MAC 115 – Introdução à Computação

Instituto de Física – Segundo Semestre de 2002 - Noturno

Prova Substitutiva – 10/12/2002

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)

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

```
3 1 4 2
```

```
#include <stdio.h>

void sei_la(int *a, int *b);

int main()
{
  int x, y, z, w;

  scanf("%d %d %d %d", &x, &y, &z, &w);
  sei_la(&x, &y);
  sei_la(&z, &w);
  sei_la(&x, &z);
  sei_la(&y, &w);

  printf("%d %d %d %d\n", x, y, z, w);
  sei_la(&y, &z);

  printf("%d %d %d %d\n", x, y, z, w);
  return 0;
}

void sei_la(int *a, int *b)
{
  int t;
  if (*b < *a) {
    t = *a;
    *a = *b;
    *b = t;
  }
}
```

```
Saída (o que vai sair na tela):
-------------------------------
```

(*b*) Forneça uma entrada que faz com que os dois `printf()` do programa acima gerem a mesma saída. Não deixe de dizer qual será essa saída.

Questão 2 (valor: 3.0)

(a) Escreva uma função de protótipo

```
void elimine(int a[], int j, int n);
```

que elimina do vetor fornecido a o item $a[j]$. Os elementos $a[j+1], \ldots, a[n-1]$ devem ser 'deslocados' uma posição para a esquerda, de forma a 'fechar' o 'espaço' deixado pela eliminação de $a[j]$. Você pode supor que $0 \leq j < n$.

(b) Escreva uma função de protótipo

```
int reduza(int a[], int i, int n);
```

que recebe um vetor a com n entradas e faz o seguinte: `reduza()` verifica se o valor $a[i]$ ocorre como $a[j]$ para algum $j > i$. Se isto não ocorrer, `reduza()` deixa o vetor a intacto e devolve o valor n. Caso contrário, `reduza()` deve eliminar de a todas as ocorrências do valor $a[i]$, a menos daquela ocorrência como $a[i]$ (isto é, todos os $a[j]$ com $a[j] = a[i]$ e $j > i$ devem ser eliminados). A função `reduza()` deve devolver o número de elementos restantes no vetor, após a eliminação de todas as duplicações do valor $a[i]$ como descrito acima. Você deve usar a função de (a), mesmo que você não a tenha escrito.

(c) Escreva uma função de protótipo

```
int elimine_duplicacoes(int a[], int n);
```

que elimina do vetor fornecido a todos os valores duplicados, deixando nele uma ocorrência de cada valor distinto que nele ocorre. O usuário desta função passa em n o número de elementos do vetor a; por sua vez, a função `elimine_duplicacoes()` deve devolver o número de elementos restantes no vetor a após todas as eliminações de duplicações. Você deve usar a função de (b), mesmo que você não a tenha escrito.

Questão 3 (valor: 4.0)

Nesta questão, diremos que uma seqüência de números $x_0, \ldots, x_{n-1}$ é *crescente* se $x_0 \leq \ldots \leq x_{n-1}$ e decrescente se $x_0 \geq \ldots \geq x_{n-1}$. Uma seqüência que não é nem crescente nem decrescente será chamada de *não-monotônica*. Nesta questão, você pode supor que $2 \leq n \leq$ `NMAX`, onde `NMAX` é uma certa constante, e que não ocorrem seqüências constantes.

(a) Escreva uma função de protótipo

```
int tipo(int l[], int n);
```

que recebe como entrada um vetor l de n inteiros e decide se a seqüência $l[0], \ldots, l[n-1]$ é crescente, decrescente, ou não-monotônica. A sua função deve devolver -1 se a seqüência for decrescente, 1 se ela for crescente, e 0 se ela for não-monotônica. (*Sugestão.* Você pode achar conveniente usar um vetor auxiliar d, com $d[i] = l[i] - l[i-1]$ $(0 < i < n)$.)

(*b*) Dada uma matriz inteira a com m linhas e n colunas, definimos o *tipo* da linha i de a como sendo `crescente`, `decrescente`, ou `não-monotônica` dependendo da seqüência $a[i][0], \ldots, a[i][n-1]$ ser crescente, decrescente, ou não-monotônica.

Escreva uma função de protótipo

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

que determina o tipo de cada linha de a, colocando o resultado no vetor `tipos[]`. Mais precisamente, a sua função deve devolver `tipos[]` de forma que `tipos`$[i] = -1$ se i-ésima linha de a é decrescente, `tipos`$[i] = 1$ se i-ésima linha de a é crescente, e `tipos`$[i] = 0$ se i-ésima linha de a é não-monotônica.

(c) Escreva uma função de protótipo

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

que lê as entradas de uma matriz inteira a com m linhas e n colunas. Escreva também uma função de protótipo

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

que imprime uma tal matriz.

(*d*) Escreva um programa que lê inteiros m e n, lê uma matriz a com m linhas e n colunas, e decide se ela é *homogênea*, isto é, se todas as suas linhas são do mesmo tipo (todas crescentes, todas decrescentes, ou todas não-monotônicas). Você pode supor que a matriz não tem linhas constantes.