Instruções sobre o Exercício

1001101/481556 — Projeto e Análise de Algoritmos
Cândida Nunes da Silva
2º Semestre de 2020

1 Problema - Soma máxima em matriz

Considere uma matriz A de 3 linhas e $n \geq 1$ colunas cujas células são valores inteiros. Desejamos encontrar uma sequência $S = (s_1, s_2, \ldots, s_n)$ de n elementos tal que todo s_j é um elemento da coluna j de A, para $j = 1, 2, \ldots, n$. Além disso, para todo $j = 2, \ldots, n$, se $s_j = A_{ij}$, então necessariamente $s_{j-1} = A_{tj-1}$ onde $t = i$ ou $t \equiv i + 1 \pmod 3$. Em outras palavras, o elemento anterior da sequência deve sempre vir da mesma linha ou da linha anterior, sendo que a linha anterior à primeira é a terceira (porque a soma é tomada módulo 3). Assim, há sempre duas possibilidades de escolha para o elemento anterior da sequência.

Dentre todas as sequências possíveis que satisfazem essas restrições, queremos encontrar uma cuja soma dos elementos é a maior possível e, finalmente, qual o valor desta soma. Por exemplo, para a matriz abaixo ...

-15	8	15	-2	9	8
3	-10	4	-3	7	-1
10	7	2	5	6	11

... uma sequência de soma máxima é a seguinte $(3, 8, 15, 5, 6, 11) = 48$. Projete um algoritmo **de complexidade O(n)** para resolver este problema. Sugestão: pense **indutivamente**, mas não use recursão na implementação.

2 Entrada

A entrada deve ser lida da entrada padrão (teclado). A entrada contém quatro linhas. A primeira linha com um inteiro N correspondente ao número de colunas da matriz sendo que $1 \leq N \leq 100000$. As 3 linhas seguintes contém uma sequência de N elementos M onde $-500 \leq M \leq 500$. Cada elemento M corresponde ao valor do inteiro localizado na coordenada correspondente na matriz.

3 Saída

A saída deve ser escrita na saída padrão (terminal). A saída deve ser **apenas** o número correspondente à soma máxima e uma quebra de linha.

4 Exemplos

Para a seguinte matriz:

-15	8	15	-2	9	8
3	-10	4	-3	7	-1
10	7	2	5	6	11

Entrada	**Saída**
6 -15 8 15 -2 9 8 3 -10 4 -3 7 -1 10 7 2 5 6 11	48

Em outro exemplo, a matriz:

7	11	63	40	2	98	37	86	6	18	65	95	87	100	45	48
92	31	57	15	49	96	35	73	91	32	62	24	75	25	27	59
36	34	84	64	69	38	74	54	90	85	46	8	66	99	43	50

Entrada	**Saída**
16 7 11 63 40 2 98 37 86 6 18 65 95 87 100 45 48 92 31 57 15 49 96 35 73 91 32 62 24 75 25 27 59 36 34 84 64 69 38 74 54 90 85 46 8 66 99 43 50	1160

5 Desenvolvimento e Apresentação

Cada aluno deve implementar a sua solução individual. A implementação da solução do problema deve ser em C em arquivo único. O nome do arquivo deve estar na forma "ex03-nomesn.c", onde "nomesn" representa o primeiro nome do aluno seguido das iniciais de seu sobrenome. Note que todas as letras são minúsculas e o separador é "-" (hífen) e não "_" (underscore).

Serão disponibilizados vários arquivos com as entradas e as respectivas saídas esperadas. É **imprescindível** que o **algoritmo** implementado esteja correto, isto é, retorne a solução esperada para **qualquer** arquivo de testes. Também é necessário que a implementação tenha a **complexidade esperada**.

6 Ambiente de Execução e Testes

O programa deve ser compilável em ambiente Unix com `gcc`. Sugere-se que os testes também sejam feitos em ambiente Unix. Deve-se esperar que a entrada seja dada na entrada padrão (teclado) e não por leitura do arquivo de testes. Da mesma forma, a saída deve ser impressa na saída padrão (terminal), e não em arquivo.

A motivação dessa exigência é apenas simplificar a implementação de entrada e saída, permitindo o uso das funções `scanf` e `printf` da biblioteca padrão para leitura e escrita dos dados, sem precisar manipular arquivos.

Por outro lado, é evidente que efetivamente entrar dados no teclado é muito trabalhoso. Em ambiente Unix, é possível usar redirecionamento de entrada na linha de comando de execução para contornar esse problema. Supondo que o nome do arquivo executável seja análogo ao arquivo fonte, e "ex03.in" seja o arquivo com um caso de teste, a linha de comando:

```
shell$ ./ex03-nomesn < ex03.in
```

executa o programa para o caso de teste, retornando a saída para o teminal. Novamente, pode-se usar o redirecionamento de saída na linha de comando para escrever a saída em um arquivo de saída de nome, por exemplo, "ex03.my.out". A respectiva linha de comando seria:

```
shell$ ./ex03-nomesn < ex03.in > ex03.my.out
```

Após a execução, a comparação pode ser feita usando o comando `diff` do Unix. Por exemplo, se o arquivo "ex03.out" contém as saídas esperadas, a linha de comando:

```
shell$ diff ex03.out ex03.my.out
```

serve para comparar automaticamente os dois arquivos, retornando nada caso sejam idênticos e as linhas onde há discrepâncias caso contrário.

7 Notas

As notas serão baseadas na correção da solução implementada, clareza do código fonte e eficiência da solução.

Trabalhos que não atendam aos requisitos mínimos expressos neste documento de forma a invibializar o teste do programa receberão nota ZERO. Em particular, receberá nota ZERO todo programa que:

- não compila em ambiente Unix;
- dá erro de execução;
- não usa entrada e saída padrão para leitura e escrita.
- apresentar sinais claros de cópia seja de outros colegas ou de código disponível na Internet.