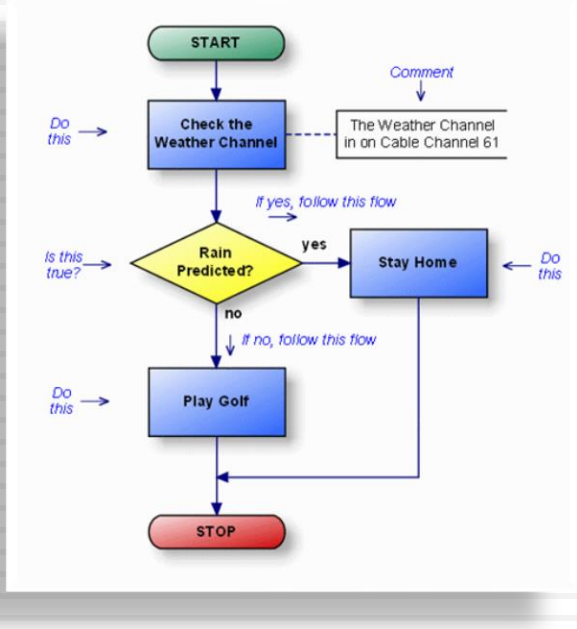
Đại Học Quốc Gia Tp. Hồ Chí Minh
Trường ĐH Khoa Học Tự Nhiên
Khoa Công Nghệ Thông Tin

NHẬP MÔN LẬP TRÌNH



CHƯƠNG 6: MẢNG HAI CHIỀU (MATRIX)

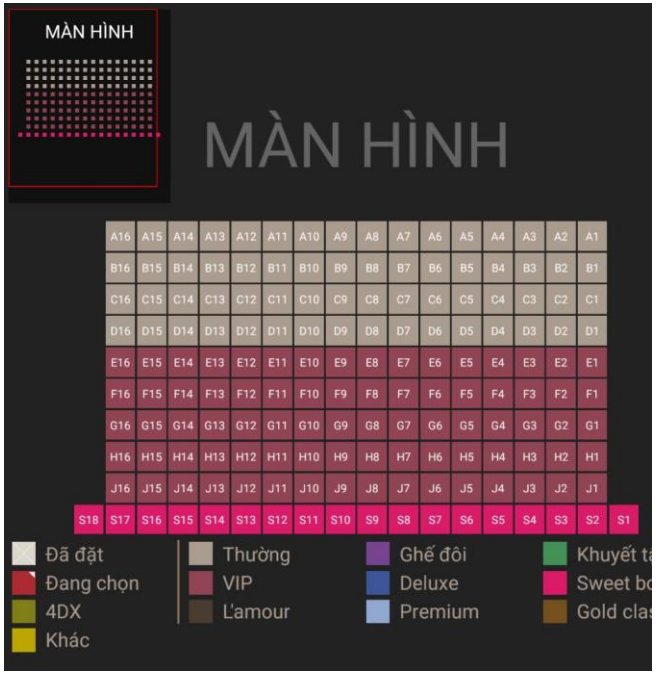
GV: Phạm Nguyễn Sơn Tùng
Email: pnstung@fit.hcmus.edu.vn

THÔNG TIN CHUNG

1. Giới thiệu tổng quan
2. Các cách khai báo mảng hai chiều
3. Các kỹ thuật xử lý trên mảng hai chiều
4. Bài tập ứng dụng tại lớp
5. Bài tập về nhà

