```
/*
	Bài 01. Hãy viết chương trình:

			a. Nhập vào số dòng và cột cần lưu trữ
			   của một mảng 2 chiều lưu trữ toàn số nguyên
			   (tối đa 20 dòng và 10 cột).
			   Nếu sai hãy yêu cầu nhập lại cho đến khi thỏa điều kiện.
			   Nhập vào giá trị từng phần tử trong mảng.

			b. Xuất lại mảng đã nhập.

			c. Tính tích các phần tử tại 1 dòng nào đó
			   do người dùng yêu cầu.
*/

#include <iostream>
using namespace std;

// Khai báo tĩnh số dòng và số cột tối đa
const int SO_DONG_TOI_DA = 20, SO_COT_TOI_DA = 10;

// Mặc định, khi mảng được truyền vào trong hàm
// sẽ là truyền tham chiếu (trong hàm thay đổi giá trị
// thì nơi gọi hàm cũng sẽ bị ảnh hưởng theo)

// Khai báo các nguyên mẫu hàm
void nhapMang(int a[SO_DONG_TOI_DA][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe);
void xuatMang(int a[SO_DONG_TOI_DA][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe);
int tinhTichMotDongNaoDo(int a[SO_DONG_TOI_DA][SO_COT_TOI_DA],
                                        int soCotThucTe, int dongCanTinhTich);

int main()
{
	// Khai báo mảng a
	// với số dòng tối đa là SO_DONG_TOI_DA
	// và số cột tối đa là SO_COT_TOI_DA
	int a[SO_DONG_TOI_DA][SO_COT_TOI_DA];

	// Trong thực tế, có thể không sử dụng hết 20 dòng và 10 cột.
	// Vì vậy, sử dụng thêm hai biến để lưu số dòng và cột
	// mà người dùng thực tế cần sử dụng
	int soDongThucTe, soCotThucTe;

	// Dòng nào đó cần tính tích
	// do người dùng yêu cầu.
	int dongCanTinhTich;

	// Kiểm soát giá trị khi người dùng nhập số dòng thực tế
	do {
		cout << "================================================\n";
		cout << "Nhap so dong can su dung (1..20): ";
		cin >> soDongThucTe;
		cout << endl;

		if (soDongThucTe <= 0 || soDongThucTe > SO_DONG_TOI_DA)
			cout << "Vui long nhap: 0 < So dong <= 20\n";
	} while (soDongThucTe <= 0 || soDongThucTe > SO_DONG_TOI_DA);

	cout << "Tong so dong dang su dung la: " << soDongThucTe << endl;

	// Kiểm soát giá trị khi người dùng nhập số cột thực tế
	do {
```

```
            cout << "================================================\n";
            cout << "Nhap so cot can su dung (1..10): ";
            cin >> soCotThucTe;
            cout << endl;

            if (soCotThucTe <= 0 || soCotThucTe > SO_COT_TOI_DA)
                  cout << "Vui long nhap: 0 < So cot <= 10\n";
      } while (soCotThucTe <= 0 || soCotThucTe > SO_COT_TOI_DA);

      cout << "Tong so cot dang su dung la: " << soCotThucTe << endl;
      cout << endl;

      // Gọi hàm nhập mảng và tuyền tham số vào
      cout << "Vui long nhap mang hai chieu\n" << endl;
      nhapMang(a, soDongThucTe, soCotThucTe);
      cout << endl;

      // Gọi hàm xuất mảng và tuyền tham số vào
      cout << "Mang hai chieu la:\n" << endl;
      xuatMang(a, soDongThucTe, soCotThucTe);
      cout << endl;

      // Kiểm soát giá trị khi người dùng nhập số dòng cần tính tích
      do {
            cout << "Tong so dong dang su dung la: " << soDongThucTe << endl;
            cout << "Tong so cot dang su dung la: " << soCotThucTe << endl;
            cout << endl;

            cout << "Nhap dong can tinh tich (1.." << soDongThucTe << "): ";
            cin >> dongCanTinhTich;
            cout << endl;

            if (dongCanTinhTich < 1 || dongCanTinhTich > soDongThucTe)
                  cout << "Vui long nhap so dong (1.." << soDongThucTe << ")!\n";
      } while (dongCanTinhTich < 1 || dongCanTinhTich > (soDongThucTe));

      // Gọi hàm tính tích của một dòng nào đó
      cout << "Tich cua dong " << dongCanTinhTich
            << " la: " << tinhTichMotDongNaoDo(a, soCotThucTe, dongCanTinhTich);
      cout << endl;

      return 0;
}

// a. Hàm nhập mảng
void nhapMang(int a[SO_DONG_TOI_DA][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe)
{
      // Để gọi được hàm này,
      // nơi gọi đã phải có những giá trị tương ứng
      // như những gì mà tham số yêu cầu

      // Nhập từng giá trị cho mảng hai chiều,
      // cần phải biết giá trị đó nằm ở dòng và cột nào

      // Sử dụng hai vòng lặp lồng nhau
      // để duyệt từng giá trị

      // Duyệt từng dòng
      for (int i = 0; i < soDongThucTe; i++)
            // Duyệt từng cột (của từng dòng)
            for (int j = 0; j < soCotThucTe; j++)
            {
```

```
                // Yêu cầu nhập giá trị
                // cho từng phần tử trong mảng
                cout << "Vui long nhap phan tu a[" << i + 1 << "][" << j + 1 << "]: ";
                cin >> a[i][j];
        }
} // nhapMang()

// b. Hàm xuất mảng
void xuatMang(int a[SO_DONG_TOI_DA][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe)
{
      // Duyệt từng dòng
      for (int i = 0; i < soDongThucTe; i++)
      {
            // Duyệt từng cột (của từng dòng)
            for (int j = 0; j < soCotThucTe; j++)
                  // In giá trị vừa mới duyệt ra
                  cout << a[i][j] << "\t";

            // Sau khi in ra hết số cột trên một dòng,
            // mới tiến hành xuống hàng để in dòng mới
            cout << endl;
      }
} // xuatMang()

// c. hàm tính tích các phần tử tại 1 dòng nào đó
int tinhTichMotDongNaoDo(int a[SO_DONG_TOI_DA][SO_COT_TOI_DA],
                                          int soCotThucTe, int dongCanTinhTich)
{
      // Tính tích các phần tử tại 1 dòng nào đó
      // do người dùng yêu cầu.
      int ketQuaTinhTichDong = 1;

      // Sau khi đã chọn được dòng cần tính tích,
      // cố định dòng đó lại
      // và duyệt toàn bộ phần tử theo cột (từ trái sang phải)
      // để tính tích
      for (int j = 0; j < soCotThucTe; j++)
                  ketQuaTinhTichDong = ketQuaTinhTichDong * a[dongCanTinhTich - 1][j];

      return ketQuaTinhTichDong;
} // tinhTichMotDongNaoNo()




/*
      Bài 02. Hãy viết chương trình dùng mảng 2 chiều
                  xuất ra giá trị dãy số có dạng như sau:

                        1    2    3    4    5
                        6    7    8    9    10
                        11   12   13   14   15
                        16   17   18   19   20
*/

#include <iostream>
using namespace std;

const int SO_DONG_TOI_DA = 20, SO_COT_TOI_DA = 10;

void nhapMaTran(int a[SO_DONG_TOI_DA][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe);
void xuatMaTran(int a[SO_DONG_TOI_DA][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe);
```

```
int main()
{
	int a[SO_DONG_TOI_DA][SO_COT_TOI_DA];
	int soDongThucTe, soCotThucTe;

	// Kiểm soát giá trị khi người dùng nhập số dòng thực tế
	do {
		cout << "Nhap so dong can su dung (1..20): ";
		cin >> soDongThucTe;
		cout << endl;

		if (soDongThucTe <= 0 || soDongThucTe > SO_DONG_TOI_DA)
		{
			cout << "Gia tri nhap khong hop le!\n";
			cout << "Vui long nhap: 0 < So dong <= 20\n";
			cout << endl;
		}
	} while (soDongThucTe <= 0 || soDongThucTe > SO_DONG_TOI_DA);

	// Kiểm soát giá trị khi người dùng nhập số cột thực tế
	do {
		cout << "Nhap so cot can su dung (1..10): ";
		cin >> soCotThucTe;
		cout << endl;

		if (soCotThucTe <= 0 || soCotThucTe > SO_COT_TOI_DA)
		{
			cout << "Gia tri nhap khong hop le!\n";
			cout << "Vui long nhap: 0 < So cot <= 10\n";
			cout << endl;
		}
	} while (soCotThucTe <= 0 || soCotThucTe > SO_COT_TOI_DA);

	cout << "Vui long nhap ma tran hai chieu\n" << endl;
	nhapMaTran(a, soDongThucTe, soCotThucTe);
	cout << endl;

	cout << "Ma tran hai chieu la:\n" << endl;
	xuatMaTran(a, soDongThucTe, soCotThucTe);
	cout << endl;

	return 0;
}

// 01. Hàm nhập vào ma trận hai chiều
void nhapMaTran(int a[SO_DONG_TOI_DA][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe)
{
	int giaTri = 1;

	for (int i = 0; i < soDongThucTe; i++)
	{
		for (int j = 0; j < soCotThucTe; j++)
			a[i][j] = giaTri++;
	}
}

// 02. Hàm in ra ma trận hai chiều
void xuatMaTran(int a[SO_DONG_TOI_DA][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe)
{
	for (int i = 0; i < soDongThucTe; i++)
	{
```

```
			for (int j = 0; j < soCotThucTe; j++)
				cout << a[i][j] << "\t";
			cout << endl;
		}
}




/*
	Bài 03. Hãy viết chương trình dùng câu lệnh while
			để nhập và xuất giá trị một mảng số nguyên
			gồm 5 hàng và 3 cột.
