MC202 — ESTRUTURAS DE DADOS

Laboratório 02 — Multiplicação de Matrizes

João é aluno do ensino médio e está aprendendo operações com matrizes. Uma dessas operações é a multiplicação. Dadas duas matrizes $A = (a_{ij})_{mxn}$ e $B = (b_{ij})_{nxp}$, o produto entre as matrizes A e B é a matriz $C = (c_{ij})_{mxp}$, onde c_{ij} é obtido multiplicando-se ordenadamente os elementos da linha i da matriz A pelos elementos da coluna j da matriz B e somando os produtos obtidos. Portanto, escrevemos $C = A.B = (c_{ij})_{mxp}$, onde:

$$c_{ij} = \sum_{k=1}^{n} a_{ik} . b_{kj}$$

Em seus estudos, João aprendeu o conceito de matriz de permutação. Uma matriz de permutação é uma matriz binária de ordem n, i.e., uma matriz quadrada composta por zeros e uns em que cada linha e cada coluna tem exatamente um número um. Essa matriz tem o efeito de gerar uma permutação dos elementos de um vetor ou das linhas de uma matriz. Veja o exemplo abaixo:

$$A.B = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 3 & 5 & 7 \\ 2 & 4 & 6 & 8 \\ 3 & 5 & 7 & 9 \\ 4 & 6 & 8 & 10 \end{pmatrix} = \begin{pmatrix} 4 & 6 & 8 & 10 \\ 1 & 3 & 5 & 7 \\ 2 & 4 & 6 & 8 \\ 3 & 5 & 7 & 9 \end{pmatrix}$$

Uma coisa boa das matrizes de permutação é que podemos economizar espaço representando a matriz apenas por um vetor p de tamanho n, em que cada posição representa o número da linha que contém o um da matriz. Por exemplo, para a matriz A acima, temos o vetor

$$p = (2\ 3\ 4\ 1)^T.$$

João gostou bastante desses conceitos e resolveu criar um programa que faça multiplicação de uma matriz real por uma matriz de permutação.

Tarefa

A sua tarefa consiste em desenvolver um programa em C que faça a leitura de um vetor de permutação p e uma matriz quadrada B de ordem n. Seu programa deve criar uma matriz de permutação A correspondente a p e calcular o produto $A.B$.

Entrada

A primeira linha da entrada é um inteiro n (2 ≤ n ≤ 312) indicando a ordem das matrizes. A próxima linha conterá um vetor p de inteiro correspondente a matriz de permutação A. E por fim, as próximas n linhas conterão os elementos da matriz B separados por espaço. Observe o exemplo abaixo:

```
3
2 1 3
1.5 3.6 7.8
2.0 5.2 3.3
1.9 6.8 9.0
```

Saída

O programa deve produzir n linhas na saída correspondente a matriz C resultante da multiplicação de $A.B$. Note o exemplo da saída abaixo:

```
2.0 5.2 3.3
1.5 3.6 7.8
1.9 6.8 9.0
```

Note que na saída, os números deverão estar formatados para uma casa decimal após a vírgula. Veja o exemplo de formatação abaixo:

```
double a = 7.5612;
printf("%.1f", a);
```

Critérios específicos

- Para as turmas E e F, este laboratório tem peso 1.
- Para as turmas G e H, este laboratório tem peso 1.
- Deverá ser submetido o seguinte arquivo: **lab02.c**.
- Tempo máximo de execução: 2 segundos.

Testando

Para compilar com o Makefile fornecido e verificar se a solução está correta basta seguir o exemplo abaixo.

```
make
./lab02 < arq01.in > arq01.out
diff arq01.out arq01.res
```

onde `arq01.in` é a entrada (casos de testes disponíveis no SuSy) e `arq01.out` é a saída do seu programa. O Makefile também contém uma regra para testar todos os testes de uma vez; nesse caso, basta digitar:

```
make testar_tudo
```

Observações gerais

No SuSy, haverá 3 tipos de tarefas com siglas diferentes para cada laboratório de programação. Todas possuirão os mesmos casos de teste. As siglas são:

1. **SANDBOX:** Esta tarefa serve para testar o programa no SuSy antes de submeter a versão final. Nessa tarefa, tanto o prazo quanto o número de submissões são ilimitados, porém arquivos submetidos aqui **não serão corrigidos**.

2. **ENTREGA:** Esta tarefa tem limite de **uma única submissão** e serve para entregar a versão final dentro do prazo estabelecido para o laboratório. Não use essa tarefa para testar o seu programa: submeta aqui quando não for mais fazer alterações no seu programa.

3. **FORAPRAZO:** Esta tarefa tem limite de **uma única submissão** e serve para entregar a versão final após o prazo estabelecido para o laboratório, mas com nota reduzida (conforme a ementa). O envio nesta tarefa irá substituir a nota obtida na tarefa ENTREGA apenas se o aluno tiver realizado as correções sugeridas no feedback ou caso não tenha enviado anteriormente em ENTREGA.

Observações sobre SuSy:

- Versão do GCC: C-ANSI 4.8.2 20140120 (Red Hat 4.8.2-15).

- Flags de compilação:
  `-ansi -Wall -pedantic-errors -Werror -g -lm`

- Utilize comentários do tipo `/* comentário */`;
  comentários do tipo `//` serão tratados como erros pelo SuSy.

Além das observações acima, esse laboratório será avaliado pelos critérios gerais:

- Indentação de código e outras boas práticas, tais como:
    - uso de comentários (apenas quando forem relevantes);
    - código simples e fácil de entender;
    - sem duplicidade (partes que fazem a mesma coisa).
- Organização do código:
    - tipos de dados criados pelo usuário e funções bem definidas e tão independentes quanto possível.
- Corretude do programa:
    - programa correto e implementado conforme solicitado no enunciado;
    - inicialização de variáveis sempre que for necessário;
    - dentre outros critérios.