



**UNIVERSIDADE FEDERAL DO PARÁ**  
**FACOMP/ICEN**

**Estruturas de Dados I – LISTA 4 – VETORES E ALOCAÇÃO DINÂMICA,  
MATRIZES E ALOCAÇÃO DINÂMICA**

**Os Exercícios de 1 a 8 são relativos ao capítulo 4**

1) Fazer um programa em C para calcular a soma dos 100 primeiros números inteiros usar vetores.  
2) Implemente um programa em C que calcula a média de um conjunto de 20 valores lidos de um vetor, sendo esses valores reais.  
3) Implemente um programa em C que calcula a variância de um conjunto de 20 valores lidos, sendo esses valores reais, a fórmula para o cálculo da variância é dado por 
$$v = \sum \left( \frac{(x - media)^2}{N} \right).$$

4) Implemente um programa em C, que lê dois vetores v e w e calcula o produto dos termos dos vetores armazenando o resultado em y, imprimir o produto entre os termos dos vetores (valores armazenados em y).

5) Implemente um programa em C que lê elementos inteiros e multiplica cada elemento pelo índice do vetor e imprime o vetor resultante.

6) Implemente um programa em C que calcule a média e a variância de um conjunto de 10 números reais modularizar as funcionalidades para o cálculo da média e da variância os protótipos das funções media e variancia devem ser :

float media(int n, float\* v) e float variancia (int n, float \*v, float m)

7) Implementar o exercício 6 utilizando alocação dinâmica de memória, sugestões: para fazer a alocação dinâmica use:

v=(float \*)malloc(n\*sizeof(float)) , testar para ver se existe memória suficiente para fazer a alocação if(v==NULL) {printf("Memória Insuficiente\n"); exit(1);} e para liberar o espaço de memória use free(v);

8) Implemente uma função em C que calcula o produto vetorial de dois vetores usar alocação dinâmica a função produto vetorial é mostrada a seguir:

```
float* prod_vetorial(float* u, float* v)
{
```

```
    float *p=(float*)malloc(3*sizeof(float));
    p[0]=u[1]*v[2]-v[1]*u[2];
    p[1]=u[2]*v[0]-v[2]*u[0];
    p[2]=u[0]*v[1]-v[0]*u[1];
    return p;
}
```

**Os exercícios 9 a 10 são relativos ao capítulo 5 Matrizes.**

9) Escreva um programa em C , que imprime os elementos de três matrizes usando alocação estática, conforme declarações e inicializações mostradas abaixo:

```
float mat[4][3]={ {1,2,3}, {4,5,6}, {7,8,9}, {10,11,12} };
```

```
float mat1[4][3]={1,2,3,4,5,6,7,8,9,10,11,12};
```

```
float mat2[][3]={1,2,3,4,5,6,7,8,9,10,11,12};
```

10) Implemente uma função para o exercício 9, que recebe a matriz mat e imprime seus elementos, sugestão para o protótipo da função void imprime(float(\*mat)[3]) ou void imprime1(float mat[ ][3]).

11) Implemente um programa em c que imprime os elementos da diagonal principal da matriz e calcula a soma dos mesmos. Onde mat é uma matriz da forma :float mat[4][3]={ {1,2,3}, {4,5,6}, {7,8,9} };

**Exercícios de adicionais do capítulo 4 – Vetores e Alocação Dinâmica**

12) Implemente a função negativos, que recebe como parâmetro um vetor de números de ponto flutuante (vet) de tamanho n e retorna quantos números negativos estão armazenados nesse vetor. Essa função deve obedecer ao protótipo:

**int negativos (int n, float \*vet)**

13) Implemente a função pares, que recebe como parâmetros um vetor de números inteiros (vet) de tamanho n e retorna quantos números pares estão armazenados nesse vetor. Essa função deve obedecer ao protótipo.

**int pares(int n,int \*vet)**

14) Implemente uma função avalia, que permite avaliação de polinômios. Cada polinômio é definido é definido por um vetor contendo coeficientes. Por exemplo, o polinômio  $3x^2+2x+12$ , terá um vetor de coeficientes igual a v[ ]={12,2,3}. A função avalia deve obedecer ao protótipo:

Prof.Lidio Campos



**UNIVERSIDADE FEDERAL DO PARÁ  
FACOMP/ICEN**

**Estruturas de Dados I – LISTA 4 – VETORES E ALOCAÇÃO DINÂMICA,  
MATRIZES E ALOCAÇÃO DINÂMICA**

**double avalia(double \*poli, int grau, double x)**

15) Implemente a função deriva, que calcula a derivada de um polinômio. Cada polinômio é definido por um vetor contendo seus coeficientes. Por exemplo, o polinômio de grau 2,  $3x^2+2x+12$ , terá um vetor de coeficientes igual a  $v[] = \{12, 2, 3\}$ . A função deriva deve obedecer ao protótipo:

**double avalia(double \*poli, int grau, double \*out)**

16) Implemente a função max\_vet, que recebe como parâmetro um vetor de números de ponto flutuante (vet) de tamanho n e retorna o maior número armazenado nesse vetor. Essa função deve obedecer ao protótipo:

**float max\_vet(int n, float \*vet)**

17) Implemente a função busca, que recebe como parâmetro um vetor de números de ponto flutuante (vet) de tamanho n e um valor x. A função deve retornar q se x pertence a esse vetor e 0 caso contrário. Essa função deve obedecer ao protótipo:

**int busca(int n, int \*vet, int x)**

**Exercícios de adicionais do capítulo 5 – Matrizes e Alocação Dinâmica**

18) Implemente uma função que indique se uma matriz quadrada de números inteiros é uma matriz identidade ou não. A função deve retornar 1 se a matriz for uma matriz identidade e caso contrário. A função recebe como parâmetros a matriz de inteiros, usando a representação de matrizes através de vetores simples e um número n, indicando a dimensão da matriz. Essa função deve obedecer ao protótipo:

**int matriz\_identidade(int \*mat, int n)**

19) Implemente uma função que calcule a transposta de uma matriz mat. A função tem como valor de retorno o ponteiro do vetor que representa a matriz transposta criada. A implementação dessa função deve ser dada por:

**float \* transposta (int m, int n, float\* mat) OBS: utilizar vetor simples**

20) Implemente uma função que calcule a transposta de uma matriz mat. A função tem como valor de retorno o ponteiro do vetor que representa a matriz transposta criada. A implementação dessa função deve ser dada por:

**float \* transposta (int m, int n, float\*\* mat) OBS: utilizar vetor de ponteiros**