*/

#include <iostream>
using namespace std;

void nhapMang(int a[5][3]);
void xuatMang(int a[5][3]);

int main()
{
	int a[5][3];

	cout << "\t--- Ma tran 5 hang và 3 cot ---\n";
	cout << endl;

	cout << "Nhap ma tran hai chieu\n" << endl;
	nhapMang(a);
	cout << endl;

	cout << "Ma tran hai chieu la:\n" << endl;
	xuatMang(a);
	cout << endl;

	return 0;
}

// 01. Hàm nhập mảng
void nhapMang(int a[5][3])
{
	// Duyệt từng dòng
	int i = 0;
	while(i < 5)
	{
		// Duyệt từng cột (của từng dòng)
		int j = 0;

		// Nhập một loạt tất cả các phần tử của một dòng
		cout << "Nhap mot loat 3 phan tu cua dong thu [" << i + 1 << "]: ";

		while (j < 3)
		{
			// Đưa một loạt các giá trị vào một dòng
			cin >> a[i][j];
			j++;
		}

		i++;
	}
```

```
}

// 02. Hàm xuất mảng
void xuatMang(int a[5][3])
{
	// Duyệt từng dòng
	int i = 0;
	while (i < 5)
	{
		// Duyệt từng cột (của từng dòng)
		int j = 0;
		while (j < 3)
		{
			// In giá trị vừa mới duyệt ra
			cout << a[i][j] << "\t";
			j++;
		}

		// Sau khi in ra hết số cột trên một dòng,
		// mới tiến hành xuống hàng để in dòng mới
		cout << endl;

		i++;
	}
}




/*
	Bài 04. Viết chương trình cho phép nhập vào
			một mảng số nguyên tối đa 10 dòng và 5 cột.
			Sau đó tìm xem một giá trị x nào đó
			(do người dùng nhập) có trong mảng hay không?
			Bao nhiêu lần? Vị trí đầu tiên xuất hiện là ở đâu?
*/

#include <iostream>
using namespace std;

const int SO_DONG_TOI_DA = 10, SO_COT_TOI_DA = 5;

void nhapMang(int a[][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe);
void xuatMang(int a[][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe);
void kiemTraGiaTriCoTrongMang(int a[][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe,
                                                    int giaTriCanTim, int &viTriDong, int &viTriCot);

int main()
{
	int a[SO_DONG_TOI_DA][SO_COT_TOI_DA];

	int soDongThucTe, soCotThucTe;

	int giaTriCanTim;

	int viTriDong, viTriCot;

	// Kiểm soát giá trị khi người dùng nhập số dòng thực tế
	do {
		cout << "Nhap so dong can su dung (1..10): ";
		cin >> soDongThucTe;
```

```
		cout << endl;

		if (soDongThucTe <= 0 || soDongThucTe > SO_DONG_TOI_DA)
		{
			cout << "Gia tri nhap khong hop le!\n";
			cout << "Vui long nhap: 0 < So dong <= 10\n";
			cout << endl;
		}
	} while (soDongThucTe <= 0 || soDongThucTe > SO_DONG_TOI_DA);

	// Kiểm soát giá trị khi người dùng nhập số cột thực tế
	do {
		cout << "Nhap so cot can su dung (1..5): ";
		cin >> soCotThucTe;
		cout << endl;

		if (soCotThucTe <= 0 || soCotThucTe > SO_COT_TOI_DA)
		{
			cout << "Gia tri nhap khong hop le!\n";
			cout << "Vui long nhap: 0 < So cot <= 5\n";
			cout << endl;
		}
	} while (soCotThucTe <= 0 || soCotThucTe > SO_COT_TOI_DA);

	cout << "Vui long nhap mang hai chieu\n" << endl;
	nhapMang(a, soDongThucTe, soCotThucTe);
	cout << endl;

	cout << "Mang hai chieu la:\n" << endl;
	xuatMang(a, soDongThucTe, soCotThucTe);
	cout << endl;

	cout << "Tim mot gia tri xem co trong mang hay khong\n" << endl;
	cout << "Vui long nhap gia tri can tim: ";
	cin >> giaTriCanTim;
	cout << endl;

	kiemTraGiaTriCoTrongMang(a, soDongThucTe, soCotThucTe,
		giaTriCanTim, viTriDong, viTriCot);
	cout << endl;

	return 0;
}

// 01. Hàm nhập mảng
void nhapMang(int a[][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe)
{
	for (int i = 0; i < soDongThucTe; i++)
	{
		cout << "Hay nhap cung luc " << soCotThucTe
			<< " phan tu cua dong thu [" << i + 1
			<< "]: ";
		for (int j = 0; j < soCotThucTe; j++)
			cin >> a[i][j];
	}
}

// 02. Hàm xuất mảng
void xuatMang(int a[][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe)
{
	for (int i = 0; i < soDongThucTe; i++)
	{
```

```
            for (int j = 0; j < soCotThucTe; j++)
                  cout << a[i][j] << "\t";

            cout << endl;
      }
}

// 03. Hàm tìm giá trị một giá trị có trong mảng,
//       nếu có thì xuất hiện bao nhiêu lần
//       và vị trí đầu tiên xuất hiện là ở đâu?
void kiemTraGiaTriCoTrongMang(int a[][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe,
                                                      int giaTriCanTim, int &viTriDong, int &viTriCot)
{
      int demSoLanXuatHien = 0;

      for (int i = 0; i < soDongThucTe; i++)
            for (int j = 0; j < soCotThucTe; j++)
                  if (a[i][j] == giaTriCanTim)
                  {
                        demSoLanXuatHien++;

                        if (demSoLanXuatHien == 1)
                        {
                              viTriDong = i;
                              viTriCot = j;
                        }
                  }

      if (demSoLanXuatHien == 0)
            cout << "Khong tim thay " << giaTriCanTim << " trong mang!\n" << endl;
      else
      {
            cout << "Da tim thay " << giaTriCanTim
                  << " xuat hien " << demSoLanXuatHien << " lan!\n" << endl;

            cout << "Vi tri dau tien xuat hien la tai dong [" << viTriDong + 1 << "]"
                  << " va cot [" << viTriCot + 1 << "]\n"<< endl;
      }
}




/*
      - Bài tập thêm -

      Bài 05. Xây dựng hàm nhập, xuất, tính tổng
                  và đếm xem mảng vừa nhập
                  có bao nhiêu phần tử là số nguyên tố
*/

#include <iostream>
#include <cmath> // Khai báo thêm thư viện để sử dụng hàm sqrt()
using namespace std;

const int SO_DONG_TOI_DA = 20, SO_COT_TOI_DA = 10;

void nhapMang(int a[][SO_COT_TOI_DA], int &soDongThucTe, int &soCotThucTe);
void xuatMang(int a[][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe);
int tinhTongCuaMang(int a[][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe);
bool laSoNguyenTo(int soNguyen);
int demSoLuongSoNguyenTo(int a[][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe);
```

```
void intCacSoLuongSoNguyenTo(int a[][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe);

int main()
{
	int a[SO_DONG_TOI_DA][SO_COT_TOI_DA];

	int soDongThucTe, soCotThucTe;

	// Kiểm soát giá trị khi người dùng nhập số dòng thực tế
	do {
		cout << "Nhap so dong can su dung (1..20): ";
		cin >> soDongThucTe;
		cout << endl;

		if (soDongThucTe <= 0 || soDongThucTe > SO_DONG_TOI_DA)
		{
			cout << "Gia tri nhap khong hop le!\n";
			cout << "Vui long nhap: 0 < So dong <= 20\n";
			cout << endl;
		}
	} while (soDongThucTe <= 0 || soDongThucTe > SO_DONG_TOI_DA);

	// Kiểm soát giá trị khi người dùng nhập số cột thực tế
	do {
		cout << "Nhap so cot can su dung (1..10): ";
		cin >> soCotThucTe;
		cout << endl;

		if (soCotThucTe <= 0 || soCotThucTe > SO_COT_TOI_DA)
		{
			cout << "Gia tri nhap khong hop le!\n";
			cout << "Vui long nhap: 0 < So cot <= 10\n";
			cout << endl;
		}
	} while (soCotThucTe <= 0 || soCotThucTe > SO_COT_TOI_DA);

	cout << "Vui long nhap mang hai chieu\n" << endl;
	nhapMang(a, soDongThucTe, soCotThucTe);
	cout << endl;

	cout << "Mang hai chieu la:\n" << endl;
	xuatMang(a, soDongThucTe, soCotThucTe);
	cout << endl;

	// Gọi hàm tính tổng và truyền tham số vào
	cout << "Mang co tong cac phan tu la: "
		<< tinhTongCuaMang(a, soDongThucTe, soCotThucTe);
	cout << endl;

	// Gọi hàm đếm số lượng số nguyên tố và tuyền tham số vào
	cout << "Mang vua nhap co tong cong "
		<< demSoLuongSoNguyenTo(a, soDongThucTe, soCotThucTe)
		<< " so nguyen to\n";
	cout << endl;

	cout << "Cac so nguyen to lan luot la: ";
	intCacSoLuongSoNguyenTo(a, soDongThucTe, soCotThucTe);
	cout << endl;

	return 0;
}
```

```
// 01. Hàm nhập mảng
void nhapMang(int a[][SO_COT_TOI_DA], int &soDongThucTe, int &soCotThucTe)
{
	for (int i = 0; i < soDongThucTe; i++)
	{
		cout << "Nhap mot loat "
			<< soCotThucTe << " phan tu tren dong thu [" << i + 1 << "]: ";
		for (int j = 0; j < soCotThucTe; j++)
			cin >> a[i][j];
	}
	cout << endl;
}

// 02. Hàm xuất mảng
void xuatMang(int a[][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe)
{
	for (int i = 0; i < soDongThucTe; i++)
	{
		for (int j = 0; j < soCotThucTe; j++)
			cout << a[i][j] << "\t";

		cout << endl;
	}
}

// 03. Hàm tính tổng
int tinhTongCuaMang(int a[][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe)
{
	// Biến trung gian để lưu kết quả tính tổng,
	// khởi tạo giá trị ban đầu = 0
	// để tránh gặp phải những giá trị rác
	// trong quá trình cộng dồn và tính tổng
	int ketQuaTinhTong = 0;

	// Duyệt từng dòng
	for (int i = 0; i < soDongThucTe; i++)
		// Duyệt từng cột (của từng dòng)
		for (int j = 0; j < soCotThucTe; j++)
			// Nhồi liên tục để tính tổng
			ketQuaTinhTong += a[i][j];

	return ketQuaTinhTong;
}

// 04. Hàm kiểm tra một số có phải là số nguyên tố
bool laSoNguyenTo(int soNguyen)
{
	// Số nguyên tố là số bắt đầu từ 2,
	// chỉ chia được hết cho 1 và chính nó

	// Nếu nhận vào một số nhỏ hơn 2
	if (soNguyen < 2)
		// Không phải là số nguyên tố
		return false;
	else if (soNguyen == 2) // Nếu nhận vào một số = 2
		// Là số nguyên tố
		// (số 2 là số nguyên tố đầu tiên)
		return true;
	else // Nếu nhận vào một số > 2
	{
		// Sử dụng vòng lặp để kiểm tra.
