

UTFPR – Universidade Tecnológica Federal do Paraná.

Disciplina: **Computação 2 (IF62A)** –Turmas: **S41-S42** - Prof. **João Alberto Fabro**

Prova 1

Aluno: _____ Data: _____

Obs.: A interpretação das questões faz parte da prova.

1) O Fatorial de um número N é formado pela multiplicação do número N pelo Fatorial do número N-1. O Fatorial de 0 (zero) vale 1. Não existe fatorial de número negativo. Crie uma função que retorne o fatorial do número N, ou -1 se o número N passado por parâmetro for negativo. (10 pontos)

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

//O Fatorial de um número N é formado pela multiplicação de N
//pelo Fatorial do número N-1.
//O Fatorial de 0 (zero) vale 1.
//Não existe fatorial de número negativo.
//Crie uma função que retorne o fatorial do número N,
//ou -1 se o número N passado por parâmetro for negativo.
int fatorial(int N)
{
    if(N<0) return(-1);
    if(N==0) return(1);
    return( N * fatorial(N-1) );
}

int fatorial2(int N)
{
    int i, res=1;
    if(N<0) return(-1);
    for(i=N;i>0;i--)
        res = res * i;
    return(res);
}

int main()
{
    int j;
    scanf("%d", &j);
    printf("%d %d", fatorial(j), fatorial2(j));
    system("pause");
}
```

2) Escreva um programa capaz de calcular a multiplicação de duas matrizes de números inteiros de tamanho MÁXIMO 10x10. Este programa deve possuir as seguintes funções:

```
void le_matriz(int M[10][10], int numlinhas, int numcolunas);
int mult_matrizes(int M1[10][10], int M2[10][10], int n1, int nc1, int n2, int nc2);
void imprime_matriz(int M[10][10], int numlinhas, int numcolunas);
```

O programa principal (int main()) deve perguntar ao usuário o número de linhas e colunas da primeira matriz, em seguida o número de linhas e colunas da segunda matriz, e só permitir o cálculo se as matrizes forem compatíveis (número de colunas da primeira igual ao número de linhas da segunda). (20 pontos)

Exemplo: $\begin{vmatrix} 11 & 2 & 3 \\ 4 & 5 & 6 \end{vmatrix} \times \begin{vmatrix} 6 & 3 \\ 15 & 2 \\ 4 & 1 \end{vmatrix} = \begin{vmatrix} 6+10+12=28 & 3+4+3=10 \\ 24+25+24=73 & 12+10+6=28 \end{vmatrix}$

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

int M1[10][10], M2[10][10], MULT[10][10];

//Escreva um programa capaz de calcular a multiplicação de duas matrizes
//de números inteiros de tamanho MÁXIMO 10x10.
//Este programa deve possuir as seguintes funções:
    void le_matriz(int M[10][10], int numlinhas, int numcolunas);
    int mult_matrizes(int M1[10][10], int M2[10][10], int n1, int nc1,
int n2, int nc2, int MULT[10][10]);
    void imprime_matriz(int M[10][10], int numlinhas, int numcolunas);

//O programa principal (int main()) deve perguntar ao usuário o número
//de linhas e colunas da primeira matriz, em seguida o número de linhas
//e colunas da segunda matriz, e só permitir o cálculo se as matrizes
//forem compatíveis (número de colunas da primeira igual
//ao número de linhas da segunda). (20 pontos)

int main()
{
    int n1, nc1, n2, nc2;

    do
    {
        printf("Digite o numero de linhas da matriz 1(0 a 9):");
        scanf("%d", &n1);
        printf("Digite o numero de colunas da matriz 1(0 a 9):");
        scanf("%d", &nc1);

        printf("Digite o numero de linhas da matriz 2(0 a 9):");
        scanf("%d", &n2);
        printf("Digite o numero de colunas da matriz 2(0 a 9):");
        scanf("%d", &nc2);
    }while (nc1>=10 || n1>=10 || nc2>=10 || n2>=10);

    if(nc1==n2)
    {
        le_matriz(M1, n1, nc1);
        le_matriz(M2, n2, nc2);
        mult_matrizes(M1, M2, n1, nc1, n2, nc2, MULT);
    }
}
```

```

        printf("As matrizes sao:\n");
        imprime_matriz(M1, nl1, nc1);
        printf("\n");
        imprime_matriz(M2, nl2, nc2);
        printf("E a multiplicacao das duas matrizes vale;\n");
        imprime_matriz(MULT, nl1, nc2);
    }
    else
        printf("Matrizes com tamanho incompativel!\n");

    system("pause");
}

