Introdução a Programação COLTEC/UFMG
Professores: João Eduardo Montandon e Virgínia Fernandes Mota
Turmas: 103 e 106
**Lista preparatória para a terceira prova** (Data de entrega: 29/06)

## Parte I - Matrizes

Desenvolver os respectivos programas em C para resolver os problemas abaixo:

1. Faça uma função que recebe, por parâmetro, uma matriz A(5,5) e retorna a soma dos seus elementos.

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

int soma_elementos(int matriz[5][5]){

    int soma = 0;
    int i, j;

    for(i=0; i<5; i++){
        for(j=0; j<5; j++){
            soma = soma + matriz[i][j];
        }
    }

    return soma;
}

int main()
{
    int matriz[5][5];
    int i, j;
    int soma;

    for(i=0; i<5; i++){
        for(j=0; j<5; j++){
            printf("\nDigite o elemento da linha %d e coluna %d ",i,j);
            scanf("%d",&matriz[i][j]);
        }
    }

    soma = soma_elementos(matriz);

    printf("\nA soma eh %d",soma);

    return 0;
}
```

2. Faça uma função que recebe, por parâmetro, uma matriz A(6,6) e retorna a soma dos elementos da sua diagonal principal e da sua diagonal secundária.

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

int soma_elementos_diagonal(int matriz[6][6]){

    int soma = 0;
    int i, j;

    for(i=0; i<6; i++){
        for(j=0; j<6; j++){
            if(i==j || i + j == 5)
                soma = soma + matriz[i][j];
        }
    }

    return soma;
}

```

```c
int main()
{
    int matriz[6][6];
    int i, j;
    int soma;

    for(i=0; i<6; i++){
        for(j=0; j<6; j++){
            printf("\nDigite o elemento da linha %d e coluna %d ",i,j);
            scanf("%d",&matriz[i][j]);
        }
    }

    soma = soma_elementos_diagonal(matriz);

    printf("\nA soma eh %d",soma);

    return 0;
}
```

3. Faça uma função que recebe, por parâmetro, uma matriz A(7,6) e retorna a soma dos elementos da linha 5 e da coluna 3.

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

int soma_elementos_linha3_coluna5(int matriz[7][6]){

    int soma = 0;
    int i, j;

    for(i=0; i<7; i++){
        for(j=0; j<6; j++){
            if(i==4 || j == 2){
                soma = soma + matriz[i][j];
            }
        }
    }

    return soma;
}

int main()
{
    int matriz[7][6];
    int i, j;
    int soma;

    for(i=0; i<7; i++){
        for(j=0; j<6; j++){
            printf("\nDigite o elemento da linha %d e coluna %d ",i,j);
            scanf("%d",&matriz[i][j]);
        }
    }

    soma = soma_elementos_linha3_coluna5(matriz);

    printf("\nA soma eh %d",soma);

    return 0;
}
```

4. Faça uma função que recebe, por parâmetro, uma matriz A(6,6) e retorna o menor elemento da sua diagonal secundária.

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

```c

int menor_elemento_diagonal_secundaria(int matriz[6][6]){

    int menor = matriz[0][5];
    int i, j;

    for(i=0; i<6; i++){
        for(j=0; j<6; j++){
            if(i + j == 5 && matriz[i][j] < menor){
                menor = matriz[i][j];
            }
        }
    }

    return menor;
}

int main()
{
    int matriz[6][6];
    int i, j;
    int menor;

    for(i=0; i<6; i++){
        for(j=0; j<6; j++){
            printf("\nDigite o elemento da linha %d e coluna %d ",i,j);
            scanf("%d",&matriz[i][j]);
        }
    }

    menor = menor_elemento_diagonal_secundaria(matriz);

    printf("\nO menor elemento elemendo da diagonal secundaria eh %d",menor);