```

```
		// Mọi số đều sẽ chia hết cho 1 và chính nó,
		// nhưng quan trọng là ngoài hai số đó ra,
		// thì nó không còn chia được hết cho số nào khác
		// (phải sử dụng i <= ...,
		//  nếu không sẽ bị sót trường hợp soNguyen = 4)
		for (int i = 2; i <= sqrt(soNguyen * 1.0); i++)

			// Nếu số nhận vào "lỡ" chia cho số nào khác
			if (soNguyen % i == 0)
				return false;
	}

	// Đi đến hết chương trình mà không gặp bất kì return nào,
	// thì đích thị đây là số nguyên tố
	return true;
}

// 05. Hàm đếm số lượng số nguyên tố trong mảng
int demSoLuongSoNguyenTo(int a[][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe)
{
	// Biến trung gian, sử dụng để đếm,
	// ban đầu khởi tạo = 0
	// để xem như chưa đếm được giá trị nào
	int tongSoLuongSoNguyenTo = 0;

	// Duyệt từng dòng
	for (int i = 0; i < soDongThucTe; i++)
		// Duyệt từng cột (của từng dòng)
		for (int j = 0; j < soCotThucTe; j++)
			// Lấy từng phần tử ra và đưa vào hàm laSoNguyenTo()
			// để kiểm tra xem có phải là số nguyên tố
			if (laSoNguyenTo(a[i][j]) == true)
				// Nếu phát hiện một số nào đó là số nguyên tố
				// thì tăng tổng số lượng lên 1
				tongSoLuongSoNguyenTo++;

	return tongSoLuongSoNguyenTo;
}

// 06. Hàm in ra các số lượng số nguyên tố trong mảng
void intCacSoLuongSoNguyenTo(int a[][SO_COT_TOI_DA], int soDongThucTe, int soCotThucTe)
{
	for (int i = 0; i < soDongThucTe; i++)
		for (int j = 0; j < soCotThucTe; j++)
			if (laSoNguyenTo(a[i][j]) == true)
				cout << a[i][j] << "  ";
}




/*
	- Bài tập thêm -

	Bài 06. Hãy viết chương trình dùng mảng 2 chiều
			xuất ra giá trị dãy số có dạng như sau:

				1   12   11   10
				2   13   16   9
				3   14   15   8
				4   5    6    7
*/
```

```cpp
#include <iostream>
using namespace std;

const int CAP_CUA_MA_TRAN = 10;

void nhapMaTran(int a[][CAP_CUA_MA_TRAN], int soCapThucTeCuaMaTran);
void xuatMaTran(int a[][CAP_CUA_MA_TRAN], int soCapThucTeCuaMaTran);

int main()
{
	int a[CAP_CUA_MA_TRAN][CAP_CUA_MA_TRAN];
	int soCapThucTeCuaMaTran;

	// Kiểm soát giá trị khi người dùng nhập số cấp thực tế
	do {
		cout << "Nhap so cap can su dung (1..10): ";
		cin >> soCapThucTeCuaMaTran;
		cout << endl;

		if (soCapThucTeCuaMaTran <= 0 || soCapThucTeCuaMaTran > CAP_CUA_MA_TRAN)
		{
			cout << "Gia tri nhap khong hop le!\n";
			cout << "Vui long nhap: 0 < So cap <= 10\n";
			cout << endl;
		}
	} while (soCapThucTeCuaMaTran <= 0 || soCapThucTeCuaMaTran > CAP_CUA_MA_TRAN);

	cout << "Nhap ma tran hai chieu\n" << endl;
	nhapMaTran(a, soCapThucTeCuaMaTran);
	cout << endl;

	cout << "Ma tran hai chieu la:\n" << endl;
	xuatMaTran(a, soCapThucTeCuaMaTran);
	cout << endl;

	return 0;
}

// 01. Hàm nhập mảng
void nhapMaTran(int a[][CAP_CUA_MA_TRAN], int soCapThucTeCuaMaTran)
{
	int giaTri = 1;

	int minDong, maxDong;
	int minCot, maxCot;

	minDong = minCot = 0;
	maxDong = maxCot = soCapThucTeCuaMaTran - 1;

	while (giaTri <= soCapThucTeCuaMaTran * soCapThucTeCuaMaTran)
	{
		// Hướng 01 - Từ trên xuống dưới
		// minCot cố định, minDong -> maxDong
		for (int i = minDong; i <= maxDong; i++)
			a[i][minCot] = giaTri++;
		minCot++;

		// Hướng 02 - Từ trái sang phải
		// maxDong cố định, minCot -> maxCot
		for (int j = minCot; j <= maxCot; j++)
			a[maxDong][j] = giaTri++;
```

```
            maxDong--;

            // Hướng 03 - Từ dưới lên trên
            // maxCot cố định, maxDong -> minDong
            for (int i = maxDong; i >= minDong; i--)
                  a[i][maxCot] = giaTri++;
            maxCot--;

            // Hướng 04 - Từ phải sang trái
            // minDong cố định, maxCot -> minCot
            for (int j = maxCot; j >= minCot; j--)
                  a[minDong][j] = giaTri++;
            minDong++;
      }
}

// 02. Hàm xuất mảng
void xuatMaTran(int a[][CAP_CUA_MA_TRAN], int soCapThucTeCuaMaTran)
{
      for (int i = 0; i < soCapThucTeCuaMaTran; i++)
      {
            for (int j = 0; j < soCapThucTeCuaMaTran; j++)
                  cout << a[i][j] << "\t";

            cout << endl;
      }
}




/*
      Bài 01. Viết 01 chương trình thực hiện toàn bộ các công việc sau:

                  a. Khai báo một mảng số nguyên gồm 3 hàng và 4 cột.

                  b. Khởi tạo giá trị của mảng vừa nhập lần lượt như sau:
                              8    4  -1 5
                              2    2     6     9
                             11    2     5     4

                  c. Xuất lại toàn bộ mảng đang lưu trữ để kiểm chứng.
*/

#include <iostream>
using namespace std;

// Khai số dòng và số cột tối đa
const int DONG_TOI_DA = 3, COT_TOI_DA = 4;

void nhapMang(int a[DONG_TOI_DA][COT_TOI_DA]);
void xuatMang(int a[DONG_TOI_DA][COT_TOI_DA]);

int main()
{
      int a[DONG_TOI_DA][COT_TOI_DA];

      // Gọi hàm nhapMang()
      nhapMang(a);
      cout << endl;
```

```
	// Gọi hàm xuatMang()
	xuatMang(a);
	cout << endl;

	return 0;
} // main()

// a. Khai báo một mảng số nguyên gồm 3 hàng và 4 cột.
// b. Khởi tạo giá trị của mảng vừa nhập.
void nhapMang(int a[DONG_TOI_DA][COT_TOI_DA])
{
	for (int i = 0; i < DONG_TOI_DA; i++)
	{
		cout << "Nhap " << COT_TOI_DA
			<< " so nguyen cho dong thu [" << i + 1 << "]: ";
		for (int j = 0; j < COT_TOI_DA; j++)
			cin >> a[i][j];
	}
} // nhapMang()

// c. Xuất lại toàn bộ mảng đang lưu trữ để kiểm chứng.
void xuatMang(int a[DONG_TOI_DA][COT_TOI_DA])
{
	cout << "Mang vua nhap la: \n";
	for (int i = 0; i < DONG_TOI_DA; i++)
	{
		for (int j = 0; j < COT_TOI_DA; j++)
			cout << a[i][j] << "\t";
		cout << endl;
	}
} // xuatMang()




/*
	Bài 02. Viết chương trình dùng câu lệnh while
		để nhập và xuất giá trị cho một mảng số nguyên
		gồm 4 hàng và 3 cột.
*/

#include <iostream>
using namespace std;

const int DONG_TOI_DA = 3, COT_TOI_DA = 4;

void nhapMang(int a[DONG_TOI_DA][COT_TOI_DA]);
void xuatMang(int a[DONG_TOI_DA][COT_TOI_DA]);

int main()
{
	int a[DONG_TOI_DA][COT_TOI_DA];

	// Gọi hàm nhapMang()
	nhapMang(a);
	cout << endl;

	// Gọi hàm xuatMang()
	xuatMang(a);
	cout << endl;

	return 0;
```

```
} // main()

// 01. Hàm để nhập một mảng số nguyên
void nhapMang(int a[DONG_TOI_DA][COT_TOI_DA])
{
	int i = 0;
	while (i < DONG_TOI_DA)
	{
		int j = 0;
		cout << "Nhap " << COT_TOI_DA
			<< " so nguyen cho dong thu [" << i + 1 << "]: ";

		while (j < COT_TOI_DA)
		{
			cin >> a[i][j];
			j++;
		}
		i++;
	}
} // nhapMang()

// 02. Hàm để nhập một mảng số nguyên
void xuatMang(int a[DONG_TOI_DA][COT_TOI_DA])
{
	cout << "Mang vua nhap la: \n";

	int i = 0;
	while (i < DONG_TOI_DA)
	{
		for (int j = 0; j < COT_TOI_DA; j++)
			cout << a[i][j] << "\t";
		cout << endl;
		i++;
	}
} // xuatMang()




/*
	Bài 03. Viết 01 chương trình thực hiện toàn bộ các công việc sau:

		a. Cho phép người dùng nhập vào một mảng số nguyên gồm r hàng và c cột
		   (tối đa 10 hàng và 15 cột, nếu nhập sai thì yêu cầu nhập lại
		   cho đến khi đúng mới tiếp tục).

		b. Thực hiện việc tính tổng các giá trị lưu trữ trong mảng.

