TRƯỜNG ĐẠI HỌC SÀI GÒN

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO MÔN HỌC

CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT

THỰC HÀNH:

CON TRỎ, CẤU TRÚC, CHUỖI VÀ TẬP TIN

SINH VIÊN THỰC HIỆN: Nguyễn Đỗ Huy – 3121411085

LỚP: DCT124C7

GVHD: ĐỖ NHƯ TÀI

Thành phố Hồ Chí Minh , Tháng 2 Năm 2025

MỤC LỤC

Con trỏ cho đối tượng và mảng 1 chiều 2

1. Con trỏ cho đối tượng 2

Bài tập 1. 2

2. Con trỏ cho mảng một chiều 3

Bài tập 2. 3

Bài tập 3. 6

Bài tập 4. 8

Bài tập 5. 11

Cấu trúc 15

Bài tập 1. 15

Bài tập 2. 19

Bài tập 3. 24

Bài tập 4. 30

Xử lý chuỗi 35

Bài tập 1. 35

Bài tập 2. 36

Bài tập 3. 38

File văn bản 41

Bài tập 1. 41

Bài tập 2. 43

Bài tập 3. 45

Bài tập 4. 48

Con trỏ cho đối tượng và mảng 1 chiều

1. Con trỏ cho đối tượng

Bài tập 1.
Cho hệ số a, b, c của phương trình bậc hai. Viết chương trình giải phương trình bậc hai (dùng con trỏ).

Moi ban nhap he so a, b, c: 1 2 1

Phuong trinh co 1 nghiem kep x = -1.

- Ý tưởng: Chương trình giải phương trình bậc hai ax^2 + bx + c = 0 bằng cách sử dụng con trỏ.
  - Nhập các hệ số a, b, c.
  - Tính delta = b^2 - 4ac để kiểm tra loại nghiệm (2 nghiệm, 1 nghiệm kép, hoặc vô nghiệm).
  - Sử dụng con trỏ để lưu và trả về các nghiệm của phương trình.
- Tập các testcase kiểm tra:

Testcase	Input (a, b, c)	Kết quả mong đợi
1	0, 0, 0	Day khong phai la phuong trinh bac hai.
2	1, 2, 10	Phuong trinh vo nghiem.
3	1, 2, 1	Phuong trinh co 1 nghiem kep x = -1.
4	1, -5, 6	Phuong trinh co 2 nghiem phan biet: x1 = 3, x2 = 2

- Mã nguồn:

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

void Ptbac2(double a, double b, double c, double *x1, double *x2, int *n) {
    double delta = b * b - 4 * a * c;

    if (delta > 0) {
        *n = 2;
        *x1 = (-b + sqrt(delta)) / (2 * a);
        *x2 = (-b - sqrt(delta)) / (2 * a);
    } else if (delta == 0) {
        *n = 1;
        *x1 = -b / (2 * a);
        *x2 = *x1; // Nghiệm kép, x1 và x2 giống nhau
    } else {
```

```
        *n = 0; // Phương trình vô nghiệm
    }
}

int main() {
    double a, b, c;
    cout << "Moi ban nhap he so a, b, c: ";
    cin >> a >> b >> c;

    if (a == 0) {
        cout << "Day khong phai la phuong trinh bac hai!" << endl;
        return 0;
    }

    double x1, x2;
    int n;

    Ptbac2(a, b, c, &x1, &x2, &n);

    if (n == 2) {
        cout << "Phuong trinh co 2 nghiem phan biet: x1 = " << x1 << ", x2 = " <<
x2 << endl;
    } else if (n == 1) {
        cout << "Phuong trinh co 1 nghiem kep x = " << x1 << endl;
    } else {
        cout << "Phuong trinh vo nghiem." << endl;
    }