```

```

void le_matriz(int M[10][10], int numlinhas, int numcolunas)
{
    int i, j;
    for(i=0; i<numlinhas; i++)
        for(j=0; j<numcolunas; j++)
            {
                printf("Valor %d %d = ", i, j);
                scanf("%d", &M[i][j]);
            }
}

```

```

int mult_matrizes(int M1[10][10], int M2[10][10], int nl1, int nc1, int
nl2, int nc2, int M[10][10])
{
    int i, j, k, soma;

    for(i=0; i<nl1; i++)
        for(j=0; j<nc2; j++)
            {
                soma = 0;
                for(k=0; k<nc1; k++)
                    {
                        soma = soma + M1[i][k]*M2[k][j];
                    }
                MULT[i][j]=soma;
            }
}

```

```

void imprime_matriz(int M[10][10], int numlinhas, int numcolunas)
{
    int i, j;
    for(i=0; i<numlinhas; i++)
        {
            for(j=0; j<numcolunas; j++)
                {
                    printf("%8d ", M[i][j]);
                }
            printf("\n");
        }
}

```

3) Escreva uma função que receba um vetor de ponteiros para char, onde cada elemento deste vetor aponta para uma linha de uma matriz de caracteres de 50x50, onde estão armazenados 50 nomes de até 49 caracteres cada, e ordene este vetor alfabeticamente - função **void ordena_alfabeticamente(char *nomes[50])**. Esta função deve mudar apenas os ponteiros de lugar, ao invés de copiar as strings. (20 pontos)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define NUMNOMES 5

void troca(char **nome1, char **nome2)
{
    char *temp;
    temp=*nome1;
    *nome1=*nome2;
    *nome2=temp;
}

void ordena_alfabeticamente(char *nomes[52])
{
    int i,j;

    for(i=0;i<NUMNOMES;i++)
        for(j=0;j<NUMNOMES-1;j++)
            if(strcmp(nomes[j],nomes[j+1])>0)
                troca(&nomes[j], &nomes[j+1]);
}

void ordena_alfabeticamente2(char *nomes[52])
{
    int i,j;
    char *temp;

    for(i=0;i<NUMNOMES;i++)
        for(j=0;j<NUMNOMES-1;j++)
            if(strcmp(nomes[j],nomes[j+1])>0)
                {
                // troca(&nomes[j], &nomes[j+1]);
                temp = nomes[j];
                nomes[j] = nomes[j+1];
                nomes[j+1] = temp;
                }
}

int main()
{
```

```
char nomes[52][51];
char *vetorponteiros[52];
int i;
printf("Digite os %d nomes:\n", NUMNOMES);
for(i=0;i<NUMNOMES;i++)
    gets(nomes[i]);
for(i=0;i<NUMNOMES;i++)
    vetorponteiros[i]=nomes[i];
ordena_alfabeticamente2(vetorponteiros);
printf("Os nomes ordenados sao:\n");
for(i=0;i<NUMNOMES;i++)
    puts(vetorponteiros[i]);
system("pause");
}
```

4)Com base nas estruturas de dados definidas no programa principal a seguir:

- Implementar a função `DigitaDadosAcademicos` que recebe um número inteiro N, e lê todos os dados de N acadêmicos para a variável “`ListaAcademicos`”; **(10 pontos)**
- Implementar a função `CalcIdade`, que recebe uma struct `academico`, e uma struct `data`, e retorna a idade do acadêmico, considerando que o segundo parâmetro contém a data atual (favor verificar se o acadêmico já fez aniversário neste ano, ou ainda não), e retorne a idade (em anos completos) do acadêmico; **(10 pontos)**
- Implementar a função `ContaAlunos`, que receba o vetor de struct “`Lista`”, e um inteiro “`CodCurso`”, e retorna a quantidade de alunos cadastrados que são do curso de código “`CodCurso`”. **(10 pontos)**
- Implementar a função `NomeAcademicoMaisVelho` que retorna o nome do acadêmico mais velho. **(10 pontos)**

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

#define NUM_ALUNOS 1000
struct data
{
    int dia, mes, ano;
};
struct academico
{
    char Nome[60];
    int CodigoCurso;
    int idade;
    struct data Nascimento;
} ListaAcademicos[NUM_ALUNOS];
```