		c. Tìm giá trị nhỏ nhất, lớn nhất đang lưu trữ trong mảng.

		d. Cho biết vị trí của giá trị nhỏ nhất hay lớn nhất ở hàng mấy?
		   Cột bao nhiêu?
		   (Giả sử người dùng nhập không có giá trị nào trùng nhau).
*/

#include <iostream>
#include <conio.h>
using namespace std;

const int MAX_DONG = 10, MAX_COT = 15;
```

```
void nhapMang(int a[MAX_DONG][MAX_COT], int dong, int cot);
void xuatMang(int a[MAX_DONG][MAX_COT], int dong, int cot);
int tinhTong(int a[MAX_DONG][MAX_COT], int dong, int cot);
void timGiaTriNhoNhat(int a[MAX_DONG][MAX_COT], int dong, int cot, int &viTriDong, int &viTriCot);
void timGiaTriLonNhat(int a[MAX_DONG][MAX_COT], int dong, int cot, int &viTriDong, int &viTriCot);

int main()
{
	int a[MAX_DONG][MAX_COT], dong, cot;
	int luaChon;
	bool daNhapMang = false;

	do {
		system("cls");

		cout << endl;
		cout << "CHUONG TINH TINH TOAN TREN MANG HAI CHIEU\n"
			<< "1. Nhap mang\n"
			<< "2. Xuat mang (sau khi da nhap mang)\n"
			<< "3. Tinh tong (sau khi da nhap mang)\n"
			<< "4. Tim gia tri nho nhat va in ra vi tri (sau khi da nhap mang)\n"
			<< "5. Tim gia tri lon nhat va in ra vi tri (sau khi da nhap mang)\n"
			<< "6. Thoat chuong trinh\n"
			<< endl
			<< "Lua chon cua ban (1..6): ";
		cin >> luaChon;
		cout << endl;

		switch (luaChon)
		{
		case 1: // Nhập mảng
			// Kiểm tra số dòng nhập vào có hợp lệ?
			do {
				cout << "Nhap so dong cua ma tran (1..10): ";
				cin >> dong;

				if (dong <= 0 || dong > MAX_DONG)
					cout << "So dong khong hop le, vui long nhap lai (1..10)" << endl;
			} while (dong <= 0 || dong > MAX_DONG);

			// Kiểm tra số cột nhập vào có hợp lệ?
			do {
				cout << "Nhap so cot cua ma tran (1..15): ";
				cin >> cot;

				if (cot <= 0 || cot > MAX_COT)
					cout << "So cot khong hop le, vui long nhap lai (1..15)" << endl;
			} while (cot <= 0 || cot > MAX_COT);

			nhapMang(a, dong, cot);
			daNhapMang = true;
			cout << endl;
			break;

		case 2: // Xuất mảng (sau khi đã nhập mảng)
			if (daNhapMang == true)
			{
				xuatMang(a, dong, cot);
				cout << endl;
			}
			else
			{
```

```
            cout << "Ban chua nhap mang!\n";
            cout << "vui long chon 1 de nhap truoc khi xu ly tiep!\n";
        }
        break;

    case 3: // Tính tổng (sau khi đã nhập mảng)
        if (daNhapMang == true)
        {
            xuatMang(a, dong, cot);
            cout << endl;

            cout << "Tong cua cac gia tri trong mang la: "
                 << tinhTong(a, dong, cot) << endl;
            cout << endl;
        }
        else
        {
            cout << "Ban chua nhap mang!\n";
            cout << "vui long chon 1 de nhap truoc khi xu ly tiep!\n";
        }
        break;

    case 4: // Tìm giá trị nhỏ nhất (sau khi đã nhập mảng)
        if (daNhapMang == true)
        {
            xuatMang(a, dong, cot);
            cout << endl;

            int viTriDong, viTriCot;
            timGiaTriNhoNhat(a, dong, cot, viTriDong, viTriCot);
            cout << "Gia tri nho nhat nam tai dong [" << viTriDong + 1
                 << "] va cot [" << viTriCot + 1 << "]\n";
            cout << endl;
        }
        else
        {
            cout << "Ban chua nhap mang!\n";
            cout << "vui long chon 1 de nhap truoc khi xu ly tiep!\n";
        }
        break;

    case 5: // Tìm giá trị lớn nhất (sau khi đã nhập mảng)
        if (daNhapMang == true)
        {
            xuatMang(a, dong, cot);
            cout << endl;

            int viTriDong, viTriCot;
            timGiaTriLonNhat(a, dong, cot, viTriDong, viTriCot);
            cout << "Gia tri lon nhat nam tai dong [" << viTriDong + 1
                 << "] va cot [" << viTriCot + 1 << "]\n";
            cout << endl;
        }
        else
        {
            cout << "Ban chua nhap mang!\n";
            cout << "vui long chon 1 de nhap truoc khi xu ly tiep!\n";
        }
        break;

    case 6: // Thoát chương trình
        cout << "Chao tam biet" << endl;
```

```
                return 0;

            default:
                if (luaChon >= 1 || luaChon <= 6)
                    cout << "Vui long nhap lua chon (1..6)!\n" << endl;
            } // switch-case

            _getch();

        } while (luaChon >= 1 || luaChon <= 6);

        return 0;
} // main()

// 01. Hàm nhập mảng
void nhapMang(int a[MAX_DONG][MAX_COT], int dong, int cot)
{
        // Nhập ma trận
        for (int i = 0; i < dong; i++)
        {
            cout << "Hay nhap cung luc " << cot
                << " phan tu cua dong thu [" << i + 1
                << "]: ";
            for (int j = 0; j < cot; j++)
                cin >> a[i][j];
        }
} // nhapMang()

// 02. Hàm xuất mảng
void xuatMang(int a[MAX_DONG][MAX_COT], int dong, int cot)
{
        cout << "Mang cua ban la:\n";
        for (int i = 0; i < dong; i++)
        {
            for (int j = 0; j < cot; j++)
                cout << a[i][j] << "\t";
            cout << endl;
        }
} // xuatMang()

// 03. Hàm tính tổng các giá trị được lưu trong mảng
int tinhTong(int a[MAX_DONG][MAX_COT], int dong, int cot)
{
        int tongGiaTri = 0;

        for (int i = 0; i < dong; i++)
            for (int j = 0; j < cot; j++)
                tongGiaTri = tongGiaTri + a[i][j];

        return tongGiaTri;
} // tinhTong()

// 04. Hàm tìm giá trị nhỏ nhất
void timGiaTriNhoNhat(int a[MAX_DONG][MAX_COT], int dong, int cot, int &viTriDong, int &viTriCot)
{
        // Giả sử vị trí a[0][0] là giá trị nhỏ nhất
        int min = a[0][0];
        viTriDong = 0, viTriCot = 0;

        for (int i = 0; i < dong; i++)
            for (int j = 0; j < cot; j++)
                if (a[i][j] < min)
```

```
                {
                        min = a[i][j];

                        viTriDong = i;
                        viTriCot = j;
                }

        cout << "Gia tri nho nhat la: " << min << endl;
} // giaTriNhoNhat()

// 05. Hàm tìm giá trị lớn nhất
void timGiaTriLonNhat(int a[MAX_DONG][MAX_COT], int dong, int cot, int &viTriDong, int &viTriCot)
{
        // Giả sử vị trí a[0][0] là giá trị lớn nhất
        int max = a[0][0];
        viTriDong = 0, viTriCot = 0;

        for (int i = 0; i < dong; i++)
                for (int j = 0; j < cot; j++)
                        if (a[i][j] > max)
                        {
                                max = a[i][j];

                                viTriDong = i;
                                viTriCot = j;
                        }

        cout << "Gia tri lon nhat la: " << max << endl;
} // giaTriLonNhat()




/*
        Bài 04: Viết 01 chương trình thực hiện toàn bộ các công việc sau:

                a. Cho phép người dùng nhập vào một mảng số nguyên
                   gồm r hàng và c cột (tối đa 5 hàng và 6 cột,
                   nếu nhập sai thì yêu cầu nhập lại cho đến khi đúng mới tiếp tục).

                b. Cho người dùng nhập vào một vị trí cột / hàng cần tính tổng.
                   Hãy tiến hành tính tổng các giá trị lưu trữ
                   trong cột / hàng mà người dùng yêu cầu.

                c. Cho người dùng nhập vào một giá trị x bất kỳ.
                   Hãy tìm xem x có tồn tại trong mảng hay không?
                   Nếu có thì tồn tại bao nhiêu lần? Tại các vị trí nào?
*/

#include <iostream>
#include <conio.h>
using namespace std;

const int MAX_DONG = 5, MAX_COT = 6;

void nhapMang(int a[MAX_DONG][MAX_COT], int dong, int cot);
void xuatMang(int a[MAX_DONG][MAX_COT], int dong, int cot);
int tinhTongTrongMotDong(int a[MAX_DONG][MAX_COT], int dong, int cot);
int tinhTongTrongMotCot(int a[MAX_DONG][MAX_COT], int dong, int cot);
void timGiaTriBatKi(int a[MAX_DONG][MAX_COT], int dong, int cot,
                                int giaTriCanTim);
```

```
int main()
{
	int a[MAX_DONG][MAX_COT], dong, cot;
	int luaChon;
	bool daNhapMang = false;

	do {
		system("cls");

		cout << endl;
		cout << "CHUONG TINH TINH TOAN TREN MANG HAI CHIEU\n"
			<< "1. Nhap mang\n"
			<< "2. Xuat mang (sau khi da nhap mang)\n"
			<< "3. Tinh tong cac gia tri trong mot cot (sau khi da nhap mang)\n"
			<< "4. Tinh tong cac gia tri trong mot hang (sau khi da nhap mang)\n"
			<< "5. Tim gia tri bat ki (sau khi da nhap mang)\n"
			<< "6. Thoat chuong trinh\n"
			<< endl
			<< "Lua chon cua ban (1..6): ";
		cin >> luaChon;
		cout << endl;

		switch (luaChon)
		{
		case 1: // Nhập mảng
			// Kiểm tra số dòng nhập vào có hợp lệ?
			do {
				cout << "Nhap so dong cua ma tran (1..5): ";
				cin >> dong;

				if (dong <= 0 || dong > MAX_DONG)
					cout << "So dong khong hop le, vui long nhap lai (1..5)" << endl;
			} while (dong <= 0 || dong > MAX_DONG);

			// Kiểm tra số cột nhập vào có hợp lệ?
			do {
				cout << "Nhap so cot cua ma tran (1..6): ";
				cin >> cot;

				if (cot <= 0 || cot > MAX_COT)
					cout << "So cot khong hop le, vui long nhap lai (1..6)" << endl;
			} while (cot <= 0 || cot > MAX_COT);

			nhapMang(a, dong, cot);
			daNhapMang = true;
			cout << endl;
			break;

		case 2: // Xuất mảng (sau khi đã nhập mảng)
			if (daNhapMang == true)
			{
				xuatMang(a, dong, cot);
				cout << endl;
			}
			else
			{
				cout << "Ban chua nhap mang!\n";
				cout << "vui long chon 1 de nhap truoc khi xu ly tiep!\n";
			}
			break;

		case 3: // Tính tổng dòng (sau khi đã nhập mảng)
```

```
                if (daNhapMang == true)
                {
                        int dongCanTinhTong;

                        xuatMang(a, dong, cot);
                        cout << endl;

                        // Kiểm tra số dòng nhập vào có hợp lệ?
