | 50 | 60 |
| 2 | 70 | 80 | 90 |

Prog12.c

```c
/* Programa 12: Exibe o Traço da Matriz  - Soma dos Elementos da Diagonal Princi

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

#define DIM 3

// Função entrada de dados
void entrada (int matriz[][DIM])
{
    int linha,
        coluna;

    for(linha=0 ; linha < DIM ; linha++)
        for(coluna=0 ; coluna < DIM ; coluna++)
        {
            printf("Entre com o elemento matriz[%d][%d]: ", linha+1, coluna+1);
            scanf("%d", &matriz[linha][coluna]);
        }
}

// Função saída de dados
void exibir (int matriz[][DIM])
{
    int linha,
        coluna;

    for(linha=0 ; linha < DIM ; linha++)
    {
        for(coluna=0 ; coluna < DIM ; coluna++)
            printf("%3d ", matriz[linha][coluna]);
        printf("\n");
    }
}

// Traço da Matriz
int traco (int matriz[][DIM])
{
    int count,
        traco=0;

    for(count=0 ; count < DIM ; count++)
        traco += matriz[count][count];
    return traco;
}
```

```c

// Função Principal
int main(void)

{
    int matriz[DIM][DIM];

    system("CLS");
    entrada (matriz);
    system("PAUSE");  // pausa temporária até o pressionamento de tecla

    system("CLS");
    exibir(matriz);
    system("PAUSE");  // pausa temporária até o pressionamento de tecla

    printf("\nTraco da matriz: %d\n", traco(matriz));
    return 0;

}
```

# MATRIZES

## Resultado obtido:

```
Entre com o elemento matriz[1][1]: 1
Entre com o elemento matriz[1][2]: 3
Entre com o elemento matriz[1][3]: 5
Entre com o elemento matriz[2][1]: 2
Entre com o elemento matriz[2][2]: 1
Entre com o elemento matriz[2][3]: 6
Entre com o elemento matriz[3][1]: 5
Entre com o elemento matriz[3][2]: 4
Entre com o elemento matriz[3][3]: 1
Pressione qualquer tecla para continuar. . .
```

```
  1   3   5
  2   1   6
  5   4   1
Pressione qualquer tecla para continuar. . .
```

# EXERCÍCIOS

❶ Escreva um algoritmo em linguagem C que permita ao usuário, a partir de sua escolha, preencher uma matriz quadrada de ordem *n*, com números inteiros, imprimindo, ao final do processamento, o produto referente aos elementos que pertençam a diagonal principal.

# EXERCÍCIOS

❷ Escreva um algoritmo em linguagem C que permita ao usuário, entrar com números inteiros, para preenchimento de uma matriz 3 x 5, imprimindo, ao final do processamento, todos os elementos de matriz, a partir do seguinte leiaute:

```
1   2   3   4   5
9   8   7   6   5
1   5   9   7   3
```

Gabarito

Gabarito

Gabarito

Gabarito

# GABARITO

❶ Escreva um algoritmo em linguagem C que permita ao usuário, a partir de sua escolha, preencher uma matriz quadrada de ordem *n*, com números inteiros, imprimindo, ao final do processamento, o produto referente aos elementos que pertençam a diagonal principal.

**UMA POSSÍVEL SOLUÇÃO**

Prog13.c

```c
/* Programa 13: Exercício 1 - Matriz Quadrada de Ordem n */

#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include <locale.h> // necessário para usar setlocale

// Função entrada de dados
void entrada (int dim, int matriz[][dim])
{
    int linha,
        coluna;

    for(linha=0 ; linha < dim ; linha++)
        for(coluna=0 ; coluna < dim ; coluna++)
        {
            printf("Entre com o elemento matriz[%d][%d]: ", linha+1, coluna+1);
            scanf("%d", &matriz[linha][coluna]);
        }
}

// Função saída de dados
void exibir (int dim, int matriz[][dim])
{
    int linha,
        coluna;

    for(linha=0 ; linha < dim ; linha++)
    {
        for(coluna=0 ; coluna < dim ; coluna++)
            printf("%3d ", matriz[linha][coluna]);
        printf("\n");
    }
}

```

# GABARITO

## SOLUÇÃO EXERCÍCIO ❶

[*] Prog13.c

```c

// Traço da Matriz
int traco (int dim, int matriz[][dim])
{
    int count,
        traco=0;

    for(count=0 ; count < dim ; count++)
        traco += matriz[count][count];
    return traco;
}

// Função Principal
int main(void)

{
    int tam;

    setlocale(LC_ALL, "Portuguese"); // Acentuação gráfica
    printf("Qual a dimensão da Matriz Quadrada? ");
    scanf("%d", &tam);

    int matriz[tam][tam];

    system("CLS");
    entrada (tam, matriz);
    system("PAUSE");  // pausa temporária até o pressionamento de tecla

    system("CLS");
    exibir(tam, matriz);
    system("PAUSE");  // pausa temporária até o pressionamento de tecla

    printf("\nTraco da matriz: %d\n", traco(tam, matriz));
    return 0;

}
```