    return 0;
}
```

2. Con trỏ cho mảng một chiều

Bài tập 2.
Cho dãy số nguyên a có n phần tử ($n \leq 100$). Hãy kiểm tra các tính chất sau của dãy a:

- Tính chẵn lẻ (các phần tử xen kẽ nhau chẵn lẻ),
- Tính toàn chẵn (dãy số chứa toàn số chẵn),

Moi ban nhap so luong phan tu: 5

Phan tu 0: 2

Phan tu 1: 5

Phan tu 2: 6

Phan tu 3: 5

Phan tu 4: 2

+ Day co tinh chat chan le

+ Day khong co tinh chat toan chan

- Ý tưởng:
    - Sử dụng mảng một chiều và con trỏ để kiểm tra các tính chất của dãy số nguyên.
    - Nhập vào số lượng phần tử và giá trị của từng phần tử.
    - Kiểm tra:
        - Tính chẵn lẻ: Duyệt qua các phần tử và kiểm tra nếu các phần tử xen kẽ giữa số chẵn và lẻ.
        - Tính toàn chẵn: Duyệt qua mảng để xác định nếu tất cả các phần tử đều là số chẵn.
- Tập các testcase kiểm tra:

Testcase	Input	Kết quả mong đợi
1	Moi ban nhap so luong phan tu: 5 Phan tu 0: 2 Phan tu 1: 5 Phan tu 2: 6 Phan tu 3: 5 Phan tu 4: 2	+ Day co tinh chat chan le + Day khong co tinh chat toan chan
2	Moi ban nhap so luong phan tu: 4 Phan tu 0: 2 Phan tu 1: 4 Phan tu 2: 6 Phan tu 3: 8	+ Day khong co tinh chat chan le + Day co tinh chat toan chan
3	Moi ban nhap so luong phan tu: 5 Phan tu 0: 3 Phan tu 1: 3 Phan tu 2: 3 Phan tu 3: 3 Phan tu 4: 3	+ Day khong co tinh chat chan le + Day khong co tinh chat toan chan
4	Moi ban nhap so luong phan tu: 101	So luong phan tu khong hop le!

- Mã nguồn

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

bool TinhChanLe(int *arr, int n) {
    for (int i = 1; i < n; i++) {
        if ((arr[i] % 2) == (arr[i - 1] % 2)) {
            return false; // Hai phần tử liên tiếp không xen kẽ chẵn lẻ
        }
    }
    return true;
}

bool TinhToanChan(int *arr, int n) {
    for (int i = 0; i < n; i++) {
        if (arr[i] % 2 != 0) {
            return false; // Tìm thấy một phần tử lẻ
        }
    }
    return true;
}

int main() {
    int n;
    cout << "Moi ban nhap so luong phan tu: ";
    cin >> n;

    if (n <= 0 || n > 100) {
        cout << "So luong phan tu khong hop le!" << endl;
        return 0;
    }

    int arr[100];
    for (int i = 0; i < n; i++) {
        cout << "Phan tu " << i << ": ";
        cin >> *(arr + i); // Sử dụng con trỏ để nhập giá trị
    }

    bool isAlternating = TinhChanLe(arr, n);
    bool isAllEven = TinhToanChan(arr, n);

    if (isAlternating) {
        cout << "+ Day co tinh chat chan le" << endl;
    } else {
        cout << "+ Day khong co tinh chat chan le" << endl;
    }

    if (isAllEven) {
```

```
        cout << "+ Day co tinh chat toan chan" << endl;
    } else {
        cout << "+ Day khong co tinh chat toan chan" << endl;
    }

    return 0;
}
```

Bài tập 3.
Cho dãy ký tự a có n phần tử (n ≤ 100). Hãy tạo dãy b chứa các ký tự nguyên âm của a và xuất b ra màn hình.

Moi ban nhap so luong phan tu: 5

Phan tu 0: a

Phan tu 1: b

Phan tu 2: c

Phan tu 3: d

Phan tu 4: e

+ Day co tinh chat chan le

+ Day khong co tinh chat toan chan

- Ý tưởng:
  - Sử dụng mảng một chiều và con trỏ để thao tác với dãy ký tự.
  - Nhập vào số lượng phần tử và giá trị của từng phần tử trong dãy ký tự a.
  - Tạo một dãy b chứa các ký tự nguyên âm (a, e, i, o, u) được lọc từ dãy a.
  - Xuất dãy b ra màn hình.
- Tập các testcase kiểm tra:

Testcase	Input	Kết quả mong đợi
1	Moi ban nhap so luong phan tu: 5 Phan tu 0: a Phan tu 1: b Phan tu 2: c Phan tu 3: d Phan tu 4: e	Day b chua cac ky tu nguyen am: a e
2	Moi ban nhap so luong phan tu: 5 Phan tu 0: u	Day b chua cac ky tu nguyen am: u e o a i

	Phan tu 1: e Phan tu 2: o Phan tu 3: a Phan tu 4: i	
3	Moi ban nhap so luong phan tu: 3 Phan tu 0: b Phan tu 1: c Phan tu 2: d	Day b chua cac ky tu nguyen am:
4	Moi ban nhap so luong phan tu: 0	So luong phan tu khong hop le!

- Mã nguồn

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

bool NguyenAm(char c) {
    c = tolower(c); // Chuyển ký tự về dạng chữ thường để kiểm tra
    return (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u');
}

void LocNguyenAm(char *a, int n, char *b, int *m) {
    *m = 0; // Khởi tạo số lượng phần tử của b
    for (int i = 0; i < n; i++) {
        if (NguyenAm(*(a + i))) {
            *(b + (*m)) = *(a + i); // Thêm ký tự nguyên âm vào b
            (*m)++;
        }
    }
}

int main() {
    int n;
    cout << "Moi ban nhap so luong phan tu: ";
    cin >> n;

    if (n <= 0 || n > 100) {
        cout << "So luong phan tu khong hop le!" << endl;
        return 0;
    }

    char a[100], b[100];
    int m; // Số lượng phần tử của dãy b
```

```
    for (int i = 0; i < n; i++) {
        cout << "Phan tu " << i << ": ";
        cin >> *(a + i); // Nhập giá trị cho mảng a
    }

    LocNguyenAm(a, n, b, &m);

    cout << "Day b chua cac ky tu nguyen am: ";
    for (int i = 0; i < m; i++) {
        cout << *(b + i) << " ";
    }
    cout << endl;