                        do {
                                cout << "Nhap so dong can tinh tong (1.." << dong << "): ";
                                cin >> dongCanTinhTong;

                                if (dongCanTinhTong <= 0 || dongCanTinhTong > dong)
                                        cout << "So dong khong hop le, vui long nhap lai (1.." << dong
<< ")\n";
                        } while (dongCanTinhTong <= 0 || dongCanTinhTong > dong);

                        cout << "Tong gia tri tai dong " << dongCanTinhTong << " la: "
                                << tinhTongTrongMotDong(a, dongCanTinhTong, cot) << endl;
                }
                else
                {
                        cout << "Ban chua nhap mang!\n";
                        cout << "vui long chon 1 de nhap truoc khi xu ly tiep!\n";
                }
                break;

        case 4: // Tính tổng cột (sau khi đã nhập mảng)
                if (daNhapMang == true)
                {
                        int cotCanTinhTong;

                        xuatMang(a, dong, cot);
                        cout << endl;

                        // Kiểm tra số dòng nhập vào có hợp lệ?
                        do {
                                cout << "Nhap so cot can tinh tong (1.." << cot << "): ";
                                cin >> cotCanTinhTong;

                                if (cotCanTinhTong <= 0 || cotCanTinhTong > cot)
                                        cout << "So cot khong hop le, vui long nhap lai (1.." << cot <<
")\n";
                        } while (cotCanTinhTong <= 0 || cotCanTinhTong > cot);

                        cout << "Tong gia tri tai cot " << cotCanTinhTong << " la: "
                                << tinhTongTrongMotCot(a, dong, cotCanTinhTong) << endl;
                }
                else
                {
                        cout << "Ban chua nhap mang!\n";
                        cout << "vui long chon 1 de nhap truoc khi xu ly tiep!\n";
                }
                break;

        case 5: // Tìm giá trị bất kì (sau khi đã nhập mảng)
                if (daNhapMang == true)
                {
                        int giaTriCanTim;

                        xuatMang(a, dong, cot);
                        cout << endl;
```

```
                        cout << "Vui long nhap gia tri can tim: ";
                        cin >> giaTriCanTim;
                        cout << endl;

                        timGiaTriBatKi(a, dong, cot, giaTriCanTim);

                        // Kiểm tra giá trị cần tìm có hợp lệ?
                  }
                  else
                  {
                        cout << "Ban chua nhap mang!\n";
                        cout << "vui long chon 1 de nhap truoc khi xu ly tiep!\n";
                  }
                  break;

            case 6: // Thoát chương trình
                  cout << "Chao tam biet" << endl;
                  return 0;

            default:
                  if (luaChon >= 1 || luaChon <= 6)
                        cout << "Vui long nhap lua chon (1..6)!\n" << endl;
            } // switch-case

            _getch();

      } while (luaChon >= 1 || luaChon <= 6);

      return 0;
} // main()

// 01. Hàm nhập mảng
void nhapMang(int a[MAX_DONG][MAX_COT], int dong, int cot)
{
      // Nhập ma trận
      for (int i = 0; i < dong; i++)
      {
            cout << "Hay nhap cung luc " << cot
                  << " phan tu cua dong thu [" << i + 1
                  << "]: ";
            for (int j = 0; j < cot; j++)
                  cin >> a[i][j];
      }
} // nhapMang()

// 02. Hàm xuất mảng
void xuatMang(int a[MAX_DONG][MAX_COT], int dong, int cot)
{
      cout << "Mang cua ban la:\n";
      for (int i = 0; i < dong; i++)
      {
            for (int j = 0; j < cot; j++)
                  cout << a[i][j] << "\t";
            cout << endl;
      }
} // xuatMang()

// 03. Hàm tính tổng các giá trị lưu trữ trong một dòng
int tinhTongTrongMotDong(int a[MAX_DONG][MAX_COT], int dongCanTinhTong, int cot)
{
      int ketQuaTinhTongDong = 0;
```

```
	for (int j = 0; j < cot; j++)
			ketQuaTinhTongDong = ketQuaTinhTongDong + a[dongCanTinhTong - 1][j];

	return ketQuaTinhTongDong;
}

// 04. Hàm tính tổng các giá trị lưu trữ trong một cột
int tinhTongTrongMotCot(int a[MAX_DONG][MAX_COT], int dong, int cotCanTinhTong)
{
	int ketQuaTinhTongCot = 0;

	for (int i = 0; i < dong; i++)
			ketQuaTinhTongCot = ketQuaTinhTongCot + a[i][cotCanTinhTong - 1];

	return ketQuaTinhTongCot;
}

// 05. Hàm tìm giá trị bất kì, tính số lần xuất hiện và in ra các vị trí xuất hiện
void timGiaTriBatKi(int a[MAX_DONG][MAX_COT], int dong, int cot,
				int giaTriCanTim)
{
	int demSoLanXuatHien = 0;

	for (int i = 0; i < dong; i++)
		for (int j = 0; j < cot; j++)
			if (a[i][j] == giaTriCanTim)
			{
				demSoLanXuatHien++;

				if (demSoLanXuatHien == 1)
					cout << giaTriCanTim << " co xuat hien trong mang va xuat hien tai:
";

				cout << "[" << i + 1 << "][" << j + 1 << "]  ";
			}

	cout << endl;

	cout << giaTriCanTim << " xuat hien tong cong: " << demSoLanXuatHien << " lan\n" << endl;

	if (demSoLanXuatHien == 0)
		cout << giaTriCanTim << " khong co xuat hien trong mang\n";
}



/*
	Bài 05. Viết chương trình cho nhập vào một ma trận vuông cấp n (chứa các số nguyên).
			Sau đó tính tổng các giá trị trên đường chéo chính
			và đường chéo phụ của ma trận vừa nhập.

				Ví dụ: cho ma trận
                                                1     4     0
                                                8     15    3
                                                6     9     2

                                Thì tổng các giá trị trên đường chéo chính là = 1 + 15 + 2 =
18,
                                tổng các giá trị trên đường chéo phụ là = 6 + 15 + 0 = 21.
*/
```

```cpp
#include <iostream>
using namespace std;

const int MAX = 10;

void nhapMaTran(int a[][MAX], int soCapThucTe);
void xuatMaTran(int a[][MAX], int soCapThucTe);
int tinhTongDuongCheoChinh(int a[][MAX], int soCapThucTe);
int tinhTongDuongCheoPhu(int a[][MAX], int soCapThucTe);

int main()
{
	int a[MAX][MAX], soCapThucTe;

	do {
		cout << "Nhap cap cua ma tran (1..10): ";
		cin >> soCapThucTe;

		if (soCapThucTe < 1 || soCapThucTe > MAX)
			cout << "Nhap sai, vui long nhap lai!\n";
	} while (soCapThucTe < 1 || soCapThucTe > MAX);

	nhapMaTran(a, soCapThucTe);
	cout << endl;

	xuatMaTran(a, soCapThucTe);
	cout << endl;

	cout << "Tong duong cheo chinh: " << tinhTongDuongCheoChinh(a, soCapThucTe) << endl;

	cout << "Tong duong cheo phu: " << tinhTongDuongCheoPhu(a, soCapThucTe) << endl;

	return 0;
}


// 01. Hàm nhập ma trận
void nhapMaTran(int a[][MAX], int soCapThucTe)
{
	cout << "Ma tran se co " << soCapThucTe << " dong va " << soCapThucTe << " cot" << endl;
	for (int i = 0; i < soCapThucTe; i++)
	{
		cout << "Nhap mot loat " << soCapThucTe << " phan tu tren dong thu [" << i << "]: ";
		for (int j = 0; j < soCapThucTe; j++)
			cin >> a[i][j];
	}
}

// 02. Hàm xuất ma trận
void xuatMaTran(int a[][MAX], int soCapThucTe)
{
	cout << "Ma tran vua nhap la:\n";
	for (int i = 0; i < soCapThucTe; i++)
	{
		for (int j = 0; j < soCapThucTe; j++)
			cout << a[i][j] << "\t";

		cout << endl;
	}
}
```

```
// 03. Hàm tính tổng của đường chéo chính
int tinhTongDuongCheoChinh(int a[][MAX], int soCapThucTe)
{
	int tongDuongCheoChinh = 0;

	for (int i = 0; i < soCapThucTe; i++)
		for (int j = 0; j < soCapThucTe; j++)
			if (i == j)
				tongDuongCheoChinh = tongDuongCheoChinh + a[i][j];

	return tongDuongCheoChinh;
}

// 04. Hàm tính tổng của đường chéo phụ
int tinhTongDuongCheoPhu(int a[][MAX], int soCapThucTe)
{
	int tongDuongCheoPhu = 0;

	for (int i = 0; i < soCapThucTe; i++)
		for (int j = 0; j < soCapThucTe; j++)
			if (j == (soCapThucTe - i - 1))
				tongDuongCheoPhu = tongDuongCheoPhu + a[i][j];

	return tongDuongCheoPhu;
}




/*
	Bài 06: Viết chương trình cho nhập vào 2 ma trận (chứa các số nguyên).
