

Aula 31: Matrizes em blocos e em 3 dimensões

Lógica de Programação – Superior · Bloco 3

José Dihego · IFBA

O que você precisa saber hoje

Objetivo: Percorrer sub-blocos de uma matriz e trabalhar com uma terceira dimensão (canais).

- Bloco $F \times F$ começando em (b_i, b_j) : i de b_i a $b_i + F - 1$
- Inícios de bloco andam de F em F : $b_i += F$
- Quatro laços: dois para os blocos, dois dentro do bloco
- Imagem colorida: $img[i][j][c]$, $c = 0$ (R), 1 (G), 2 (B)

```
for (int bi = 0; bi < n; bi += f)
    for (int bj = 0; bj < n; bj += f) {
        int maior = img[bi][bj];
        for (int i = bi; i < bi + f; i++)
            for (int j = bj; j < bj + f; j++)
                if (img[i][j] > maior) maior = img[i][j];
        printf("%d ", maior);
    }
```

Erros que custam pontos

Errado	Certo	Por quê
<pre>for (bi = 0; bi < n; bi++)</pre>	<pre>bi += f</pre>	blocos sobrepostos
<pre>int maior = 0;</pre>	<pre>int maior = img[bi][bj];</pre>	funciona aqui, mas falha com negativos
<pre>int img[100][100][3]; /* no main */</pre>	<pre>/* global */</pre>	matrizes grandes podem estourar a pilha

```
int img[4][4] = {{1, 3, 2, 1}, {4, 6, 5, 0}, {7,  
    2, 9, 8}, {3, 1, 4, 4}}, f = 2;  
for (int bi = 0; bi < 4; bi += f)  
    for (int bj = 0; bj < 4; bj += f) {  
        int maior = img[bi][bj];  
        for (int i = bi; i < bi + f; i++) for (  
            int j = bj; j < bj + f; j++) if (img[i][j] >  
                maior) maior = img[i][j];  
    }
```

bloco (bi, bj)	valores	máximo
(0, 0)	1 3 4 6	6
(0, 2)	2 1 5 0	5
(2, 0)	7 2 3 1	7
(2, 2)	9 8 4 4	9

Avaliação 3 – 2024.2, Questão I

avaliacao_3_ano_2024_2.pdf

Max pooling com filtro $F \times F$ e passo F

Estratégia:

1. Ler N , F e a imagem
2. Para cada bloco $F \times F$ (passo F), achar o máximo
3. Imprimir uma linha de máximos por linha de blocos

Enunciado completo (com figuras e exemplos) no PDF da prova.

Solução (1/2)

```
1  #include <stdio.h>           // printf e scanf
2  #define MAX 100             // maior imagem aceita
3
4  int main(void) {             // início do programa
5      int n, f, img[MAX][MAX]; // dimensão da imagem, do filtro e a imagem
6      scanf("%d %d", &n, &f);  // lê N e F (N múltiplo de F)
7      for (int i = 0; i < n; i++) // lê a imagem
8          for (int j = 0; j < n; j++) scanf("%d", &img[i][j]); // cada coluna
9      for (int bi = 0; bi < n; bi += f) { // linha de início de cada bloco (passo F)
10         for (int bj = 0; bj < n; bj += f) { // coluna de início de cada bloco
```

Solução (2/2)

```
11         int maior = img[bi][bj];           // começa com o canto do bloco
12         for (int i = bi; i < bi + f; i++) // percorre o bloco F x F
13             for (int j = bj; j < bj + f; j++) // colunas do bloco
14                 if (img[i][j] > maior) maior = img[i][j]; // guarda o máximo
15         printf("%d ", maior);               // um valor por bloco
16     }
17     printf("\n");                          // fim de uma linha de blocos
18 }
19 return 0;                                // fim sem erro
20 }
```

Avaliação 3 – 2025.1, Questão II

avaliacao_3_ano_2025_1.pdf

Imagem RGB para tons de cinza (na prova, 100 x 100 x 3)

Dicas:

- Matriz `img[100][100][3]` (global)
- Cinza = $0,299 R + 0,587 G + 0,114 B$
- Arredonde: `(int)(cinza + 0.5)`

Para conferir: entrada 1 1 200 150 100 → a saída deve conter 159

A solução comentada está na página do curso: <https://josedihego.net/logica/#aula-31>