    return 0;
}
```

Bài tập 4.
Cho dãy số thực a có n phần tử và b có m phần tử (n,m ≤ 100) tăng dần. Hãy tạo dãy c từ hai dãy a, b sao cho tăng dần (không dùng sắp xếp).

+ Day so a

Moi ban nhap so luong phan tu: 3

Phan tu 0: 1.1

Phan tu 1: 3.2

Phan tu 2: 5.3

+ Day so b

Moi ban nhap so luong phan tu: 4

Phan tu 0: 2.1

Phan tu 1: 3.2

Phan tu 2: 5.4

Phan tu 3: 8.4

+ Day so c

Day so co 7 phan tu: 1.1 2.1 3.2 3.2 5.4 5.4 8.4

- Ý tưởng:
    - Sử dụng mảng một chiều và con trỏ để thao tác với các dãy số thực.
    - Nhập hai dãy số a và b, đảm bảo chúng được sắp xếp tăng dần.

- Gộp hai dãy a và b thành dãy c mà không cần sắp xếp, giữ nguyên thứ tự tăng dần.
- Xuất dãy c ra màn hình.

- Tập các testcase kiểm tra:

Testcase	Input	Kết quả mong đợi
1	+ Day so a Moi ban nhap so luong phan tu: 3 Phan tu 0: 1.1 Phan tu 1: 3.2 Phan tu 2: 5.3 + Day so b Moi ban nhap so luong phan tu: 2 Phan tu 0: 2.1 Phan tu 1: 4.4	+ Day so c Day so co 5 phan tu: 1.1 2.1 3.2 4.4 5.3
2	+ Day so a Moi ban nhap so luong phan tu: 3 Phan tu 0: 1.1 Phan tu 1: 3.2 Phan tu 2: 5.3 + Day so b Moi ban nhap so luong phan tu: 4 Phan tu 0: 2.1 Phan tu 1: 3.2 Phan tu 2: 5.3 Phan tu 3: 6.6	+ Day so c Day so co 7 phan tu: 1.1 2.1 3.2 3.2 5.3 5.3 6.6
3	+ Day so a Moi ban nhap so luong phan tu: 101	So luong phan tu khong hop le!
4	+ Day so a Moi ban nhap so luong phan tu: 3 Phan tu 0: -3.3 Phan tu 1: -1.1 Phan tu 2: 2.2 + Day so b	+ Day so c Day so co 5 phan tu: -3.3 -2.2 -1.1 0 2.2

	Moi ban nhap so luong phan tu: 2 Phan tu 0: -2.2 Phan tu 1: 0	

- Mã nguồn

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

void mergeArrays(double *a, int n, double *b, int m, double *c, int *k) {
    int i = 0, j = 0, idx = 0;
    while (i < n && j < m) {
        if (*(a + i) < *(b + j)) {
            *(c + idx) = *(a + i);
            i++;
        } else {
            *(c + idx) = *(b + j);
            j++;
        }
        idx++;
    }

    // Copy các phần tử còn lại của a (nếu có)
    while (i < n) {
        *(c + idx) = *(a + i);
        i++;
        idx++;
    }

    // Copy các phần tử còn lại của b (nếu có)
    while (j < m) {
        *(c + idx) = *(b + j);
        j++;
        idx++;
    }

    *k = idx; // Cập nhật số lượng phần tử của c
}

int main() {
    int n, m, k;

    cout << "+ Day so a" << endl;
    cout << "Moi ban nhap so luong phan tu: ";
    cin >> n;
    if (n <= 0 || n > 100) {
```

```
        cout << "So luong phan tu khong hop le!" << endl;
        return 0;
    }

    double a[100];
    for (int i = 0; i < n; i++) {
        cout << "Phan tu " << i << ": ";
        cin >> *(a + i);
    }

    cout << "+ Day so b" << endl;
    cout << "Moi ban nhap so luong phan tu: ";
    cin >> m;
    if (m <= 0 || m > 100) {
        cout << "So luong phan tu khong hop le!" << endl;
        return 0;
    }

    double b[100], c[200];
    for (int i = 0; i < m; i++) {
        cout << "Phan tu " << i << ": ";
        cin >> *(b + i);
    }

    mergeArrays(a, n, b, m, c, &k);

    cout << "+ Day so c" << endl;
    cout << "Day so co " << k << " phan tu: ";
    for (int i = 0; i < k; i++) {
        cout << *(c + i) << " ";
    }
    cout << endl;

    return 0;
}
```

Bài tập 5.
Cho dãy số nguyên a có n phần tử (n ≤ 100). Hãy tách dãy a thành dãy b chứa số chẵn và dãy c chứa số lẻ.

+ Day so a

Moi ban nhap so luong phan tu: 3

Phan tu 0: 2

Phan tu 1: 5

Phan tu 2: 8

+ Day so b chua so chan

Day so co 2 phan tu: 2 8

+ Day so c

Day so co 1 phan tu: 5

- Ý tưởng:
    - Input:
        - Nhập số lượng phần tử n và giá trị từng phần tử của mảng a.
    - Xử lý:
        - Sử dụng con trỏ để duyệt từng phần tử của mảng a.
        - Tạo hai mảng b và c để chứa các số chẵn và số lẻ, tương ứng.
        - Kiểm tra tính chẵn lẻ của từng phần tử trong a:
            - Nếu số đó chẵn, thêm vào b.
            - Nếu số đó lẻ, thêm vào c.
    - Output:
        - In ra các phần tử của dãy b (chứa số chẵn).
        - In ra các phần tử của dãy c (chứa số lẻ).
- Tập các testcase kiểm tra:

Testcase	Input	Kết quả mong đợi
1	+ Day so a Moi ban nhap so luong phan tu: 3 Phan tu 0: 2 Phan tu 1: 5 Phan tu 2: 8	+ Day so b chua so chan Day so co 2 phan tu: 2 8 + Day so c chua so le Day so co 1 phan tu: 5
2	+ Day so a Moi ban nhap so luong phan tu: 5 Phan tu 0: 1 Phan tu 1: 2 Phan tu 2: 3 Phan tu 3: 4 Phan tu 4: 5	+ Day so b chua so chan Day so co 2 phan tu: 2 4 + Day so c chua so le Day so co 3 phan tu: 1 3 5
3	+ Day so a Moi ban nhap so luong phan tu: 0	So luong phan tu khong hop le!