			Sau đó tính kết quả tổng và tích hai ma trận đó.
*/

// Lỗi khi hai ma trận khác cấp
// Cần xem lại hàm tính tổng và tích ma trận

#include <iostream>
#include <ctime>
using namespace std;

const int MAX = 50;

void nhapMaTran(int arr[][MAX], int &dong, int &cot);
void xuatMaTran(int arr[][MAX], int dong, int cot);
void tinhTongMaTran(int maTranA[][MAX], int maTranB[][MAX], int maTranKetQua[][MAX],
				int dong, int cot);
void tinhTichMaTran(int maTranA[][MAX], int dongA, int cotA,
				int maTranB[][MAX], int dongB, int cotB,
				int maTranKetQua[][MAX]);

int main()
{
	// Kết hợp với hàm tạo giá trị ngẫu nhiên rand()
	// để giá trị của mảng khác nhau sau mỗi lần thực thi
	srand(time(0));

	int maTranA[MAX][MAX], maTranB[MAX][MAX], maTranKetQua[MAX][MAX];

	int dongA, cotA, dongB, cotB;

	// Nhập hai ma trận
```

```
	cout << "Ma tran a:\n";
	nhapMaTran(maTranA, dongA, cotA);
	cout << endl;

	cout << "Ma tran b:\n";
	nhapMaTran(maTranB, dongB, cotB);
	cout << endl;

	// Xuất hai ma trận
	cout << "Ma tran a:\n";
	xuatMaTran(maTranA, dongA, cotA);
	cout << endl;

	cout << "Ma tran b:\n";
	xuatMaTran(maTranB, dongB, cotB);
	cout << endl;

	// Tính tổng hai ma trận
	if (dongA == dongB && cotA == cotB)
	{
		tinhTongMaTran(maTranA, maTranB, maTranKetQua, dongA, cotA);

		cout << "Tong ma tran a + b:\n";
		xuatMaTran(maTranKetQua, dongA, cotA);
		cout << endl;
	}
	else
	{
		cout << "Hai ma tran nay khong the tinh tong\n";
		cout << endl;
	}

	// Tính tích hai ma trận
	if (cotA == dongB)
	{
		tinhTichMaTran(maTranA, dongA, cotA, maTranB, dongB, cotB, maTranKetQua);

		cout << "Tich ma tran a * b:\n";
		xuatMaTran(maTranKetQua, dongA, cotA);
		cout << endl;
	}
	else
	{
		cout << "Hai ma tran nay khong the tinh tong\n";
		cout << endl;
	}

	return 0;
}

// 01. Hàm nhập ma trận
void nhapMaTran(int arr[][MAX], int &dong, int &cot)
{
	do {
		cout << "Nhap so dong (1..50): ";
		cin >> dong;

		if (dong < 1 || dong > MAX)
			cout << "Nhap sai, vui long nhap lai!";
	} while (dong < 1 || dong > MAX);

	do {
```

```
		cout << "Nhap so cot (1..50): ";
		cin >> cot;

		if (cot < 1 || cot > MAX)
			cout << "Nhap sai, vui long nhap lai!";
	} while (cot < 1 || cot > MAX);

	// --- Legacy ---
	//cout << "Ma tran se co " << dong << " dong va " << cot << " cot" << endl;
	//for (int i = 0; i < dong; i++)
	//{
	//	cout << "Nhap mot loat " << cot << " phan tu tren dong thu [" << i << "]: ";
	//	for (int j = 0; j < cot; j++)
	//		cin >> arr[i][j];
	//}
	// --- Legacy ---

	for (int i = 0; i < dong; i++)
		for (int j = 0; j < cot; j++)
			arr[i][j] = rand() % 5;
}

// 02. Hàm xuất ma trận
void xuatMaTran(int arr[][MAX], int dong, int cot)
{
	for (int i = 0; i < dong; i++)
	{
		for (int j = 0; j < cot; j++)
			cout << arr[i][j] << "\t";

		cout << endl;
	}
}

// 03. Hàm tính tổng hai ma trận
void tinhTongMaTran(int maTranA[][MAX], int maTranB[][MAX], int maTranKetQua[][MAX],
				int dong, int cot)
{
	for (int i = 0; i < dong; i++)
		for (int j = 0; j < cot; j++)
			maTranKetQua[i][j] = maTranA[i][j] + maTranB[i][j];
}

// 04. Hàm tính tích hai ma trận
void tinhTichMaTran(int maTranA[][MAX], int dongA, int cotA,
				int maTranB[][MAX], int dongB, int cotB,
				int maTranKetQua[][MAX])
{
	for (int i = 0; i < dongA; i++)
		for (int j = 0; j < cotB; j++)
		{
			maTranKetQua[i][j] = 0;

			for (int k = 0; k < cotA; k++)
				maTranKetQua[i][j] += maTranA[i][k] * maTranB[k][j];
		}
}



/*
```

```
	Bài 07. Viết 01 chương trình thực hiện các yêu cầu sau:

			a. Xây dựng các hàm:
				- Nhập vào 1 mảng số nguyên gồm r hàng và c cột;
				- Xuất giá trị 1 mảng số nguyên gồm r hàng và c cột.

			b. Xây dựng hàm trả về giá trị trung bình cộng
			   của các phần tử trong 1 mảng số nguyên gồm r hàng và c cột

			c. Xây dựng hàm nhận vào 1 mảng số nguyên gồm r hàng và c cột;
			   giá trị số nguyên x.
			   Kiểm tra xem x có tồn tại trong mảng hay không
			   và trả về kết quả tương ứng.

			d. Xây dựng hàm nhận vào 1 mảng số nguyên gồm r hàng và c cột
			   (giả sử các giá trị nhập không trùng nhau) và 2 giá trị số nguyên x và y
			   (x và y phải đều tồn tại trong mảng).
			   Tiến hành hoán đổi 2 giá trị x và y này.
			   Lưu ý: sử dụng lại hàm đã xây dựng ở câu c.

			e. Xây dựng hàm nhận vào 1 mảng số nguyên gồm r hàng và c cột;
			   vị trí cột cần tính tích.
			   Sau đó trả về tích các giá trị số trong cột đã nhận vào.

			f. Viết hàm main để kiểm tra các hàm đã xây dựng.
*/

#include <iostream>
#include <ctime>
using namespace std;

const int MAX = 10;

void nhapMang(int a[][MAX], int dong, int cot);
void xuatMang(int a[][MAX], int dong, int cot);
double tinhTrungBinhCong(int a[][MAX], int dong, int cot);
bool kiemTraCoTonTai(int a[][MAX], int dong, int cot, int soCanTim);
void hoanDoiGiaTri(int a[][MAX], int dong, int cot, int &x, int &y);
int tinhTichCot(int a[][MAX], int dong, int cot, int cotTinhTich);

int main()
{
	srand(time(0));

	int a[MAX][MAX], dong, cot;
	int x, y;
	int cotTinhTich;
	//int soCanTim;

	do {
		cout << "Nhap so dong (1..50): ";
		cin >> dong;

		if (dong <= 0 || dong > MAX)
			cout << "Nhap sai, vui long nhap lai";
	} while (dong <= 0 || dong > MAX);

	do {
		cout << "Nhap so cot (1..50): ";
		cin >> cot;

		if (cot <= 0 || cot > MAX)
```

```
            cout << "Nhap sai, vui long nhap lai";
    } while (cot <= 0 || cot > MAX);

    // Nhập mảng
    nhapMang(a, dong, cot);
    cout << endl;

    // Xuất mảng
    cout << "Ma tran la:\n";
    xuatMang(a, dong, cot);
    cout << endl;

    // Tính trung bình cộng
    cout << "Trung binh cong cua cac phan tu trong mang la: "
         << tinhTrungBinhCong(a, dong, cot) << endl;
    cout << endl;

    // Kiểm tra tồn tại
    //cout << "Nhap so can tim trong mang: ";
    //cin >> soCanTim;
    //cout << endl;

    //if (kiemTraCoTonTai(a, dong, cot, soCanTim) == true)
    //    cout << "So [" << soCanTim << "] co ton tai trong mang!\n";
    //else
    //    cout << "So [" << soCanTim << "] khong ton tai trong mang!\n";

    cout << endl;

    // Nhập các giá trị để hoán đổi
    cout << "Nhap 2 gia tri x va y de hoan doi\n";
    cout << "(Hoan doi 2 gia tri hien co trong ma tran)\n";
    cout << endl;

    // Hoán đổi x và y sau khi đã hợp lệ
    hoanDoiGiaTri(a, dong, cot, x, y);
    cout << endl;

    // Xuất ma trận sau khi đã hoán đổi
    cout << "Ma tran sau khi hoan doi " << x << " va " << y << " la:\n";
    xuatMang(a, dong, cot);
    cout << endl;

    // Tính tích của cột tuỳ chọn
    do {
        cout << "Nhap cot can tinh tich (1.." << cot << "): ";
        cin >> cotTinhTich;
        cout << endl;

        if (cotTinhTich <= 0 || cotTinhTich > cot)
            cout << "Vui long nhap cot tinh tich (1.." << cot << ")\n" << endl;
    } while (cotTinhTich <= 0 || cotTinhTich > cot);

    cout << "Tich cua cot " << cotTinhTich << " la: "
         << tinhTichCot(a, dong, cot, cotTinhTich) << endl;
    cout << endl;

    return 0;
}

// 01. Hàm nhập mảng
void nhapMang(int arr[][MAX], int dong, int cot)
```

```
{
	for (int i = 0; i < dong; i++)
		for (int j = 0; j < cot; j++)
			arr[i][j] = rand() % 5;
}

// 02. Hàm xuất mảng
void xuatMang(int arr[][MAX], int dong, int cot)
{
	for (int i = 0; i < dong; i++)
	{
		for (int j = 0; j < cot; j++)
			cout << arr[i][j] << "\t";

		cout << endl;
	}
}

// 03. Hàm tính trung bình cộng
double tinhTrungBinhCong(int a[][MAX], int dong, int cot)
{
	double ketQua;
	int tong = 0;

	for (int i = 0; i < dong; i++)
		for (int j = 0; j < cot; j++)
			tong = tong + a[i][j];

	ketQua = (double)tong / (dong * cot);

	return ketQua;
}

// 04. Hàm kiểm tra một số có tồn tại trong mảng
bool kiemTraCoTonTai(int a[][MAX], int dong, int cot, int soCanTim)
{
	bool ketQua = false;

	for (int i = 0; i < dong; i++)
		for (int j = 0; j < cot; j++)
			if (a[i][j] == soCanTim)
				ketQua = true;

	return ketQua;
}

// 05. Hàm hoán đổi vị trí của hai số
void hoanDoiGiaTri(int a[][MAX], int dong, int cot, int &x, int &y)
{
	// Kiểm tra x có tồn tại
	do {
		cout << "Nhap x = ";
		cin >> x;

		if (kiemTraCoTonTai(a, dong, cot, x) == false)
		{
			cout << "So [" << x << "] khong ton tai trong mang!\n";
			cout << endl;
		}
	} while (kiemTraCoTonTai(a, dong, cot, x) == false);

	// Kiểm tra y có tồn tại
```

```
	do {
		cout << "Nhap y = ";
		cin >> y;

		if (kiemTraCoTonTai(a, dong, cot, y) == false)
		{
			cout << "So [" << y << "] khong ton tai trong mang!\n";
			cout << endl;
		}
	} while (kiemTraCoTonTai(a, dong, cot, y) == false);

	// Nếu nhập x và y hợp lệ
	for (int i = 0; i < dong; i++)
		for (int j = 0; j < cot; j++)
			if (a[i][j] == x)
				a[i][j] = y;
			else
				if (a[i][j] == y)
					a[i][j] = x;
}

// 06. Hàm tính tích của một cột tuỳ
int tinhTichCot(int a[][MAX], int dong, int cot, int cotTinhTich)
{
	int ketQua = 1;

	for (int i = 0; i < dong; i++)
		ketQua = ketQua * a[i][cotTinhTich - 1];

	return ketQua;
}




/*
	Bài 08. Xây dựng các hàm:

			- Khởi tạo ngẫu nhiên giá trị cho 1 mảng số nguyên
			  gồm n hàng và n cột (ma trận vuông cấp n);

			- Xuất giá trị lưu trữ trong 1 mảng số nguyên gồm n hàng và n cột.