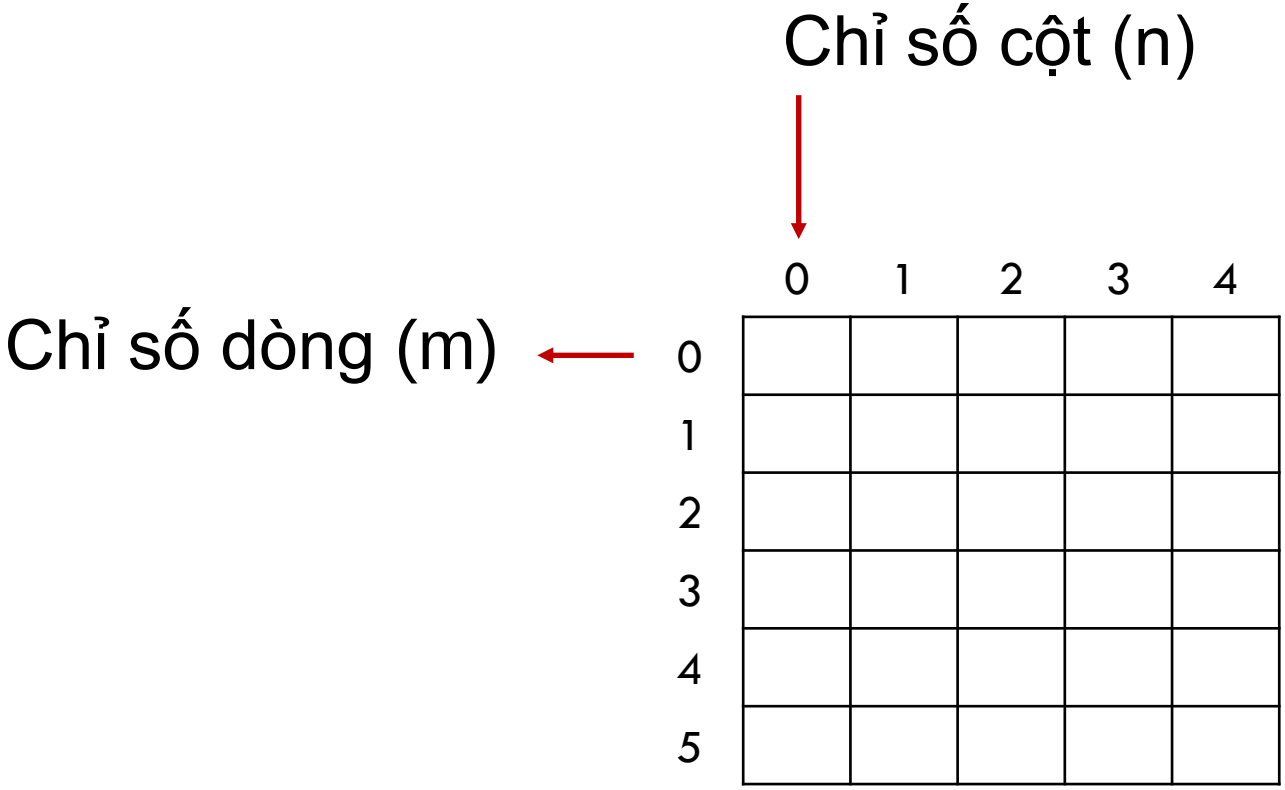
GIỚI THIỆU TỔNG QUAN MẢNG MỘT CHIỀU

Định nghĩa: Mảng 2 chiều (hay còn gọi ma trận) là cấu trúc dữ liệu, biểu diễn các đối tượng ở dạng dòng và cột.



HÌNH ẢNH MINH HỌA MẢNG HAI CHIỀU

Cho mảng hai chiều 4 dòng, 4 cột:



CÁC CÁCH KHAI BÁO MẢNG MỘT CHIỀU

- Có 2 cách khai báo để sử dụng mảng hai chiều:

Sử dụng mảng 2 chiều tĩnh

Sử dụng mảng 2 chiều động

CÁC CÁCH KHAI BÁO MẢNG HAI CHIỀU

- Có 2 cách khai báo để sử dụng mảng hai chiều:

Sử dụng mảng 2 chiều tĩnh

- Mảng tĩnh là mảng cấp phát bộ nhớ ngay từ đầu.

Sử dụng mảng 2 chiều động

- Mảng động là mảng sẽ được cấp phát vùng nhớ trong khi chương trình chạy.