- Mã nguồn

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

void ChiaMang(int *a, int n, int *b, int *m, int *c, int *k) {
    *m = 0; // Số lượng phần tử chẵn trong b
    *k = 0; // Số lượng phần tử lẻ trong c
    for (int i = 0; i < n; i++) {
        if (*(a + i) % 2 == 0) {
            *(b + (*m)) = *(a + i); // Thêm số chẵn vào b
            (*m)++;
        } else {
            *(c + (*k)) = *(a + i); // Thêm số lẻ vào c
            (*k)++;
        }
    }
}

int main() {
    int n;
    cout << "+ Day so a" << endl;
    cout << "Moi ban nhap so luong phan tu: ";
    cin >> n;

    if (n <= 0 || n > 100) {
        cout << "So luong phan tu khong hop le!" << endl;
        return 0;
    }

    int a[100], b[100], c[100];
    int m, k; // Số lượng phần tử của dãy b và c

    for (int i = 0; i < n; i++) {
        cout << "Phan tu " << i << ": ";
        cin >> *(a + i);
    }

    ChiaMang(a, n, b, &m, c, &k);

    cout << "+ Day so b chua so chan\nDay so co " << m << " phan tu: ";
    for (int i = 0; i < m; i++) {
        cout << *(b + i) << " ";
    }
    cout << endl;

    cout << "+ Day so c chua so le\nDay so co " << k << " phan tu: ";
    for (int i = 0; i < k; i++) {
        cout << *(c + i) << " ";
```

```
    }
    cout << endl;