			- Viết hàm tính tổng, tích 2 ma trận vuông cấp n.

			- Viết hàm main để kiểm tra các hàm đã xây dựng.
*/

#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;

const int MAX = 50;

void khoiTaoNgauNhienGiaTri(int a[][MAX], int &n);
void xuatMang(int a[][MAX], int n);
int tinhTong(int a[][MAX], int n);
long tinhTich(int a[][MAX], int n);

int main()
{
```

```
	int a[MAX][MAX], b[MAX][MAX], n;

	cout << "Mang a: \n";
	khoiTaoNgauNhienGiaTri(a, n);
	cout << endl;

	cout << "Mang b: \n";
	khoiTaoNgauNhienGiaTri(b, n);
	cout << endl;

	cout << "Mang a: \n";
	xuatMang(a, n);
	cout << endl;

	cout << "Mang b: \n";
	xuatMang(b, n);
	cout << endl;

	cout << "Tong 2 ma tran a va b: " << tinhTong(a, n) + tinhTong(b, n) << endl;

	cout << "Tich 2 ma tran a va b: " << tinhTich(a, n) * tinhTich(b, n) << endl;

	return 0;
}

void khoiTaoNgauNhienGiaTri(int a[][MAX], int &n)
{
	do {
		cout << "Nhap so cap cua ma tran (1..50): ";
		cin >> n;

		if (n < 1 || n > MAX)
			cout << "Nhap sai, vui long nhap lai trong khoang 1..50\n" << endl;
	} while (n < 1 || n > MAX);

	for (int i = 0; i < n; i++)
		for (int j = 0; j < n; j++)
			a[i][j] = rand() % 5 + 1;
}

void xuatMang(int a[][MAX], int n)
{
	for (int i = 0; i < n; i++)
	{
		for (int j = 0; j < n; j++)
			cout << a[i][j] << "\t";
		cout << endl;
	}
}

int tinhTong(int a[][MAX], int n)
{
	int tongGiaTri = 0;

	for (int i = 0; i < n; i++)
		for (int j = 0; j < n; j++)
			tongGiaTri = tongGiaTri + a[i][j];

	return tongGiaTri;
}

long tinhTich(int a[][MAX], int n)
```

```
{
	long tichGiaTri = 1;

	for (int i = 0; i < n; i++)
		for (int j = 0; j < n; j++)
			tichGiaTri = tichGiaTri * a[i][j];

	return tichGiaTri;
}




/*
	Bài 09. Viết chương trình gán các giá trị số nguyên từ 1 đến n^2
			cho các phần tử của ma trận vuông có kích thước n x n
			(2 <= n <= 10) được nhập từ bàn phím theo dạng zigzag cột.

			Ví dụ: n = 4

				Cột nhỏ nhất	Cột lớn nhất
					|		|
					v		V

		Dòng nhỏ nhất ->	1	8	9	16
					2	7	10	15
					3	6	11	14
		Dòng lớn nhất ->	4	5	12	13
*/

#include <iostream>
using namespace std;

const int SO_CAP_TOI_DA_CUA_MA_TRAN = 10;

void nhapMaTranZigzag(int a[][SO_CAP_TOI_DA_CUA_MA_TRAN], int soCapThucTe);
void xuatMaTran(int a[][SO_CAP_TOI_DA_CUA_MA_TRAN], int soCapThucTe);

int main()
{
	// Vì là ma trận vuông
	// nên số dòng và cột là bằng nhau
	// (sử dụng chung một giá trị)
	int a[SO_CAP_TOI_DA_CUA_MA_TRAN][SO_CAP_TOI_DA_CUA_MA_TRAN];

	// Số cấp thực tế của ma trận
	int soCapThucTe;

	do {
		cout << "Vui long nhap so cap cua ma tran (2 <= n <= 10): ";
		cin >> soCapThucTe;

		if (soCapThucTe < 2 || soCapThucTe > SO_CAP_TOI_DA_CUA_MA_TRAN)
		{
			cout << "Gia tri nhap khong hop le!\n";
			cout << "Vui long nhap: 0 < So cap <= 10\n";
			cout << endl;
		}
	} while (soCapThucTe < 2 || soCapThucTe > SO_CAP_TOI_DA_CUA_MA_TRAN);

	nhapMaTranZigzag(a, soCapThucTe);
```

```
	cout << endl;

	xuatMaTran(a, soCapThucTe);
	cout << endl;

	return 0;
}

// Hàm in ma trận Zigzag (cột)
void nhapMaTranZigzag(int a[][SO_CAP_TOI_DA_CUA_MA_TRAN], int soCapThucTe)
{
	// Xác định có bao nhiêu hướng
	//    - Cột chẵn: số tăng dần từ dòng nhỏ nhất -> lớn nhất
	//    - Cột lẻ: số tăng dần từ dòng lớn nhất -> nhỏ nhất
	//          -> Tuỳ vào cột mà giá trị tương ứng sẽ được gán


	// Biến để tăng dần và gán giá trị vào ma trận
	int giaTri = 1;

	// Vòng lặp đầu tiên sẽ là cột - đối tượng phụ thuộc
	for (int j = 0; j < soCapThucTe; j++)
		// Cột chẵn
		if (j % 2 == 0)
			// Gán giá trị từ dòng nhỏ nhất -> lớn nhất
			for (int i = 0; i < soCapThucTe; i++)
				a[i][j] = giaTri++;
		else // Cột lẻ
			// Gán giá trị từ dòng lớn nhất -> nhỏ nhất
			for (int i = soCapThucTe - 1; i >= 0; i--)
				a[i][j] = giaTri++;
}

void xuatMaTran(int a[][SO_CAP_TOI_DA_CUA_MA_TRAN], int soCapThucTe)
{
	// In ma trận ra
	cout << "Mang dang luu tru:\n";

	// Duyệt từng dòng
	for (int i = 0; i < soCapThucTe; i++)
	{
		// Duyệt từng cột (của từng dòng)
		for (int j = 0; j < soCapThucTe; j++)
			// In giá trị vừa mới duyệt ra
			cout << a[i][j] << "\t";

		// Sau khi in ra hết số cột trên một dòng,
		// mới tiến hành xuống hàng để in dòng mới
		cout << endl;
	}
}




/*
	Bài 10: Viết chương trình gán các giá trị số nguyên từ 1 đến n2
			cho các phần tử của ma trận vuông có kích thước n x n
			(2 <= n <= 10) được nhập từ bàn phím theo dạng xoắn ốc.