    return 0;
}
```

5. Faça um procedimento que receba duas matrizes de dimensões especificadas pelo usuário e retorne a multiplicação de uma pela outra, quando possível.

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

float **mult(int l1, int c1 , int l2, int c2, int **a, int **b, int **mr)
{
   int i, j , v;

   for (i = 0 ; i < l1; i++ )
      for (j = 0; j < c2; j++)
         for (v = 0; v < l2; v++)
            mr[i][j] = mr[i][j] + a[i][v] * b[v][j];

   return(mr);
}

int main()
{
    int **matriz1, **matriz2, **matriz;
    int i, j;
    int l1, c1, l2, c2;
    int soma;

    printf("Digite o numero de linhas da primeira matriz ");
    scanf("%d", &l1);

    printf("Digite o numero de colunas da primeira matriz ");
    scanf("%d", &c1);

    printf("Digite o numero de linhas da segunda matriz ");
```

```
30      scanf("%d", &l2);
31
32      printf("Digite o numero de colunas da segunda matriz ");
33      scanf("%d", &c2);
34
35      if(c1 != l2){
36          printf("\nNao eh possivel multiplicar as matrizes. Dimensoes incompativeis.");
37      }
38      else{
39
40          matriz1 = (int**)malloc(l1*sizeof(int*));
41          for(i=0; i<l1; i++)
42              matriz1[i] = (int*)malloc(c1*sizeof(int));
43
44          matriz2 = (int**)malloc(l2*sizeof(int*));
45          for(i=0; i<l2; i++)
46              matriz2[i] = (int*)malloc(c2*sizeof(int));
47
48          matriz = (int**)malloc(l1*sizeof(int*));
49          for(i=0; i<l1; i++)
50              matriz[i] = (int*)malloc(c2*sizeof(int));
51
52          for(i=0; i<l1; i++){
53              for(j=0; j<c1; j++){
54                  printf("\nDigite o elemento da linha %d e coluna %d da primeira matriz ",
                        i,j);
55                  scanf("%d",&matriz1[i][j]);
56              }
57          }
58
59          for(i=0; i<l2; i++){
60              for(j=0; j<c2; j++){
61                  printf("\nDigite o elemento da linha %d e coluna %d da segunda matriz ",i
                        ,j);
62                  scanf("%d",&matriz2[i][j]);
63              }
64          }
65
66          for(i=0; i<l1; i++){
67              for(j=0; j<c2; j++)
68                  matriz[i][j] = 0;
69          }
70
71          matriz = mult(l1,c1,l2,c2,matriz1,matriz2,matriz);
72
73          printf("\nMatriz resultado\n");
74          for(i=0; i<l1; i++){
75              for(j=0; j<c2; j++){
76                  printf("%d ",matriz[i][j]);
77              }
78              printf("\n");
79          }
80
81      }
82
83      return 0;
84  }
```

**Parte II - Estruturas**

Desenvolver os respectivos programas em C para resolver os problemas abaixo:

6. Crie uma estrutura ponto que representará um **ponto** no espaço. Essa estrutura conterá três números reais (x, y e z). Faça um programa que crie um vetor de estruturas com n pontos (especificados pelo usuário). Crie funções para:

a) Ler os 5 Pontos;

b) Imprimir os pontos lidos;

c) Calcular a distância de 2 pontos pelo seus indices;

d) Calcular a distância de todos os pontos consecutivos (primeiro com o segundo, segundo com o terceiro etc);

Ao final, crie um menu que permita selecionar cada uma das funções acima.

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

typedef struct{

    float x, y, z;

}ponto;

void le_cinco_pontos(ponto *n_pontos, int n){

    int i;

    for(i=0; i<n; i++){
        printf("\nDigite o x do ponto %d ",i+1);
        scanf("%f",&n_pontos[i].x);
        printf("\nDigite o y do ponto %d ",i+1);
        scanf("%f",&n_pontos[i].y);
        printf("\nDigite o z do ponto %d ",i+1);
        scanf("%f",&n_pontos[i].z);
    }
}

void imprime_pontos(ponto *n_pontos, int n){

    int i;
    for(i=0; i<n; i++)
        printf("Ponto %d - (%f, %f, %f)\n",i+1,n_pontos[i].x,n_pontos[i].y,n_pontos[i].z)
            ;
}

void distancia(ponto *n_pontos, int n){

    int i, p1, p2;
    float d;
    float x, y, z;

    printf("\nDigite o indice do ponto 1 ");
    scanf("%d",&p1);

    printf("\nDigite o indice do ponto 2 ");
    scanf("%d",&p2);

    x = n_pontos[p1].x - n_pontos[p2].x;
    y = n_pontos[p1].y - n_pontos[p2].y;
    z = n_pontos[p1].z - n_pontos[p2].z;

    d = sqrt(x*x+y*y+z*z);

     printf("\nA distancia entre os pontos %d e %d eh %f\n",p1,p2,d);