    return 0;
}
```

Cấu trúc

Bài tập 1.
Một phòng ban có tối đa 100 nhân viên, mỗi nhân viên (NhanVien) gồm các thông tin sau: mã số (MaSo) – chuỗi tối đa 10 ký tự, họ lót (Ho) – chuỗi tối đa 10 ký tự, tên (Ten) – chuỗi tối đa 50 ký tự, Phái (Phai) – số nguyên có giá trị 0 cho Nữ và 1 cho Nam, số năm làm việc (ThamNien) – số nguyên dương lớn hơn hay bằng 0. Phòng ban (PhongBan) bao gồm mảng tối đa 100 nhân viên (aNhanVien) và số lượng nhân viên hiện có (SoLuong). Hãy cho biết sĩ số nam, nữ và in ra danh sách nhân viên tăng dần theo thâm niên.

Yêu cầu: Viết chương trình thực hiện yêu cầu đề bài, gồm các phần sau:

a) Khai báo cấu trúc NhanVien và PhongBan với thông tin mô tả phía trên.

b) Hàm nhập danh sách các nhân viên cho phòng ban có ràng buộc dữ liệu.

void NhapPhongBan(PhongBan &pb) {...}

c) Hàm xuất thông tin phòng ban ra màn hình.

void XuatPhongBan(PhongBan pb) {...}

d) Hàm đếm sĩ số nhân viên trong phòng ban.

void DemSiSo(PhongBan ds, int &sonam, int &sonu) {...}

e) Hàm sắp xếp danh sách nhân viên tăng dần theo thâm niên.

void SapXepTangTheoThamNien(PhongBan &pb) {...}

f) Hàm main sử dụng các hàm trên xử lý các yêu cầu đề bài.

int main() {...}

- Ý tưởng
    - Cấu trúc dữ liệu:
        - NhanVien:
            - MaSo: chuỗi tối đa 10 ký tự.
            - Ho: chuỗi tối đa 10 ký tự.
            - Ten: chuỗi tối đa 50 ký tự.
            - Phai: số nguyên (0 = Nữ, 1 = Nam).
            - ThamNien: số nguyên dương ≥ 0.
        - PhongBan:
            - aNhanVien: mảng tối đa 100 nhân viên.
            - SoLuong: số lượng nhân viên hiện có.
    - Chức năng chính:
        - Nhập danh sách nhân viên (NhapPhongBan):
            - Nhập số lượng nhân viên (kiểm tra không vượt quá 100).

        - Nhập thông tin từng nhân viên với ràng buộc dữ liệu hợp lệ.
    - Xuất danh sách nhân viên (XuatPhongBan):
        - In thông tin từng nhân viên ra màn hình.
    - Đếm sĩ số (DemSiSo):
        - Đếm số lượng nhân viên nam và nữ trong phòng ban.
    - Sắp xếp danh sách tăng dần theo thâm niên (SapXepTangTheoThamNien):
        - Sử dụng thuật toán sắp xếp (bằng Bubble Sort hoặc Selection Sort).
    - Hàm main:
        - Gọi các hàm trên theo yêu cầu đề bài.
- Testcase kiểm tra:

Testcase	Input	Kết quả mong đợi
1	Nhap so luong nhan vien: 3 Nhap thong tin nhan vien thu 1: Nhap Ma So (toi da 10 ky tu): NV001 Nhap Ho (toi da 10 ky tu): Nguyen Nhap Ten (toi da 50 ky tu): An Nhap Phai (0 = Nu, 1 = Nam): 1 Nhap Tham Nien (>= 0): 5 Nhap thong tin nhan vien thu 2: Nhap Ma So (toi da 10 ky tu): NV002 Nhap Ho (toi da 10 ky tu): Tran Nhap Ten (toi da 50 ky tu): Binh Nhap Phai (0 = Nu, 1 = Nam): 0 Nhap Tham Nien (>= 0): 3 Nhap thong tin nhan vien thu 3: Nhap Ma So (toi da 10 ky tu): NV003 Nhap Ho (toi da 10 ky tu): Le Nhap Ten (toi da 50 ky tu): Cuong Nhap Phai (0 = Nu, 1 = Nam): 1 Nhap Tham Nien (>= 0): 7	Danh sach nhan vien trong phong ban: Nhan vien thu 1: Ma So: NV001, Ho: Nguyen, Ten: An, Phai: Nam, Tham Nien: 5 nam Nhan vien thu 2: Ma So: NV002, Ho: Tran, Ten: Binh, Phai: Nu, Tham Nien: 3 nam Nhan vien thu 3: Ma So: NV003, Ho: Le, Ten: Cuong, Phai: Nam, Tham Nien: 7 nam Si so nam: 2, si so nu: 1 Danh sach nhan vien tang dan theo tham nien: Danh sach nhan vien trong phong ban: Nhan vien thu 1: Ma So: NV002, Ho: Tran, Ten: Binh, Phai: Nu, Tham Nien: 3 nam Nhan vien thu 2: Ma So: NV001, Ho: Nguyen, Ten: An, Phai: Nam, Tham Nien: 5 nam Nhan vien thu 3: Ma So: NV003, Ho: Le, Ten: Cuong, Phai: Nam, Tham Nien: 7 nam
2	Nhap so luong nhan vien: -1	Nhap so luong nhan vien:

3	Nhap so luong nhan vien: 101	Nhap so luong nhan vien:

- Mã nguồn

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

struct NhanVien {
    char MaSo[11];
    char Ho[11];
    char Ten[51];
    int Phai;        // 0 = Nữ, 1 = Nam
    int ThamNien;    // Số năm làm việc ≥ 0
};

struct PhongBan {
    NhanVien aNhanVien[100];
    int SoLuong; // Số lượng nhân viên hiện có
};

void NhapNhanVien(NhanVien &nv) {
    cout << "Nhap Ma So (toi da 10 ky tu): ";
    cin >> nv.MaSo;
    cout << "Nhap Ho (toi da 10 ky tu): ";
    cin >> nv.Ho;
    cout << "Nhap Ten (toi da 50 ky tu): ";
    cin.ignore();
    cin.getline(nv.Ten, 51);
    do {
        cout << "Nhap Phai (0 = Nu, 1 = Nam): ";
        cin >> nv.Phai;
    } while (nv.Phai != 0 && nv.Phai != 1);
    do {
        cout << "Nhap Tham Nien (>= 0): ";
        cin >> nv.ThamNien;
    } while (nv.ThamNien < 0);
}

void NhapPhongBan(PhongBan &pb) {
    do {
        cout << "Nhap so luong nhan vien: ";
        cin >> pb.SoLuong;
    } while (pb.SoLuong <= 0 || pb.SoLuong > 100);
    for (int i = 0; i < pb.SoLuong; i++) {
        cout << "Nhap thong tin nhan vien thu " << i + 1 << ":\n";
        NhapNhanVien(pb.aNhanVien[i]);
    }
```

```
}

void XuatNhanVien(NhanVien nv) {
    cout << "Ma So: " << nv.MaSo << ", Ho: " << nv.Ho
         << ", Ten: " << nv.Ten << ", Phai: " << (nv.Phai == 1 ? "Nam" : "Nu")
         << ", Tham Nien: " << nv.ThamNien << " nam\n";
}

void XuatPhongBan(PhongBan pb) {
    cout << "Danh sach nhan vien trong phong ban:\n";
    for (int i = 0; i < pb.SoLuong; i++) {
        cout << "Nhan vien thu " << i + 1 << ":\n";
        XuatNhanVien(pb.aNhanVien[i]);
    }
}

void DemSiSo(PhongBan pb, int &soNam, int &soNu) {
    soNam = soNu = 0;
    for (int i = 0; i < pb.SoLuong; i++) {
        if (pb.aNhanVien[i].Phai == 1)
            soNam++;
        else
            soNu++;
    }
}

void SapXepTangTheoThamNien(PhongBan &pb) {
    for (int i = 0; i < pb.SoLuong - 1; i++) {
        for (int j = i + 1; j < pb.SoLuong; j++) {
            if (pb.aNhanVien[i].ThamNien > pb.aNhanVien[j].ThamNien) {
                NhanVien temp = pb.aNhanVien[i];
                pb.aNhanVien[i] = pb.aNhanVien[j];
                pb.aNhanVien[j] = temp;
            }
        }
    }
}

int main() {
    PhongBan pb;
    int soNam, soNu;

    NhapPhongBan(pb);
    XuatPhongBan(pb);

    DemSiSo(pb, soNam, soNu);
    cout << "Si so nam: " << soNam << ", si so nu: " << soNu << "\n";
```

```
    SapXepTangTheoThamNien(pb);
    cout << "Danh sach nhan vien tang dan theo tham nien:\n";
    XuatPhongBan(pb);

    return 0;
}
```

Bài tập 2.
Một cửa hàng bán hoa có tối đa 20 loại hoa, mỗi loại hoa gồm các thông tin sau: Tên loại (Ten), Số lượng (SoLuong), Đơn vị tính (DVT), Đơn giá (DonGia). Với tên loại, số lượng mà khách hàng yêu cầu, hãy cho biết loại đó có hay không. Nếu có, có đủ số lượng bán hay không và tổng tiền nếu đủ số lượng bán.