KHAI BÁO MẢNG 2 CHIỀU TĨNH

	0	1	2	3	4
0		15	19	-8	
1					
2					
3					
4					
5					

Kiểu dữ liệu

Tên của mảng 2 chiều

Số lượng **dòng (m)** trong mảng 2 chiều

Số lượng **cột (n)** trong mảng 2 chiều

```
int a[5][4];
a[0][1] = 15;
a[0][2] = 19;
a[0][3] = -8;
…
```

TRUY XUẤT VÀ THAO TÁC

```
//Truy xuất phần tử trong mảng
int x = a[0][1];
int y = a[0][2];

//Thao tác phần tử trong mảng
a[1][0] = 5;
a[1][1] = 9;

//Sai - báo lỗi
a[-1][0] = 7
```

KHAI BÁO MA TRẬN TĨNH DẠNG HÀM

```
void Nhap(int a[][100], int& m, int& n)
{
    cout << "Nhap so dong m: ";
    cin >> m;
    cout << " Nhap so cot n: ";
    cin >> n;
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            cout << "a[" << i << "][" << j << "]: ";
            cin>>a[i][j];
        }
    }
}
```

KHAI BÁO MA TRẬN TĨNH DẠNG HÀM

```
void Xuat(int a[][100], int m, int n)
{
    cout << "---- Noi dung cua mang 2 chieu ----"<<endl;
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            cout << a[i][j]<<" ";
        }
        cout << endl;
    }
}
```

KHAI BÁO MA TRẬN TĨNH DẠNG HÀM

```
int main()
{
    int a[100][100];
    int m, n;
    Nhap(a, m, n);
    Xuat(a, m, n);
    return 0;
}
```

CÁC KỸ THUẬT XỬ LÝ MA TRẬN

1. Kỹ thuật duyệt toàn bộ ma trận.
2. Kỹ thuật duyệt các phần tử trên biên ma trận.
3. Kỹ thuật xử lý trên dòng.
4. Kỹ thuật xử lý trên cột.
5. Kỹ thuật hoán vị trên ma trận.
6. Sắp xếp ma trận.
7. Kỹ thuật xử lý ma trận con.
8. Ma trận vuông.

1. KỸ THUẬT DUYỆT TOÀN BỘ MA TRẬN

Định nghĩa: Là kỹ thuật duyệt toàn bộ ma trận đi từ trên xuống dưới và từ trái qua phải để thực hiện 1 yêu cầu nào đó (giống như mảng 1 chiều).

- Lính canh
- Tính tổng
- Đếm
- Liệt kê

	0	1	2	3	4
0	1	2	3	4	5
1	6	7	8	9	10
2	11	12	13	14	15
3	16	17	18	19	20
4	21	22	23	24	25
5	26	27	28	29	30

1. KỸ THUẬT DUYỆT TOÀN BỘ MA TRẬN

Ví dụ minh họa: Tìm lớn nhì trong ma trận các số nguyên.

```
int TimPhanTuLonNhat(int a[][MAX], int m, int n)
{
    int max = a[0][0];
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            if (max < a[i][j])
                max = a[i][j];
        }
    }
    return max;
}
```

1. KỸ THUẬT DUYỆT TOÀN BỘ MA TRẬN

```
int TimPhanTuLonNhi(int a[][MAX], int m, int n)
{
    int max = TimPhanTuLonNhat(a, m, n);
    int max2 = a[0][0];
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            if (a[i][j] > max2 && a[i][j] < max)
                max2 = a[i][j];
        }
    }
    return max2;
}
```

KỸ THUẬT DUYỆT TOÀN BỘ MA TRẬN

Bài tập 1: Tìm các phần tử đối xứng trong ma trận.

Bài tập 2: Đếm xem có bao nhiêu phần tử chia hết cho 2 trong ma trận

2. KỸ THUẬT DUYỆT THEO BIÊN

Định nghĩa: là kỹ thuật dùng để duyệt các phần tử theo biên của ma trận để xử lý yêu cầu của bài toán.