//4)Com base nas estruturas de dados definidas no programa principal a seguir:

```
//Implementar a função DigitaDadosAcademicos
//que recebe um número inteiro N, e lê todos os dados de N acadêmicos
//para a variável "ListaAcademicos"; (10 pontos)

//Implementar a função CalcIdade, que recebe uma struct academico,
//e uma struct data, e retorna a idade do acadêmico,
//considerando que o segundo parâmetro contém a data atual
//(favor verificar se o acadêmico já fez aniversário neste ano,
//ou ainda não), e retorne a idade (em anos completos) do acadêmico; (10
pontos)

//Implementar a função ContaAlunos, que receba o vetor de struct
"Lista",
//e um inteiro "CodCurso", e retorna a quantidade de alunos
//cadastrados que são do curso de código "CodCurso". (10 pontos)

//Implementar a função NomeAcademicoMaisVelho que retorna o nome do
//acadêmico mais velho. (10 pontos)
void DigitaDadosAcademicos(int N);
int CalcIdade (struct academico Aluno, struct data Hoje);
int ContaAlunos(struct academico Lista[NUM_ALUNOS],int CodCurso);
```

```

char *NomeAcademicoMaisVelho(struct academico Lista[NUM_ALUNOS]);

int main()
{
int i,N;
struct data Hoje;
Hoje.dia = 6;
Hoje.mes = 10;
Hoje.ano = 2008;
do
{
printf("Digite o Numero de alunos: (<%d)", NUM_ALUNOS);
scanf("%d", &N);
} while(N>NUM_ALUNOS);
DigitaDadosAcademicos(N);
for(i=0;i<N;i++)
{
printf("\nAluno %s\t", ListaAcademicos[i].Nome);
ListaAcademicos[i].idade = CalcIdade(ListaAcademicos[i], Hoje);
printf("Idade: %d.\n", ListaAcademicos[i].idade );
}
printf("\nNumero de alunos do curso 1:\n");
printf("%d\n", ContaAlunos(ListaAcademicos, 1) );
printf("\nNumero de alunos do curso 2: ");
printf("%d\n", ContaAlunos(ListaAcademicos, 2) );
printf("\nE o acadêmico mais velho eh: ");
printf("%s\n", NomeAcademicoMaisVelho(ListaAcademicos) );
system("pause");
}

void DigitaDadosAcademicos(int N)
{
int i;
for(i=0;i<NUM_ALUNOS;i++)
strcpy(ListaAcademicos[i].Nome, "");
for(i=0;i<N;i++)
{
fflush(stdin);
printf("Digite os dados do academico %d:\n",i);
printf("Nome:");
gets(ListaAcademicos[i].Nome);
printf("Codigo do Curso:");
scanf("%d", &ListaAcademicos[i].CodigoCurso);
printf("Data de Nascimento:\n");
printf("Dia:");
scanf("%d", &ListaAcademicos[i].Nascimento.dia);
printf("Mes:");
scanf("%d", &ListaAcademicos[i].Nascimento.mes);
printf("Ano:");
scanf("%d", &ListaAcademicos[i].Nascimento.ano);
}
}

```

```
int CalcIdade (struct academico Aluno, struct data Hoje)
{
    int idade = Hoje.ano - Aluno.Nascimento.ano;
    if (Aluno.Nascimento.mes > Hoje.mes)
        idade--;
    if (Aluno.Nascimento.mes == Hoje.mes)
        if (Aluno.Nascimento.dia > Hoje.dia)
            idade--;
    return(idade);
}

int ContaAlunos(struct academico Lista[NUM_ALUNOS],int CodCurso)
{
    int i, cont=0;
    for(i=0;i<NUM_ALUNOS;i++)
        if(Lista[i].CodigoCurso == CodCurso)
            cont++;
    return(cont);
}
```

```

char *NomeAcademicoMaisVelho(struct academico Lista[NUM_ALUNOS])
{
    int i,indicemaisvelho, idademaisvelho;
    idademaisvelho = 0;
    for(i=0;i<NUM_ALUNOS;i++)
        if(Lista[i].idade>idademaisvelho)
            {
                idademaisvelho = Lista[i].idade;
                indicemaisvelho = i;
            }
    return(Lista[indicemaisvelho].Nome);
}

char *NomeAcademicoMaisVelho2(struct academico Lista[NUM_ALUNOS])
{
    int i,indicemaisvelho, idademaisvelho;
    idademaisvelho = 0;
    for(i=0;i<NUM_ALUNOS;i++)
        {
            if(Lista[i].idade>idademaisvelho)
                {
                    idademaisvelho = Lista[i].idade;
                    indicemaisvelho = i;
                }
            if(Lista[i].idade == idademaisvelho)
                {
                    if(Lista[i].Nascimento.mes < Lista[indicemaisvelho].Nascimento.mes)
                        indicemaisvelho = i;
                    if(Lista[i].Nascimento.mes == Lista[indicemaisvelho].Nascimento.mes)
                        if(Lista[i].Nascimento.dia < Lista[indicemaisvelho].Nascimento.dia)
                            indicemaisvelho = i;
                }
        }
    return(Lista[indicemaisvelho].Nome);
}