Yêu cầu: Viết chương trình thực hiện yêu cầu đề bài, gồm các phần sau:

a) Khai báo cấu trúc LoaiHoa và DanhSachLoaiHoa với thông tin mô tả phía trên.

b) Hàm nhập danh sách các loại hoa.

void NhapDanhSach(DanhSachLoaiHoa &ds) {...}

c) Hàm xuất danh sách ra màn hình.

void XuatDanhSach(DanhSachLoaiHoa ds) {...}

d) Hàm tìm loại hoa theo tên loại.

int TimLoaiHoa(DanhSachLoaiHoa ds, char *tenloai) {...}

Trả về vị trí loại hoa trong danh sách, nếu không có trả về -1.

e) Hàm xử lý bán hoa cho khách hàng biết trước tên loại và số lượng.

void XuLyBanHoa(DanhSachLoaiHoa &ds,char *tenloai, int soluong) {...}

Chúc năng: Kiểm tra hoa có tồn tại không. Nếu có, kiểm tra xem số lượng có đủ không. Nếu đủ, in giá tiền, biết giá tiền bằng số lượng khách hàng mua * đơn giá.

f) Hàm main sử dụng các hàm trên xử lý các yêu cầu đề bài.

void main() {...}

- Ý tưởng
    - Cấu trúc dữ liệu:
        - LoaiHoa:
            - Ten: tên loại hoa (chuỗi, tối đa 50 ký tự).
            - SoLuong: số lượng hoa hiện có (số nguyên ≥ 0).
            - DVT: đơn vị tính (chuỗi, tối đa 10 ký tự, ví dụ: "bó", "cành").

        - DonGia: đơn giá bán (số thực ≥ 0).
    - DanhSachLoaiHoa:
        - dsHoa: mảng tối đa 20 loại hoa.
        - SoLuong: số lượng loại hoa hiện có.
- Chức năng chính:
    - Nhập danh sách các loại hoa (NhapDanhSach):
        - Nhập thông tin từng loại hoa với ràng buộc dữ liệu hợp lệ.
    - Xuất danh sách loại hoa (XuatDanhSach):
        - In thông tin từng loại hoa ra màn hình.
    - Tìm loại hoa theo tên (TimLoaiHoa):
        - Tìm kiếm tên loại hoa trong danh sách.
        - Trả về vị trí nếu tìm thấy, ngược lại trả về -1.
    - Xử lý bán hoa (XuLyBanHoa):
        - Kiểm tra loại hoa có trong danh sách không.
        - Nếu có:
            - Kiểm tra số lượng có đủ không.
            - Nếu đủ, tính tổng tiền và giảm số lượng trong kho.
            - Nếu không đủ, thông báo số lượng không đủ.
        - Nếu không có, thông báo không tìm thấy.
    - Hàm main:
        - Gọi các hàm trên theo yêu cầu đề bài.

- Testcase kiểm tra:

Testcase	Input	Kết quả mong đợi
1	Nhap so luong loai hoa (toi da 20): 3 Nhap thong tin loai hoa thu 1: Nhap ten loai hoa (toi da 50 ky tu): Hoa Hong Nhap so luong (>= 0): 50 Nhap don vi tinh (toi da 10 ky tu): canh Nhap don gia (>= 0): 10.5 Nhap thong tin loai hoa thu 2: Nhap ten loai hoa (toi da 50 ky tu): Hoa Lan Nhap so luong (>= 0): 30 Nhap don vi tinh (toi da 10 ky tu): canh Nhap don gia (>= 0): 15 Nhap thong tin loai hoa thu 3:	Danh sach cac loai hoa: Loai hoa thu 1: Ten: Hoa Hong, So luong: 50, Don vi tinh: canh, Don gia: 10.5 Loai hoa thu 2: Ten: Hoa Lan, So luong: 30, Don vi tinh: canh, Don gia: 15 Loai hoa thu 3: Ten: Hoa Cuc, So luong: 40, Don vi tinh: bong, Don gia: 8

	Nhap ten loai hoa (toi da 50 ky tu): Hoa Cuc Nhap so luong (>= 0): 40 Nhap don vi tinh (toi da 10 ky tu): bong Nhap don gia (>= 0): 8 Nhap ten loai hoa can mua: Hoa Hong Nhap so luong can mua: 20	Danh sach sau khi ban: Danh sach cac loai hoa: Loai hoa thu 1: Ten: Hoa Hong, So luong: 30, Don vi tinh: canh, Don gia: 10.5 Loai hoa thu 2: Ten: Hoa Lan, So luong: 30, Don vi tinh: canh, Don gia: 15 Loai hoa thu 3: Ten: Hoa Cuc, So luong: 40, Don vi tinh: bong, Don gia: 8
2	Nhap so luong loai hoa (toi da 20): 21	Nhap so luong loai hoa (toi da 20):
3	Nhap so luong loai hoa (toi da 20): 1 Nhap thong tin loai hoa thu 1: Nhap ten loai hoa (toi da 50 ky tu): Hoa Hong Nhap so luong (>= 0): 10 Nhap don vi tinh (toi da 10 ky tu): canh Nhap don gia (>= 0): 20000 Danh sach cac loai hoa: Loai hoa thu 1: Ten: Hoa Hong, So luong: 10, Don vi tinh: canh, Don gia: 20000 Nhap ten loai hoa can mua: Hoa Lan Nhap so luong can mua: 2	Khong tim thay loai hoa "Hoa Lan" trong danh sach. Danh sach sau khi ban: Danh sach cac loai hoa: Loai hoa thu 1: Ten: Hoa Hong, So luong: 10, Don vi tinh: canh, Don gia: 20000