	0	1	2	3	4
0	1	2	3	4	5
1	6				7
2	8				9
3	10				11
4	12				13
5	14	15	16	17	18

2. KỸ THUẬT DUYỆT THEO BIÊN

Ví dụ minh họa: Tính tổng các phần tử bao quanh ngoài ma trận.

```
int TongBienMaTran(int a[][MAX], int m, int n)
{
    int tong = 0;
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            if (i == 0 || j == 0 || i == m - 1 || j == n - 1)
                tong = tong + a[i][j];
        }
    }
    return tong;
}
```

KỸ THUẬT XỬ LÝ TRÊN BIÊN

Bài tập 1: Tính tổng các phần từ nằm ở biên ma trận số nguyên.

Bài tập 2: .Tìm số hoàn thiện cuối cùng trên biên ma trận số nguyên.

Bài tập 3: Hãy đánh dấu các số trên biên ma trận theo hình vuông từ 1 -> k.

Bài tập 4: Cho số bắt đầu của biên 1 ma trận là 1 tính giá trị tiếp theo của ma trận nguyên bằng tổng các giá trị trước đó.

Bài tập 5: Hãy sắp xếp ma trận theo biên.

3. KỸ THUẬT XỬ LÝ TRÊN DÒNG

Định nghĩa: là kỹ thuật dùng tính toán theo yêu cầu đề bài nhưng chỉnh thao tác trên dòng bất kỳ.

	0	1	2	3	4
0	-1	2	-7	14	-5
1	25	-7	8	25	10
2	11	12	18	6	-2
3	4	17	18	9	33
4	2	2	3	4	25
5	6	13	22	11	9

3. KỸ THUẬT XỬ LÝ TRÊN DÒNG

Ví dụ minh họa: Tính tổng 1 dòng bất kỳ trong ma trận khi người dùng nhập vào.

```
int Tong1Dong(int a[][MAX], int m, int n, int dong)
{
    int tong = 0;
    for (int j = 0; j < n; j++)
    {
        tong = tong + a[dong][j];
    }
    return tong;
}
```

KỸ THUẬT XỬ LÝ TRÊN DÒNG

Bài tập 1: Tính tổng từng dòng. Trên ma trận số nguyên.

Bài tập 2: Liệt kê các dòng có chứa số 0. Trên ma trận số nguyên.

Bài tập 3: Tìm giá trị lớn nhất trên từng dòng. Trên ma trận số nguyên.

Bài tập 4: Tìm dòng chứa nhiều số chẵn nhất. Trên ma trận số nguyên.

Bài tập 5: Kiểm tra xem trên ma trận số nguyên có dòng nào tăng dần hay không?

4. KỸ THUẬT XỬ LÝ TRÊN CỘT

Định nghĩa: là kỹ thuật dùng tính toán theo yêu cầu đề bài nhưng chỉnh thao tác trên dòng bất kỳ.

	0	1	2	3	4
0	-1	2	-7	14	-5
1	25	-7	8	25	10
2	11	12	18	6	-2
3	4	17	18	9	33
4	2	2	3	4	25
5	6	13	22	11	9

4. KỸ THUẬT XỬ LÝ TRÊN CỘT

Ví dụ minh họa: Xóa một cột bất kỳ trong ma trận số nguyên do người dùng nhập vào.

```
void XoaCot(int a[][100], int m, int& n, int cot)
{
    for (int i = 0; i < m; i++)
        for (int j = cot; j < n - 1; j++)
            a[i][j] = a[i][j + 1];
    n--;
}
```

KỸ THUẬT XỬ LÝ TRÊN CỘT

Bài tập 1: Liệt kê các cột toàn âm trong ma trận các số nguyên.

Bài tập 2: Tìm trung bình cộng từng cột của ma trận số nguyên.

Bài tập 3: Thay toàn bộ giá trị trên những cột lẽ bằng số đầu tiên của cột đó.