# GABARITO

## SOLUÇÃO EXERCÍCIO ❶

# GABARITO

❷ Escreva um algoritmo em linguagem C que permita ao usuário, entrar com números inteiros, para preenchimento de uma matriz 3 x 5, imprimindo, ao final do processamento, todos os elementos da matriz, a partir do seguinte leiaute exemplo:

```
1  2  3  4  5
9  8  7  6  5
1  5  9  7  3
```

**UMA POSSÍVEL SOLUÇÃO**

# GABARITO

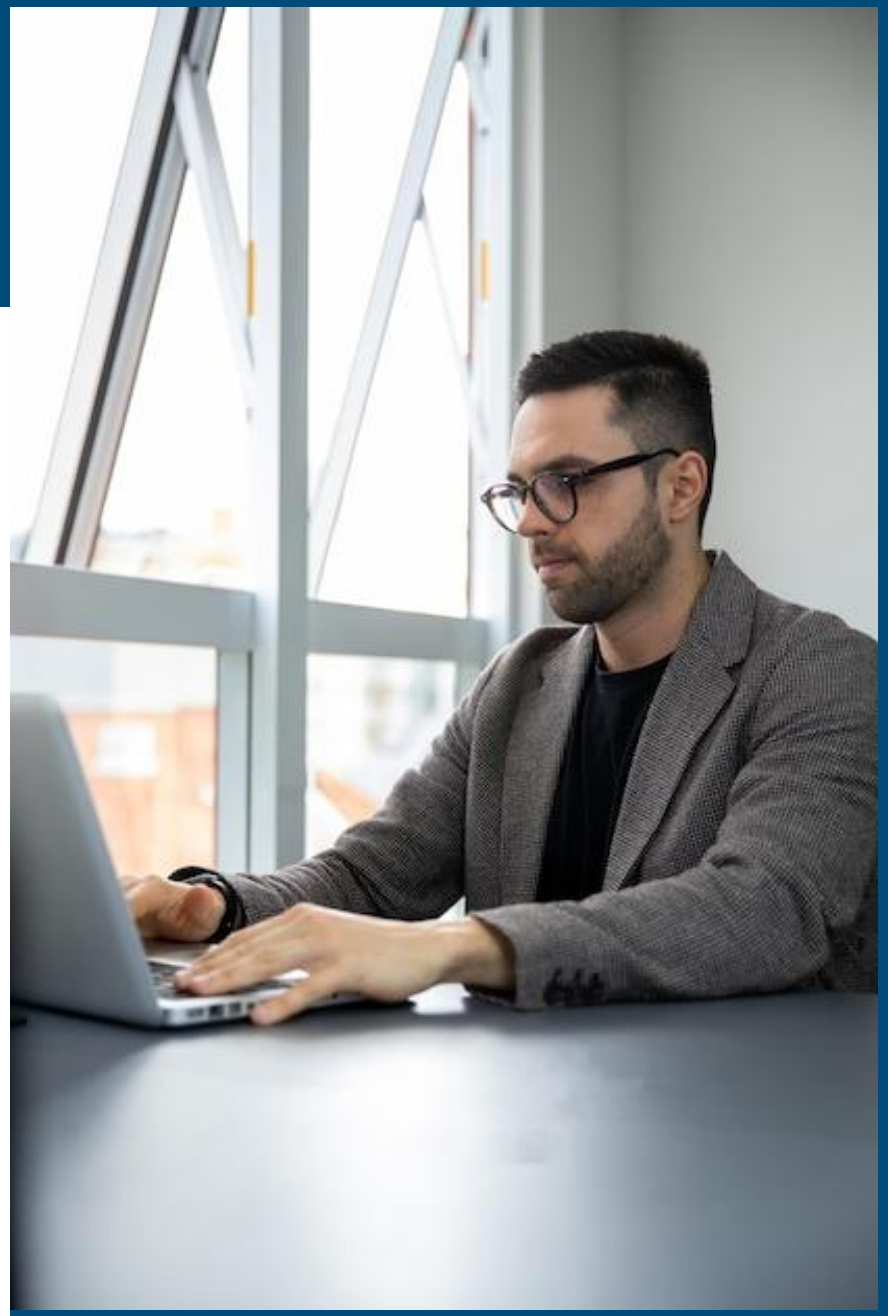
## SOLUÇÃO EXERCÍCIO ❷

Prog14.c

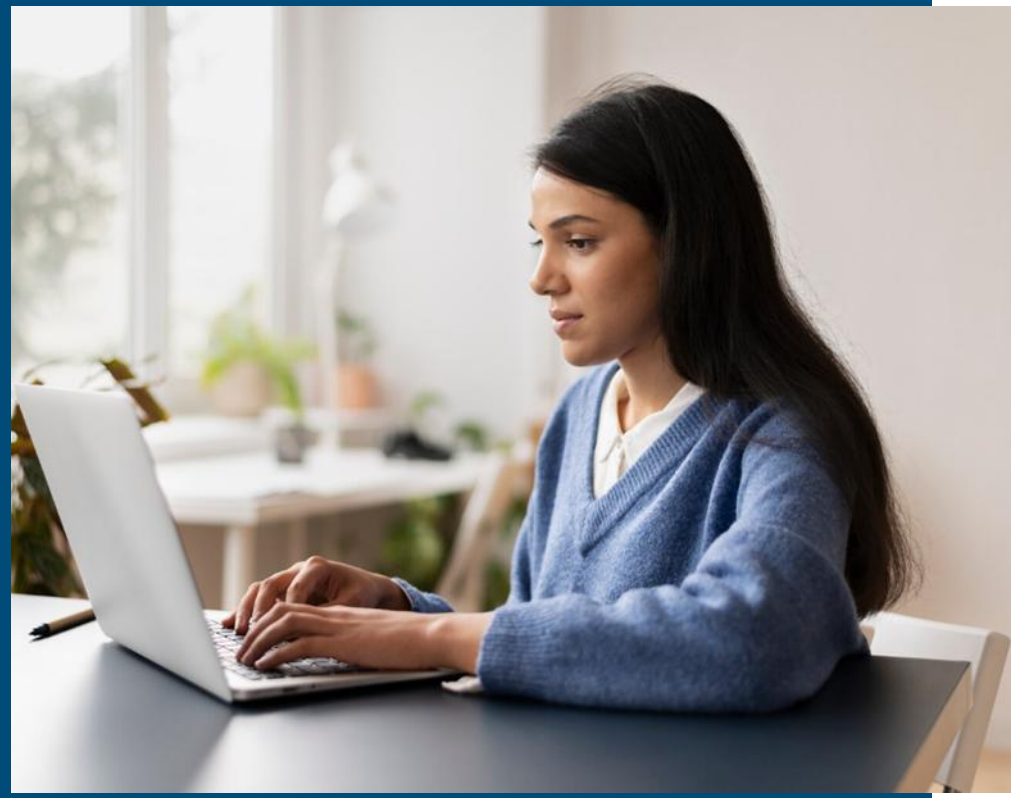
```c
/* Programa 14: Exercício 2 - Matriz 3 x 5 */

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

#define LIN 3
#define COL 5

// Função entrada de dados
void entrada (int matriz[LIN][COL])
{
    int linha,
        coluna;

    for(linha=0 ; linha < LIN ; linha++)
        for(coluna=0 ; coluna < COL ; coluna++)
        {
            printf("Entre com o elemento matriz[%d][%d]: ", linha+1, coluna+1);
            scanf("%d", &matriz[linha][coluna]);
        }
}

// Função saída de dados
void exibir (int matriz[LIN][COL])
{
    int linha,
        coluna;

    for(linha=0 ; linha < LIN ; linha++)
    {
        for(coluna=0 ; coluna < COL ; coluna++)
            printf("%3d ", matriz[linha][coluna]);
        printf("\n");
    }
}
```

```c

// Função Principal
int main(void)

{
    int matriz[LIN][COL];

    system("CLS");
    entrada (matriz);

    system("PAUSE");  // pau

    system("CLS");
    exibir(matriz);

    system("PAUSE");  // pau
    return 0;

}
```

# Laboratório de Práticas



# LABORATÓRIO DE PRÁTICAS



Proposta de Atividades Práticas para maximizar o aprendizado do estudante no estudo da disciplina...

## Algoritmos e
## Laboratório de Programação

# EXERCÍCIOS

FAÇA UM PROGRANA EM C PARA RECEBER DUAS NOTAS DE UMA TURMA COM 10 ALUNOS.