- Mã nguồn

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

```cpp
struct LoaiHoa {
    char Ten[51];       // Tên loại hoa
    int SoLuong;        // Số lượng hiện có
    char DVT[11];       // Đơn vị tính
    float DonGia;       // Đơn giá
};

struct DanhSachLoaiHoa {
    LoaiHoa dsHoa[20];  // Mảng tối đa 20 loại hoa
    int SoLuong;        // Số lượng loại hoa hiện có
};

void NhapLoaiHoa(LoaiHoa &hoa) {
    cout << "Nhap ten loai hoa (toi da 50 ky tu): ";
    cin.ignore();
    cin.getline(hoa.Ten, 51);
    do {
        cout << "Nhap so luong (>= 0): ";
        cin >> hoa.SoLuong;
    } while (hoa.SoLuong < 0);
    cout << "Nhap don vi tinh (toi da 10 ky tu): ";
    cin.ignore();
    cin.getline(hoa.DVT, 11);
    do {
        cout << "Nhap don gia (>= 0): ";
        cin >> hoa.DonGia;
    } while (hoa.DonGia < 0);
}

void NhapDanhSach(DanhSachLoaiHoa &ds) {
    do {
        cout << "Nhap so luong loai hoa (toi da 20): ";
        cin >> ds.SoLuong;
    } while (ds.SoLuong <= 0 || ds.SoLuong > 20);
    for (int i = 0; i < ds.SoLuong; i++) {
        cout << "Nhap thong tin loai hoa thu " << i + 1 << ":\n";
        NhapLoaiHoa(ds.dsHoa[i]);
    }
}

void XuatLoaiHoa(LoaiHoa hoa) {
    cout << "Ten: " << hoa.Ten << ", So luong: " << hoa.SoLuong
         << ", Don vi tinh: " << hoa.DVT << ", Don gia: " << hoa.DonGia << "\n";
}

void XuatDanhSach(DanhSachLoaiHoa ds) {
    cout << "Danh sach cac loai hoa:\n";
```

```
    for (int i = 0; i < ds.SoLuong; i++) {
        cout << "Loai hoa thu " << i + 1 << ":\n";
        XuatLoaiHoa(ds.dsHoa[i]);
    }
}

int TimLoaiHoa(DanhSachLoaiHoa ds, char *tenloai) {
    for (int i = 0; i < ds.SoLuong; i++) {
        if (strcmp(ds.dsHoa[i].Ten, tenloai) == 0)
            return i;
    }
    return -1;
}

void XuLyBanHoa(DanhSachLoaiHoa &ds, char *tenloai, int soluong) {
    int viTri = TimLoaiHoa(ds, tenloai);
    if (viTri == -1) {
        cout << "Khong tim thay loai hoa \"" << tenloai << "\" trong danh sach.\n";
        return;
    }

    LoaiHoa &hoa = ds.dsHoa[viTri];
    if (hoa.SoLuong >= soluong) {
        float tongTien = soluong * hoa.DonGia;
        hoa.SoLuong -= soluong;
        cout << "Ban thanh cong " << soluong << " " << hoa.DVT
             << " hoa \"" << hoa.Ten << "\". Tong tien: " << tongTien << "\n";
    } else {
        cout << "Khong du so luong hoa \"" << hoa.Ten << "\" trong kho. "
             << "So luong hien co: " << hoa.SoLuong << "\n";
    }
}

int main() {
    DanhSachLoaiHoa ds;
    NhapDanhSach(ds);
    XuatDanhSach(ds);

    char tenHoa[51];
    int soLuongMua;

    cout << "\nNhap ten loai hoa can mua: ";
    cin.ignore();
    cin.getline(tenHoa, 51);
    cout << "Nhap so luong can mua: ";
    cin >> soLuongMua;
```

```
    XuLyBanHoa(ds, tenHoa, soLuongMua);
    cout << "\nDanh sach sau khi ban:\n";
    XuatDanhSach(ds);

    return 0;
}
```

Bài tập 3.

Quản lý điện thoại Tên file chương trình:QuanLyDienThoai.*

Thông tin một điện thoại gồm:

• Mã điện thoại (tối đa 10 ký tự)

• Nhãn hiệu (tối đa 20 ký tự)

• Giá (số nguyên)

Viết chương trình quản lý một danh sách n (n ≤ 100) điện thoại. Chương trình có những chức năng sau:

a) Thêm 1 điện thoại vào danh sách điện điện thoại. Nếu điện thoại đã tồn tại trong danh sách thì hay danh sách bị đầy thì thông báo lỗi.

b) Cho mã điện thoại, in nhãn hiệu và giá của điện thoại đó. Nếu mã điện thoại không có trong danh sách thì báo lỗi.

c) Cho mã điện thoại, cập nhật lại giá của điện thoại đó. Nếu mã điện thoại không có trong danh sách thì báo lỗi.

d) Cho mã điện thoại, xóa điện thoại đó. Nếu mã điện thoại không có trong danh sách thì báo lỗi.

e) Xuất tất cả điện thoại trong danh sách.

f) Xây dựng menu cho chương trình này.