Bài tập 4: Tìm dòng nào có chứa nhiều số chẵn nhất. Trong ma trận số nguyên.

Bài tập 5: Kiểm tra xem trên ma trận số nguyên có cột nào giảm dần hay không?

5. KỸ THUẬT HOÁN VỊ TRÊN MA TRẬN

Định nghĩa: là kỹ thuật dùng để hoán đổi các dòng và các cột với nhau, hoặc hoán đổi 2 vị trí bất kỳ.

	0	1	2	3	4
0	-1	2	-7	14	-5
1	25	-7	8	25	10
2	11	12	18	6	-2
3	4	17	18	9	33
4	2	2	3	4	25
5	6	13	22	11	9

5. KỸ THUẬT HOÁN VỊ TRÊN MA TRẬN

```
void HoanVi(int& a, int& b)
{
    int t = a;
    a = b;
    b = t;
}
```

```
void HoanViDong(int a[][MAX], int m, int n, int dong1, int dong2)
{
    if ((dong1 >= 0 && dong1 < m) && (dong2 >= 0 && dong2 < m))
    {
        for (int j = 0; j < n; j++)
            HoanVi(a[dong1][j], a[dong2][j]);
    }
}
```

6. KỸ THUẬT SẮP XẾP MA TRẬN

Định nghĩa: là kỹ thuật sắp xếp các phần tử trên ma trận theo một thứ tự nào đó theo yêu cầu đề bài.

	0	1	2	3	4
0	1	15	8	16	9
1	14	23	10	17	3
2	5	13	22	4	21
3	20	24	2	27	19
4	11	12	26	30	7
5	28	25	6	18	29

	0	1	2	3	4
0	1	2	3	4	5
1	6	7	8	9	10
2	11	12	13	14	15
3	16	17	18	19	20
4	21	22	23	24	25
5	26	27	28	29	30

6. KỸ THUẬT SẮP XẾP MA TRẬN

Định nghĩa: là kỹ thuật sắp xếp các phần tử trên ma trận theo một thứ tự nào đó theo yêu cầu đề bài.

	0	1	2	3	4
0	1	15	8	16	9
1	14	23	10	17	3
2	5	13	22	4	21
3	20	24	2	27	19
4	11	12	26	30	7
5	28	25	6	18	29

	0	1	2	3	4
0	1	2	3	4	5
1	10	9	8	7	6
2	11	12	..		
3					
4					
5					

6. KỸ THUẬT SẮP XẾP MA TRẬN

Ví dụ minh họa: Sắp xếp ma trận theo thứ tự tăng dần từ trái qua phải và từ trên xuống dưới.

```
void SapXepTrenDuoiTraiPhai(int a[][MAXN], int m, int n)
{
    int b[MAXN*MAXM];
    int k = 0;
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            b[k++] = a[i][j];
        }
    }
    insertionSort(b, k);
    k = 0;
    for (int i = 0; i < m; i++)
        for (int j = 0; j < n; j++)
            a[i][j] = b[k++];
}
```

7. KỸ THUẬT XỬ LÝ MA TRẬN CON

Định nghĩa: là kỹ thuật tìm một ma trận con trong ma trận ban đầu để đáp ứng yêu cầu nào đó.

```
a[i][j] + a[i + 1][j] + a[i][j + 1] + a[i + 1][j + 1];
```

	0	1	2	3	4
0	1	15	8	16	9
1	14	23	10	17	3
2	5	13	22	4	21
3	20	24	2	27	19
4	11	12	26	30	7
5	28	25	6	18	29

7. KỸ THUẬT XỬ LÝ MA TRẬN CON

Ví dụ minh họa: Tìm 1 ma trận con bất kỳ có kích thước 2x2 tổng các phần tử lớn hơn 50.