			Ví dụ: n = 4
```

```
                              Cột nhỏ nhất Cột lớn nhất
                                          |           |
                                          v           V

            Dòng nhỏ nhất ->  1     2     3     4
                                    12    13    14    5
                                    11    16    15    6
            Dòng lớn nhất ->  10    9     8     7
*/

#include <iostream>
using namespace std;

const int SO_CAP_TOI_DA_CUA_MA_TRAN = 10;

void nhapMaTranXoanOc(int a[][SO_CAP_TOI_DA_CUA_MA_TRAN], int soCapThucTe);
void xuatMaTran(int a[][SO_CAP_TOI_DA_CUA_MA_TRAN], int soCapThucTe);

int main()
{
	// Vì là ma trận vuông
	// nên số dòng và cột là bằng nhau
	// (sử dụng chung một giá trị)
	int a[SO_CAP_TOI_DA_CUA_MA_TRAN][SO_CAP_TOI_DA_CUA_MA_TRAN];

	// Số cấp thực tế của ma trận
	int soCapThucTe;

	do {
		cout << "Vui long nhap so cap cua ma tran (2 <= n <= 10): ";
		cin >> soCapThucTe;

		if (soCapThucTe < 2 || soCapThucTe > SO_CAP_TOI_DA_CUA_MA_TRAN)
		{
			cout << "Gia tri nhap khong hop le!\n";
			cout << "Vui long nhap: 0 < So cap <= 10\n";
			cout << endl;
		}
	} while (soCapThucTe < 2 || soCapThucTe > SO_CAP_TOI_DA_CUA_MA_TRAN);

	nhapMaTranXoanOc(a, soCapThucTe);
	cout << endl;

	xuatMaTran(a, soCapThucTe);
	cout << endl;

	return 0;
}


// 01. Hàm nhập ma trận kiểu xoắn ốc

/*
                              Cột nhỏ nhất Cột lớn nhất
                                          |           |
                                          v           V

            Dòng nhỏ nhất ->  1     2     3     4
                                    12    13    14    5
                                    11    16    15    6
            Dòng lớn nhất ->  10    9     8     7
*/
```

```
void nhapMaTranXoanOc(int a[][SO_CAP_TOI_DA_CUA_MA_TRAN], int soCapThucTe)
{
	int dongNhoNhat, dongLonNhat, cotNhoNhat, cotLonNhat;

	// Vi ma trận sẽ bắt đầu từ 0 nên:
	dongNhoNhat = cotNhoNhat = 0;
	dongLonNhat = cotLonNhat = soCapThucTe - 1;

	// Biến để tăng dần và gán giá trị vào ma trận
	int giaTriGan = 1;

	// Thực hiện việc xoán ốc nhiều lần,
	// cho đến khi nhận được giá trị cuối cùng
	//		Ví dụ: ma trận cấp 5 -> chạy giá trị đến 25 -> dừng
	while (giaTriGan <= soCapThucTe * soCapThucTe)
	{
		// Xác định có bao nhiêu hướng

		// Hướng 1:
		// dongNhoNhat cố định,
		// chạy giá trị từ cotNhoNhat -> cotLonNhat
		// Vì đây là dongNhoNhat, nên sau khi đã chạy giá trị,
		// để tránh bị gán đè thì phải tăng lên 1
		//			1	2	3	4
		for (int j = cotNhoNhat; j <= cotLonNhat; j++)
			a[dongNhoNhat][j] = giaTriGan++;
		dongNhoNhat++;

		// Hướng 2:
		// cotLonNhat cố định,
		// chạy giá trị từ dongNhoNhat -> dongLonNhat
		// Vì đây là cotLonNhat, nên sau khi đã chạy giá trị,
		// để tránh bị gán đè thì phải giảm xuống 1
		//					5
		//					6
		//					7
		for (int i = dongNhoNhat; i <= dongLonNhat; i++)
			a[i][cotLonNhat] = giaTriGan++;
		cotLonNhat--;

		// Hướng 3:
		// dongLonNhat cố định,
		// chạy giá trị từ cotLonNhat -> cotNhoNhat
		// Vì đây là dongLonNhat, nên sau khi đã chạy giá trị,
		// để tránh bị gán đè thì phải giảm xuống 1
		//			10	9	8
		for (int j = cotLonNhat; j >= cotNhoNhat; j--)
			a[dongLonNhat][j] = giaTriGan++;
		dongLonNhat--;

		// Hướng 4:
		// cotNhoNhat cố định,
		// chạy giá trị từ dongLonNhat -> dongNhoNhat
		// Vì đây là cotNhoNhat, nên sau khi đã chạy giá trị,
		// để tránh bị gán đè thì phải tăng lên 1
		//			12
		//			11
		for (int i = dongLonNhat; i >= dongNhoNhat; i--)
			a[i][cotNhoNhat] = giaTriGan++;
		cotNhoNhat++;
	}
```

```
}

// 02. Hàm xuất ma trận
void xuatMaTran(int a[][SO_CAP_TOI_DA_CUA_MA_TRAN], int soCapThucTe)
{
	// In ma trận ra
	cout << "Mang dang luu tru:\n";

	// Duyệt từng dòng
	for (int i = 0; i < soCapThucTe; i++)
	{
		// Duyệt từng cột (của từng dòng)
		for (int j = 0; j < soCapThucTe; j++)
			// In giá trị vừa mới duyệt ra
			cout << a[i][j] << "\t";

		// Sau khi in ra hết số cột trên một dòng,
		// mới tiến hành xuống hàng để in dòng mới
		cout << endl;
	}
}




/*
	Bài 11. Viết chương trình đặt chỗ ngồi cho 1 máy bay.
			Máy bay có 13 hàng ghế, mỗi hàng có 6 ghế.
			Hàng 1 và 2 là hạng thương gia,
			hàng 3 đến 7 là hạng phổ thông,
			hàng 8 trở đi là hạng tiết kiệm
*/

// Chưa hoàn thiện về các lựa chọn: luaChonHangVe, luaChonDeTiepTucDatVe

#include <iostream>
#include <conio.h>

using namespace std;

const int MAX = 20;

void xuatViTriChoNgoi(int a[][MAX], int hang, int cot);
bool kiemTraChoNgoiCoTrong(int a[][MAX], int viTriHang, int viTriCot);
void khoiTaoViTriDatChoMacDinh(int a[][MAX], int hang, int cot);
int chuyenKiTuThuongThanhHoa(int a[][MAX], char &viTriGheNgoi);

int main()
{
	int a[MAX][MAX] = { {0} };
	int hang = 13, cot = 6;
	int viTriHangDaChon, viTriCotDaChon;
	int luaChonHangVe, luaChonDeTiepTucDatVe;

	do {
		do {
			system("cls");
			cout << endl;

			khoiTaoViTriDatChoMacDinh(a, hang, cot);
			xuatViTriChoNgoi(a, hang, cot);
			cout << endl;
```

```
		cout << "Danh sach cac loai ve:\n";
		cout << "1. Hang thuong gia - hang ghe (1..2)\n";
		cout << "2. Hang pho thong - hang ghe (3..7)\n";
		cout << "3. Hang tiet kiem - hang ghe (8..13)\n";
		cout << "Vui long chon hang ve (1..3): ";
		cin >> luaChonHangVe;

		cout << endl;

		switch (luaChonHangVe)
		{
		case 1:
			do {
				cout << "Hang thuong gia - hang ghe (1..2)\n";
				cout << "Moi ban nhap hang ghe (1..2): ";
				cin >> viTriHangDaChon;
				cout << endl;

				if (viTriHangDaChon != 1 && viTriHangDaChon != 2)
					cout << "Nhap sai, vui long nhap lai - hang ghe (1..2)!\n" <<
endl;
			} while (viTriHangDaChon != 1 && viTriHangDaChon != 2);
			break;

		case 2:
			do {
				cout << "Hang pho thong - hang ghe (3..7)\n";
				cout << "Moi ban nhap hang ghe (3..7): ";
				cin >> viTriHangDaChon;
				cout << endl;

				if (viTriHangDaChon < 3 || viTriHangDaChon > 7)
					cout << "Nhap sai, vui long nhap lai - hang ghe (3..7)!\n" <<
endl;
			} while (viTriHangDaChon < 3 || viTriHangDaChon > 7);
			break;

		case 3:
			do {
				cout << "Hang tiet kiem - hang ghe (8..13)\n";
				cout << "Moi ban nhap hang ghe (8..13): ";
				cin >> viTriHangDaChon;
				cout << endl;

				if (viTriHangDaChon < 8 && viTriHangDaChon > 13)
					cout << "Nhap sai, vui long nhap lai - hang ghe (8..13)!\n" <<
endl;
			} while (viTriHangDaChon < 8 && viTriHangDaChon > 13);
			break;

			_getch();

		default:
			cout << "Nhap sai, vui long nhap lai (1..3)!\n";
		}

	} while (luaChonHangVe < 1 || luaChonHangVe > 3);

	char viTriGheNgoi;
	do {
		cout << "Moi nhap vi tri ghe (A..F): ";
```

```
			cin >> viTriGheNgoi;

			chuyenKiTuThuongThanhHoa(a, viTriGheNgoi);

			if (viTriGheNgoi < 'A' || viTriGheNgoi > char(65 + (cot - 1)))
				cout << "Nhap sai, vui long nhap lai (A..F)\n";
		} while (viTriGheNgoi < 'A' || viTriGheNgoi > char(65 + (cot - 1)));

		viTriCotDaChon = int(viTriGheNgoi) - 65;

		if (!kiemTraChoNgoiCoTrong(a, viTriHangDaChon - 1, viTriCotDaChon))
		{
			a[viTriHangDaChon - 1][viTriCotDaChon] = 1;
			system("cls");

			cout << "Xin chuc mung! Ban da dat cho thanh cong!\n" << endl;

			cout << "[Hang ghe cua ban la hang " << viTriHangDaChon
				<< " va ghe " << viTriGheNgoi << "]\n" << endl;

			cout << "Bang dat cho hien tai la: \n";
			xuatViTriChoNgoi(a, hang, cot);
		}
		else
			cout << "Rat tiec, vi tri ma ban vua dat hien khong con trong!\n";

		cout << "\nBan co muon tiep tuc dat ve?\n";
		cout << "1. Tiep tuc\n";
		cout << "2. Khong tiep tuc\n";
		cout << "Lua chon: ";
		cin >> luaChonDeTiepTucDatVe;

	} while (luaChonDeTiepTucDatVe == 1);
	cout << "\nHen gap lai, chuc mot ngay tot lanh\n" << endl;

	return 0;
}

void khoiTaoViTriDatChoMacDinh(int a[][MAX], int hang, int cot)
{
	a[1][0] = 1;
	a[5][3] = 1;
	a[9][5] = 1;
}

void xuatViTriChoNgoi(int a[][MAX], int hang, int cot)
{
	cout << "\t";
	for (int i = 0; i < cot; i++)
		cout << char(65 + i) << "\t";

	cout << endl;

	for (int i = 0; i < hang; i++)
	{
		cout << "Hang " << i + 1 << "\t";

		for (int j = 0; j < cot; j++)
			if (kiemTraChoNgoiCoTrong(a, i, j))
				cout << "x" << "\t";
			else
				cout << "*" << "\t";
```

```
			cout << endl;
		}
}

bool kiemTraChoNgoiCoTrong(int a[][MAX], int viTriHang, int viTriCot)
{
		if (a[viTriHang][viTriCot] == 1)
			return true;
		return false;
}

int chuyenKiTuThuongThanhHoa(int a[][MAX], char &viTriGheNgoi)
{
		if (viTriGheNgoi >= 'a' && viTriGheNgoi <= 'z')
			viTriGheNgoi = int(viTriGheNgoi - 32);

		return viTriGheNgoi;
}
```