PARA CADA ALUNO, INFORMAR SUA MÉDIA E SUA SITUAÇÃO.
AO FINAL, INFORMAR A MÉDIA DA TURMA

# RESOLUÇÃO

```
/* Média do aluno e Media da Turma */
#include <stdio.h>
#include <conio.h>   // necessário para usar getch()
#include <stdlib.h>  // necessário para limpar a tela (CLS)
#include <locale.h>

int main(void) {
	setlocale(LC_ALL, "portuguese");
	float n1, n2, media, mediaturma=0;
	int i, tam=3;
	for (i=0; i<tam; i++)     {
		printf("\nPrimeira nota do aluno: ");
		scanf("%f",&n1);
		printf("\nSegunda nota do aluno: ");
		scanf("%f",&n2);
		media=(n1+n2)/2;
		printf("\nA média desse aluno foi: %.1f \n", media);

		mediaturma+=media;
		system("pause");
		system("CLS");
	}
	mediaturma=mediaturma/tam;
	printf("\n\nA MÉDIA DESSA TURMA FOI: %.1f \n\n", mediaturma);
	system("pause");
	return 0;
}
```

# EXERCÍCIOS

FAÇA UM PROGRAMA EM C PARA RECEBER DUAS NOTAS DE UMA TURMA COM 10 ALUNOS.
PARA CADA ALUNO, INFORMAR SUA MÉDIA E SUA SITUAÇÃO.
AO FINAL, INFORMAR A MÉDIA DA TURMA

SITUAÇÃO:

- MÉDIA MENOR QUE 5 => "REPROVADO"
- MÉDIA MAIOR OU IGUAL A 5 E MENOR QUE 6 => "EM RECUPERAÇÃO"
- MÉDIA MAIOR OU IGUAL A 6 => "APROVADO"

# RESOLUÇÃO

```c
#include <stdio.h>
#include <conio.h>   // necessário para usar getch()
#include <stdlib.h>  // necessário para limpar a tela (CLS)
#include <locale.h>

int main(void) {
	setlocale(LC_ALL,"portuguese");
	int i, tam=2;
	char nome[50];
	float n1, n2, media, mediaturma=0;

	for(i=0; i<tam; i++) {
		printf("\nPrimeira nota do %d. aluno: ", i+1);
		scanf("%f", &n1);

		printf("\nSegunda nota do %d. aluno: ", i+1);
		scanf("%f", &n2);

		media=(n1+n2)/2;
		printf("\n\nA média do %d. aluno foi: %.1f\n\n", i+1, media);
		mediaturma+= media;     //mediaturma = mediaturma + media;

		if (media < 5){
			printf("\nAluno REPROVADO!");
		} else  if(media < 6){
			printf("\nAluno em RECUPERAÇÃO!");
		} else
			printf("\nAluno A P R O V A D O ! ! !");
	}
	mediaturma=mediaturma/tam;
	printf("\n\nA média dessa turma é: %.1f\n\n",mediaturma);
	system("pause");
}
```

**/* Média do aluno, situação do aluno e  Media da Turma */**

# EXERCÍCIOS – VETOR PREÇO VENDA PRODUTO

Escreva um programa, na linguagem C, para ler o código, o nome e o preço de compra de 3 produtos. Ao final, informar o nome, o preço de compra e o preço de venda de cada produto, que será o preço de compra acrescido de 50%.
O programa deve usar vetores.

# RESOLUÇÃO

```c
#include <stdio.h>
#include <locale.h>

#define QTDEPROD 3
int main() {
   setlocale(LC_ALL, "portuguese");

   int codigo[QTDEPROD];
   char nome[QTDEPROD][50]; // Assume que o nome do produto tem no máximo 50 caracteres
   float preco_compra[QTDEPROD];
   float preco_venda[QTDEPROD];

   // Lendo os dados dos produtos
   printf(">>>ENTRADA - DADOS<<<\n");
   for (int i = 0; i<QTDEPROD; i++) {
        printf("\nCódigo do %do. produto: ", i+1);
        scanf("%d", &codigo[i]);

        printf("\nNome: ");
        scanf("%s", nome[i]);

        printf("\nPreço de compra: ");
        scanf("%f", &preco_compra[i]);

        printf("-------------------------");
   }
```

```c
   // Calculando o preço de venda e mostrando os resultados
   printf("\n\n>>>SAÍDA - INFORMAÇÕES<<<\n");
   printf("-----------------------------\n");
   for (int i = 0; i < QTDEPROD; i++) {
        preco_venda[i] = preco_compra[i] * 1.5; // Aumento de 50%

        printf("%do. produto: %s\n", i+1, nome[i]);
        //printf("Nome: %s\n", nome[i]);
        printf("Preço de compra: %.2f\n", preco_compra[i]);
        printf("Preço de venda: %.2f\n", preco_venda[i]);

        printf("---------------------------------\n");
   }

   return 0;
}
```

# REVISÃO DE BASES: ESTRUTURAS BÁSICAS MANIPULAÇÃO STRING

Prof. MSc. Lázaro Oliveira.

#ConhecimentoTransforma

Semana 02