}

void distancia_consecutiva(ponto *n_pontos, int n){

    int i, j;
    float x,y,z,d;
    for(i=0; i < n-1; i++){

        x = n_pontos[i].x - n_pontos[i+1].x;
        y = n_pontos[i].y - n_pontos[i+1].y;
        z = n_pontos[i].z - n_pontos[i+1].z;

```

```
64          d = sqrt(x*x+y*y+z*z);
65
66          printf("\nA distancia entre os pontos %d e %d eh %f\n",i,i+1,d);
67      }
68
69  }
70
71  int main()
72  {
73      ponto *n_pontos;
74      int i, j, n, menu = 1;
75
76      while(menu != 0){
77          printf("\nEscolha uma opcao\n\n1 - Ler 5 pontos\n2 - Imprimir pontos\n3 -
                Calcular a distancia entre dois pontos\n4 - Calcular a distancia consecutiva
                entre os pontos\n0 - Sair\n\nATENCAO - As opcoes 2, 3 e 4 requerem que a
                opcao 1 seja executada anteriormente\n");
78          scanf("%d",&menu);
79          printf("\n");
80
81          system("cls");
82
83          if(menu == 1){
84              n = 5;
85              n_pontos = (ponto*)malloc(n*sizeof(ponto));
86              le_cinco_pontos(n_pontos,n);
87          }
88          else if(menu == 2)
89              imprime_pontos(n_pontos,n);
90          else if (menu == 3)
91              distancia(n_pontos,n);
92          else if (menu == 4)
93              distancia_consecutiva(n_pontos,n);
94      }
95
96      return 0;
97  }
```

7. Faça um programa para gerenciar uma biblioteca. Para tal, crie uma estrutura livro que conterá os campos título, autor, editora, ano e um inteiro chamado emprestado que servirá para controlar se o livro está na biblioteca (valor 1) ou se está emprestado (valor 0). Crie também um campo codigo que receberá um valor inteiro positivo único representando o código de cada livro. Crie na função principal (main) um vetor de estruturas com capacidade para 10 livros. Inicialmente faça um loop e atribua -1 ao campo código em todos os livros do vetor. O valor -1 representará uma posição disponível no vetor. O programa deverá fornecer um menu com as seguintes opções:

a) Incluir livro (nessa opção serão lidos todos os campos de um livro na primeira posição disponível do vetor);

b) Listar livros (Exibe todos os livros cadastrados);

c) Empresta livro (pede o código do livro e caso esteja na biblioteca, executa o empréstimo)

d) Recebe livro (altera o campo emprestado de zero para um. O acesso do livro é feito através de seu código).

e) Lista livros emprestados.

f) Lista livros disponíveis para empréstimo.

Todas as opções acima serão implementadas em funções apropriadas.
Extra: Existe uma estrutura de dados chamada lista que facilita bastante a criação deste programa.

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <math.h>
```

```c

typedef struct{

    char titulo[20];
    char autor[20];
    char editora[20];
    int ano;
    int emprestado;
    int codigo;

}livro;

void incluir_livro(livro *livros, int n){

    int i;
    for(i=0;i<n;i++){
        if(livros[i].codigo == -1){
            printf("Digite o nome do livro ");
            gets(livros[i].titulo);
            printf("Digite o autor do livro ");
            gets(livros[i].autor);
            printf("Digite a editora do livro ");
            gets(livros[i].editora);
            printf("Digite o ano do livro ");
            scanf("%d",&livros[i].ano);
            livros[i].emprestado = 1;
            printf("Digite o codigo do livro ");
            scanf("%d",&livros[i].codigo);
            return;
        }
    }

    printf("Biblioteca cheia.");
}

void listar_livros(livro *livros,int n){

    int i;

    for(i=0;i<n;i++){
        if(livros[i].codigo != -1){
            printf("\n\nLivro de codigo %d\n",livros[i].codigo);
            printf("\nTitulo - %s",livros[i].titulo);
            printf("\nAutor - %s",livros[i].autor);
            printf("\nEditora - %s",livros[i].editora);
            printf("\nAno - %d",livros[i].ano);
            if(livros[i].emprestado == 0)
                printf("\nEmprestado - Sim");
            else
                printf("\nEmprestado - Nao");
            printf("\n---------------------- ");
        }
    }
}

void receber_livro(livro *livros, int n){

    int codigo, i;

    printf("\nDigite o codigo do livro a ser recebido ");
    scanf("%d",&codigo);

    for(i=0; i<n; i++){
        if(livros[i].codigo == codigo){
            livros[i].emprestado = 1;
            printf("\nLivro devolvido com sucesso!\n");
            return;
        }
    }

