MAC 115 – Introdução à Computação

Instituto de Física – Segundo Semestre de 2000 - Diurno

Prova Substitutiva – 15/12/2000

Nome do aluno: _____ Turma:_____

Assinatura: _____

Professor(a): _____

Nº USP: _____ Curso: _____

Instruções:

1. Não destaque as folhas deste caderno.
2. A prova pode ser feita a lápis. Cuidado com a legibilidade.
3. Há 3 questões na prova. Verifique antes de começar a prova se o seu caderno de questões está completo.
4. Não é permitido o uso de folhas avulsas para rascunho.
5. Nas questões que envolvem elaboração de programas, coloque comentários suficientes para que o programa seja facilmente compreendido.
6. Não é necessário apagar rascunhos no caderno de questões, mas indique claramente onde estão suas respostas.

Não escreva nesta parte da folha

Questão	Nota
1	
2	
3	
Total	

BOA SORTE!

Questão 1 (valor: 3.0)

Simule a execução do programa abaixo *destacando a sua saída* (o que vai sair na tela). Dados de entrada (a serem lidos): nenhum.

```
#include <stdio.h>
#define TRUE 1
#define FALSE 0

int f1(int a, int b, int c);
void f2(int a, int b, int *pmin, int *pmax);
void f3(int v[], int n);
/*-----------------------------------------------*/

int main()
{
    int x, y, z, w, a[4];

    x = 4;  y = 7;  z = 5;

    printf("Funcao f1:\n");
    printf("x = %d  y = %d  z = %d \n", x, y, z);
    printf("f1(%d, %d, %d) = %d \n\n", x, y, z, f1(x, y, z));

    printf("Funcao f2:\n");
    f2(x, y, &z, &w);
    printf("x = %d  y = %d z = %d  w = %d\n\n", x, y, z, w);

    printf("Funcao f3:\n");
    a[0] = 3;  a[1] = 7;  a[2] = 1;  a[3] = 5;

    printf(" a[0] = %d  a[3] = %d\n", a[0], a[3]);
    f3(a, 4);
    printf(" a[0] = %d  a[3] = %d\n\n", a[0], a[3]);

    return 0;
}
/*-----------------------------------------------*/
int f1(int a, int b, int c)
{
    if ((a + b + c) % 2 == 0)
        return TRUE;
    else
        return FALSE;
}
/*-----------------------------------------------*/
void f2(int a, int b, int *pmin, int *pmax)
{
    if (a > b)
        { *pmax = a; *pmin = b; }
    else
        { *pmax = b; *pmin = a; }

    printf("Dentro da funcao f2: maior = %d, menor = %d.\n", *pmax, *pmin);
}
```

```
/*-----------------------------------------------*/
void f3(int v[], int n)
{
    int i, min, max;

    min = max = v[0];
    for (i = 1; i < n; i++) {
        if (v[i] < min) min = v[i];
        if (v[i] > max) max = v[i];
    }
    v[0] = min;
    v[n - 1] = max;
}
```

```
#=================================================================#
# função f1 #      função f2         #     função f3               #
#==========#======================#=============================#
# a | b | c #  a |  b  |*pmin |*pmax # v[0] v[1] v[2] v[3] min max n #
#---|---|---#-----|-----|------|------#-----------------------------#
#   |   |   #     |     |      |      #                             #
#   |   |   #     |     |      |      #                             #
#   |   |   #     |     |      |      #                             #
#   |   |   #     |     |      |      #                             #
#   |   |   #     |     |      |      #                             #
#=================================================================#
```

```
#=======================#
#   função main()       #
#=======================#
#  x  |  y  |  z  |  w   #
#-----|-----|-----|------#
#     |     |     |      #
#     |     |     |      #
#     |     |     |      #
#     |     |     |      #
#=======================#
```

Saída (o que vai sair na tela):

Questão 2 (valor: 3.0)

Nesta questão, vamos representar inteiros (possivelmente com um número grande de dígitos) em vetores. Por exemplo, para representar o inteiro

`3141592653589793238462643383279502884197169399375`1,

armazenamos em um vetor, digamos `v[]`, os dígitos `1`, `5`, `7`, `3`, `9`, `9`, etc, nesta ordem, isto é, `v[0] = 1`, `v[1] = 5`, `v[2] = 7`, e assim por diante (é mais conveniente termos os dígitos na 'ordem inversa'). A idéia é escrever um programa que soma dois inteiros (grandes) dados. Você deve supor que, nesta questão, os inteiros a serem somados têm no máximo `1000` dígitos.

(a) Escreva uma função de protótipo

```
void leia_vetor(int v[], int n);
```

que lê as entradas de um vetor inteiro `v[]` de `n` elementos. Escreva também uma função de protótipo

```
void imprima_vetor(int v[], int n);
```

que imprime o vetor inteiro `v[]` de `n` elementos. Sua função `imprima_vetor()` deve imprimir os elementos de `v[]` na ordem 'reversa': primeiro `v[n-1]`, depois `v[n-2]`, etc, sem deixar espaço entre estes inteiros.