```

O programa principal é:

```
#include <stdio.h>
#include <stdlib.h>
#define NUM_ALUNOS 1000
struct data
{
    int dia, mes, ano;
};
struct academico
{
    char Nome[60];
    int CodigoCurso;
    int idade;
    struct data Nascimento;
} ListaAcademicos[NUM_ALUNOS];

void DigitaDadosAcademicos(int N);
int CalcIdade (struct academico Aluno, struct data Hoje);
int ContaAlunos(struct academico Lista[NUM_ALUNOS],int CodCurso);
char * NomeAcademicoMaisAntigo(struct academico Lista[NUM_ALUNOS]);

int main()
{
    int i,N;
    struct data Hoje;
    Hoje.dia = 6;
    Hoje.mes = 10;
    Hoje.ano = 2008;
    do
    {
        printf("Digite o Numero de alunos: (<%d)", NUM_ALUNOS);
        scanf("%d", &N);
    } while(N>NUM_ALUNOS);
    DigitaDadosAcademicos(N);
    for(i=0;i<N;i++)
    {
        printf("\nAluno %s\t", ListaAcademicos[i].Nome);
        ListaAcademicos[i].idade = CalcIdade(ListaAcademicos[i], Hoje);
        printf("Idade: %d.\n", ListaAcademicos[i].idade) );
    }
    printf("\nNumero de alunos do curso 1: %d\n");
    printf("%d\n", ContaAlunos(ListaAcademicos, 1) );
    printf("\nNumero de alunos do curso 2: ");
    printf("%d\n", ContaAlunos(ListaAcademicos, 2) );
    printf("\nE o acadêmico mais velho eh: ");
    printf("%s\n", NomeAcademicoMaisVelho(ListaAcademicos) );
    system("pause");
}
```

5. No código abaixo existem erros de lógica. Indique os erros e explique como corrigi-los.
(10 pontos)

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

void selectSort(char a[10], int max);

int main()
{
    char vet[10] = {'c','j','x','m','q','p','r','b','n','a'};
    int i;
    for(i=0; i<10; i++){
        printf("%c ",vet[i]);
    }
    selectSort(vet[],10);
    printf("\n");
    for(i=0; i<10; i++){
        printf("%c ",vet[i]);
    }
    system("PAUSE");
    return 0;
}

void selectSort(char a[10], int max){
    int i,j,menor,troca=0;
    char valor;
    for(i=0;i<max;i++){
        valor = a[i];
        menor = i;
        troca = 0;
        for(j=i+1;j<max;j++){
            if(a[j]<valor){
                menor = j;
                valor = a[j];
                troca = 1;
            }
        }
        if(troca==1){
            a[i] = valor;
            a[menor] = a[i];
        }
    }
}
```

```

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

void selectSort(char a[10], int max);

int main()
{
    char vet[10] = {'c','j','x','m','q','p','r','b','n','a'};
    int i;
    for(i=0; i<10; i++){
        printf("%c ",vet[i]);
    }
    selectSort(vet,10);//A chamada não precisa []
    printf("\n");
    for(i=0; i<10; i++){
        printf("%c ",vet[i]);
    }
    system("PAUSE");
    return 0;
}

void selectSort(char a[10], int max){
    int i,j,menor,troca=0;
    char valor;
    for(i=0;i<max;i++){
        valor = a[i];
        menor = i;
        troca = 0;
        for(j=i+1;j<max;j++){
            if(a[j]<valor){
                menor = j;
                valor = a[j];
                troca = 1;
            }
        }
        if(troca==1){
            a[menor] = a[i];//Trocar estas duas linhas!!!!
            a[i] = valor;
        }
    }
}

```