	0	1	2	3	4
0	1	15	8	16	9
1	14	23	10	17	3
2	5	13	22	4	21
3	20	24	2	27	19
4	11	12	26	30	7
5	28	25	6	18	29

7. KỸ THUẬT XỬ LÝ MA TRẬN CON

Ví dụ minh họa: Tìm 1 ma trận con bất kỳ có kích thước 2x2 tổng các phần tử lớn hơn 50.

```
bool MaTranConLonHon50(int a[][MAXN], int m, int n)
{
    int Tong = 0;
    for (int i = 0; i < m-1; i++)
    {
        for (int j = 0; j < n-1; j++)
        {
            Tong = a[i][j] + a[i + 1][j] + a[i][j + 1] + a[i + 1][j + 1];
            if (Tong > 50)
            {
                cout << "("<<i << ", " << j<<")" <<" ";
                cout << "(" << i + 1<< ", " << j << ")"<<endl;
                cout << "(" << i << ", " << j + 1<< ")"<<" ";
                cout << "(" << i + 1 << ", " << j + 1 << ")"<<endl;
                return true;
            }
        }
    }
    return false;
}
```

8. KỸ THUẬT XỬ LÝ TRÊN MA TRẬN VUÔNG

Định nghĩa: Ma trận vuông là ma trận có số dòng và số cột bằng nhau và có thêm một số định nghĩa sau.

Đường chéo chính

	0	1	2	3	4	5
0	-1	2	-7	14	-5	5
1	25	-7	8	25	10	0
2	11	12	18	6	-2	-1
3	4	17	18	9	33	3
4	2	2	3	4	25	6
5	6	13	22	11	9	8

Đường chéo phụ

	0	1	2	3	4	5
0	-1	2	-7	14	-5	5
1	25	-7	8	25	10	0
2	11	12	18	6	-2	-1
3	4	17	18	9	33	3
4	2	2	3	4	25	6
5	6	13	22	11	9	8

8. KỸ THUẬT XỬ LÝ TRÊN MA TRẬN VUÔNG

```
void NhapMaTranVuong(int a[][MAXN], int& n)
{
    cout << "Nhap n: ";
    cin >> n;
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            cout << "Nhap a[" << i << "][" << j << "]: ";
            cin >> a[i][j];
        }
    }
}
```

8. KỸ THUẬT XỬ LÝ TRÊN MA TRẬN VUÔNG

```
void XuatMaTranVuong(int a[][MAXN], int n)
{
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            cout << a[i][j] << " ";
        }
        cout << endl;
    }
}
```

8. KỸ THUẬT XỬ LÝ TRÊN MA TRẬN VUÔNG

Đường chéo chính

	0	1	2	3	4	5
0	-1	2	-7	14	-5	5
1	25	-7	8	25	10	0
2	11	12	18	6	-2	-1
3	4	17	18	9	33	3
4	2	2	3	4	25	6
5	6	13	22	11	9	8

Ma trận vuông đường chéo chính chia làm 3 khu vực:

- KV1: dòng < cột.
- KV2: dòng = cột.
- KV3: dòng > cột.

8. KỸ THUẬT XỬ LÝ TRÊN MA TRẬN VUÔNG

Đường chéo phụ

	0	1	2	3	4	5
0	-1	2	-7	14	-5	5
1	25	-7	8	25	10	0
2	11	12	18	6	-2	-1
3	4	17	18	9	33	3
4	2	2	3	4	25	6
5	6	13	22	11	9	8

Ma trận vuông đường chéo phụ chia làm 3 khu vực:

- KV1: dòng + cột < n - 1.
- KV2: dòng + cột = n - 1.
- KV3: dòng + cột > n - 1.

8. KỸ THUẬT XỬ LÝ TRÊN MA TRẬN VUÔNG

Ví dụ minh họa: Tính tổng tam giác trên của đường chéo chính.

```
int TongTamGiacTrenDuongCheoChinh(int a[][MAXN], int n)
{
    int tong = 0;
    for (int i = 0; i < n - 1; i++)
    {
        for (int j = i + 1; j < n; j++)
        {
            tong += a[i][j];
        }
    }
    return tong;
}
```