```

```
74      printf("\nCodigo nao encontrado. Tente novamente.\n");
75      return;
76  }
77
78  void emprestar_livro(livro *livros, int n){
79
80      int codigo, i;
81
82      printf("\nDigite o codigo do livro a ser emprestado ");
83      scanf("%d",&codigo);
84
85      for(i=0; i<n; i++){
86          if(livros[i].codigo == codigo){
87              if(livros[i].emprestado == 1){
88                  livros[i].emprestado = 0;
89                  printf("\nLivro emprestado com sucesso!\n");
90              }
91              else
92                  printf("Livro ja esta emprestado.");
93              return;
94          }
95      }
96
97      printf("\nCodigo nao encontrado. Tente novamente.\n");
98      return;
99  }
100
101 void listar_livros_empretados(livro *livros,int n){
102
103     int i;
104
105     printf("\n\nLIVROS EMPRESTADOS\n\n");
106
107     for(i=0;i<n;i++){
108         if(livros[i].codigo != -1 && livros[i].emprestado == 0){
109             printf("\n\nLivro de codigo %d\n",livros[i].codigo);
110             printf("\nTitulo - %s",livros[i].titulo);
111             printf("\nAutor - %s",livros[i].autor);
112             printf("\nEditora - %s",livros[i].editora);
113             printf("\nAno - %d",livros[i].ano);
114             if(livros[i].emprestado == 0)
115                 printf("\nEmprestado - Sim");
116             else
117                 printf("\nEmprestado - Nao");
118             printf("\n---------------------- ");
119         }
120     }
121 }
122
123 void listar_livros_nao_empretados(livro *livros,int n){
124
125     int i;
126
127     printf("\n\nLIVROS NAO EMPRESTADOS\n\n");
128
129     for(i=0;i<n;i++){
130         if(livros[i].codigo != -1 && livros[i].emprestado == 1){
131             printf("\n\nLivro de codigo %d\n",livros[i].codigo);
132             printf("\nTitulo - %s",livros[i].titulo);
133             printf("\nAutor - %s",livros[i].autor);
134             printf("\nEditora - %s",livros[i].editora);
135             printf("\nAno - %d",livros[i].ano);
136             if(livros[i].emprestado == 0)
137                 printf("\nEmprestado - Sim");
138             else
139                 printf("\nEmprestado - Nao");
140             printf("\n---------------------- ");
141         }
142     }
143 }
```

```c

int main()
{
    livro *livros;
    int i, j, menu;
    char lixo[10];

    int n = 10;

    livros = (livro*)malloc(n*sizeof(livro));

    for(i=0; i<n; i++)
        livros[i].codigo = -1;

    do{
        printf("\n\nEscolha uma opcao\n\n1 - Incluir Livro");
        printf("\n2 - Listar todos os livro\n3 - Receber livro\n4 - Emprestar livro");
        printf("\n5 - Listar livros emprestados\n6 - Listar livros diponiveis\n0 - Sair\n
            \n");
        scanf("%d",&menu);
        gets(lixo);
        printf("\n");

        system("cls");

        if(menu == 1){
           incluir_livro(livros,n);
        }
        else if(menu == 2)
            listar_livros(livros,n);
        else if (menu == 3)
            receber_livro(livros,n);
        else if (menu == 4)
            emprestar_livro(livros,n);
        else if (menu == 5)
            listar_livros_empretados(livros,n);
        else if (menu == 6)
            listar_livros_nao_empretados(livros,n);
    }while(menu != 0);

    return 0;
}
```