(*b*) Escreva uma função de protótipo

```
int some(int s[], int a[], int n_a, int b[], int n_b);
```

que recebe em `a[]` e, respectivamente, em `b[]` as seqüências de dígitos dos inteiro a e b (como explicado no início desta questão), e que devolve em `s[]` uma representação da soma $a + b$. A sua função deve receber em `n_a` e em `n_b` o número de dígitos de a e b, respectivamente, e deve devolver o número de dígitos na soma $a + b$.

Exemplo. Suponha que $a = 82434256$ e $b = 33752337$. Então, a sua função será chamada com

```
a = {6, 5, 2, 4, ... }
b = {7, 3, 3, 2, ... }
```

e `n_a = n_b = 8`. A sua função deve então produzir o vetor

```
s = {3, 9, 5, 6, ... }
```

pois $a + b = 116186593$, e o valor devolvido por `some()` deve ser 9 ($a + b$ tem 9 dígitos).

(*c*) Escreva um programa que recebe dois inteiros positivos a e b com n_a e n_b dígitos, respectivamente, e determina a soma $a + b$. Os valores de n_a e n_b são dados pelo usuário, e os números a e b são fornecidos dígito por dígito, na ordem reversa (unidades primeiro, depois dezenas, depois centenas, etc).

Exemplo. Suponha que $a = 82434251$ e $b = 33752337$. Então, a entrada do seu programa será

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

e a saída deve ser

```
82434251
+
33752337
=
116186588
```

Questão 3 (valor: 4.0)

Dada uma matriz inteira, definimos o *peso* de uma linha dessa matriz como sendo a soma do *menor* e do *maior* elemento dessa linha. Por exemplo, na matriz

$$A = \begin{pmatrix} -4 & 7 & 8 & 10 & -17 & 28 \\ 2 & 3 & -5 & 8 & 15 & 25 \\ -8 & 10 & 14 & -16 & 20 & 48 \\ 7 & 8 & 9 & 12 & 18 & -45 \end{pmatrix},$$

as linhas têm peso 11, 20, 32, e -17.

(a) Escreva uma função de protótipo

```
void leia_matriz(int a[][NMAX], int m, int n);
```

que lê as entradas da matriz inteira `a` (`m x n`). Escreva também uma função de protótipo

```
void imprima_matriz(int a[][NMAX], int m, int n);
```

que imprime a matriz inteira `a` (`m x n`).

(*b*) Escreva uma função de protótipo

```
void menor_maior(int v[], int comp, int *pmin, int *pmax);
```

que recebe um vetor inteiro `v[]` e um inteiro `comp` (o número de elementos de `v[]`), e devolve o menor e o maior elemento do vetor `v[]` em `*pmin` e `*pmax`.

(*c*) Escreva uma função de protótipo

```
int processe_matriz(int a[][NMAX], int m, int n);
```

que recebe uma matriz inteira `a` (`m x n`), e imprime o seguinte:

- o menor e o maior elemento de cada linha e o peso desta linha,
- o índice da linha de maior peso,
- o peso máximo encontrado.

(Em caso de haver mais de uma linha de peso máximo, `processe_matriz()` deve imprimir o índice de qualquer uma delas.) Ao ser chamada com a matriz A acima, sua função deve imprimir algo como

```
Pesos:
0: -17  28  peso = 11
1:  -5  25  peso = 20
2: -16  48  peso = 32
3: -45  18  peso = -17
Linha de maior peso: 2
Peso maximo: 32
```

Finalmente, `processe_matriz()` deve devolver o índice da linha de maior peso (2, no exemplo acima). Você deve obrigatoriamente usar a função `menor_maior()` acima, mesmo que você não a tenha feito.

(d) Escreva um programa que recebe como entrada dois inteiros m e n (com $0 < m, n \leq 50$) e uma matriz inteira A ($m \times n$), e que, primeiro, imprime a matriz lida. Uma vez impressa a matriz, o seu programa deve chamar função do item (c) para esta matriz. Seu programa deve ainda imprimir os elementos da linha de A com peso máximo.

Exemplo. Suponha que a entrada de seu programa é

```
 4    6
-4    7    8   10  -17   28
 2    3   -5    8   15   25
-8   10   14  -16   20   48
 7    8    9   12   18  -45
```

A saída de seu programa poderia então ser algo como

```
Matriz lida:
  -4   7   8  10 -17  28
   2   3  -5   8  15  25
  -8  10  14 -16  20  48
   7   8   9  12  18 -45
Pesos:
0: -17  28  peso = 11
1:  -5  25  peso = 20
2: -16  48  peso = 32
3: -45  18  peso = -27
Linha de maior peso: 2
Peso maximo: 32
Elementos da linha de peso maximo:
-8 10 14 -16 20 48
```