**Parte III - Recursividade**

Desenvolver os respectivos programas em C para resolver os problemas abaixo:

8. Faça uma função recursiva para exibir na tela de 1 a 10.

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

void uma_a_dez(int n){

    if(n < 10){
        printf("%d ",n);
        uma_a_dez(n+1);
    }
    else
        printf("%d",n);

}

int main()
{
    uma_a_dez(1);
    return 0;
}
```

9. Faça uma função recursiva para exibir na tela os números ímpares de 1 a 30.

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

void impares(int n){

    if(n < 30){
        if(n%2 != 0)
            printf("%d ",n);

        impares(n+1);
    }

}

int main()
{
    impares(1);
    return 0;
}
```

10. Faça uma função recursiva para exibir os números compreendidos entre a e b, sendo a e b parâmetros da função. Se b for menor que a, exiba uma mensagem de erro.

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

void recursao(int n, int b){

    if(n < b){
        printf("%d ",n);
        recursao(n+1,b);
    }
}

void valores(int a, int b){

    int n;

    if(b < a)
        printf("Erro! b < a!");
    else{
        n = a+1;
        recursao(n,b);
    }
}

int main()
{
    int a, b;

    printf("Digite o valor de a ");
    scanf("%d",&a);

    printf("Digite o valor de b ");
    scanf("%d",&b);

    valores(a,b);

    return 0;
}
```

11. Pesquise a sequência de Fibonacci e faça uma função recursiva para exibir seus n primeiros termos, sendo n um parâmetro da função.

```c
#include <stdio.h>

int fibonacci(int num)
{
    if(num==1 || num==2)
        return 1;
    else
        return fibonacci(num-1) + fibonacci(num-2);
}

int main()
{
    int n,i;
    printf("Digite a quantidade de termos da sequencia de Fibonacci: ");
    scanf("%d", &n);
    printf("\nA sequencia de Fibonacci e: \n");
    for(i=0; i<n; i++)
        printf("%d ", fibonacci(i+1));
    printf("\n") ;
    return 0;
}
```

12. Faça uma função recursiva que receba um vetor de números reais e seu tamanho e retorne a soma de seus elementos.

```c
#include <stdio.h>

void soma(float *vetor, int n, float *somatorio,int i){

    if(i < n){
        *somatorio = *somatorio + vetor[i];
        soma(vetor,n,somatorio,i+1);
    }
}

int main()
{
    int n,i;
    float *vetor;
    float somatorio = 0;

    printf("Digite o tamanho do vetor ");
    scanf("%d", &n);

    vetor = (float*)malloc(sizeof(float));

    for(i=0; i<n; i++){
        printf("Digite o elemento %d do vetor ",i+1);
        scanf("%f",&vetor[i]);
    }

    soma(vetor,n,&somatorio,0);

    printf("\nA soma dos elementos eh %f\n", somatorio);

    return 0;
}
```

13. Faça uma função recursiva que receba um vetor de números reais e seu tamanho e retorne o menor elemento deste vetor.

```c
#include <stdio.h>

void encontra_menor(float *vetor, int n, int i, float *menor){
```

```
4
5       if(i < n){
6           if(vetor[i] < *menor){
7               *menor = vetor[i];
8           }
9           encontra_menor(vetor,n,i+1,menor);
10      }
11  }
12
13  int main()
14  {
15      int n,i;
16      float *vetor;
17      float menor;
18
19      printf("Digite o tamanho do vetor ");
20      scanf("%d", &n);
21
22      vetor = (float*)malloc(sizeof(float));
23
24      for(i=0; i<n; i++){
25          printf("Digite o elemento %d do vetor ",i+1);
26          scanf("%f",&vetor[i]);
27      }
28
29      menor = vetor[0];
30      encontra_menor(vetor,n,1,&menor);
31
32      printf("\nO menor elemento do vetor eh %f\n", menor);
33
34      return 0;
35  }
```