- Ý tưởng
    - Cấu trúc dữ liệu:
        - DienThoai:
            - MaDT (char[]): Mã điện thoại (tối đa 10 ký tự).
            - NhanHieu (char[]): Nhãn hiệu điện thoại (tối đa 20 ký tự).
            - Gia (int): Giá tiền của điện thoại.
        - DanhSachDienThoai:
            - ds (mảng DienThoai): Mảng chứa danh sách điện thoại (tối đa 100 phần tử).
            - SoLuong (int): Số lượng điện thoại hiện có trong danh sách.

- Chức năng chính
    - Thêm điện thoại (ThemDienThoai):
        - Kiểm tra danh sách có đầy không.
        - Kiểm tra mã điện thoại đã tồn tại chưa.
        - Nếu hợp lệ, thêm điện thoại vào danh sách.
    - Tìm điện thoại theo mã (TimDienThoai):
        - Tìm điện thoại trong danh sách theo mã.
        - Trả về vị trí của điện thoại trong danh sách, hoặc -1 nếu không tìm thấy.
    - In thông tin điện thoại theo mã (InThongTinDienThoai):
        - Kiểm tra mã điện thoại.
        - Nếu tìm thấy, in nhãn hiệu và giá.
        - Nếu không tìm thấy, báo lỗi.
    - Cập nhật giá điện thoại (CapNhatGia):
        - Kiểm tra mã điện thoại.
        - Nếu tìm thấy, cập nhật giá.
        - Nếu không tìm thấy, báo lỗi.
    - Xóa điện thoại (XoaDienThoai):
        - Kiểm tra mã điện thoại.
        - Nếu tìm thấy, xóa bằng cách dồn các phần tử sau lên trước.
        - Nếu không tìm thấy, báo lỗi.
    - Xuất danh sách điện thoại (XuatDanhSach):
        - In toàn bộ danh sách điện thoại ra màn hình.
    - Menu:
        - Xây dựng menu để người dùng lựa chọn các chức năng.

- Testcase kiểm tra:

Testcase	Input (tuần tự các chức năng)	Kết quả mong đợi
1	--- MENU --- 1. Them dien thoai 2. In thong tin dien thoai theo ma 3. Cap nhat gia dien thoai 4. Xoa dien thoai 5. Xuat danh sach dien thoai 0. Thoat Lua chon cua ban: 1 Nhap ma dien thoai (toi da 10 ky tu): DT001 Nhap nhan hieu (toi da 20 ky tu): Samsung	Them dien thoai thanh cong.

	Nhap gia: 15000000 Lua chon cua ban: 1 Nhap ma dien thoai (toi da 10 ky tu): DT002 Nhap nhan hieu (toi da 20 ky tu): Nokia Nhap gia: 10000000 Lua chon cua ban: 1 Nhap ma dien thoai (toi da 10 ky tu): DT003 Nhap nhan hieu (toi da 20 ky tu): Iphone Nhap gia: 20000000 Lua chon cua ban: 2 Nhap ma dien thoai: DT002 Lua chon cua ban: 3 Nhap ma dien thoai: DT001 Nhap gia moi: 16000000 Lua chon cua ban: 4 Nhap ma dien thoai: DT003 Lua chon cua ban: 5 Lua chon cua ban: 0	Them dien thoai thanh cong. Them dien thoai thanh cong. Nhan hieu: Nokia, Gia: 10000000 Cap nhat gia thanh cong. Xoa dien thoai thanh cong. Ma: DT001, Nhan hieu: Samsung, Gia: 16000000 Ma: DT002, Nhan hieu: Nokia, Gia: 10000000 Thoat chuong trinh.
2	Lua chon cua ban: 1 Nhap ma dien thoai (toi da 10 ky tu): DT001 Nhap nhan hieu (toi da 20 ky tu): Asus	Dien thoai da ton tai trong danh sach.

	Nhap gia: 12000000	
3	Lua chon cua ban: 2 Nhap ma dien thoai: DT004	Khong tim thay dien thoai voi ma "DT004".
4	Lua chon cua ban: 3 Nhap ma dien thoai: DT004 Nhap gia moi: 20000000	Khong tim thay dien thoai voi ma "DT004".
5	Lua chon cua ban: 4 Nhap ma dien thoai: DT004	Khong tim thay dien thoai voi ma "DT004".
6	(Nếu không có mã điện thoại nào trong danh sách) Lua chon cua ban: 5	Danh sach dien thoai rong.

- Mã nguồn

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

struct DienThoai {
    char MaDT[11];     // Mã điện thoại
    char NhanHieu[21]; // Nhãn hiệu
    int Gia;           // Giá
};

struct DanhSachDienThoai {
    DienThoai ds[100]; // Mảng chứa danh sách điện thoại
    int SoLuong;       // Số lượng điện thoại hiện có
};

void ThemDienThoai(DanhSachDienThoai &ds, DienThoai dt) {
    if (ds.SoLuong >= 100) {
        cout << "Danh sach da day. Khong the them dien thoai.\n";
        return;
    }
    for (int i = 0; i < ds.SoLuong; i++) {
        if (strcmp(ds.ds[i].MaDT, dt.MaDT) == 0) {
            cout << "Dien thoai da ton tai trong danh sach.\n";
            return;
        }
    }
    ds.ds[ds.SoLuong++] = dt;
    cout << "Them dien thoai thanh cong.\n";
}

int TimDienThoai(DanhSachDienThoai ds, char *maDT) {
    for (int i = 0; i < ds.SoLuong; i++) {
        if (strcmp(ds.ds[i].MaDT, maDT) == 0)
```

```
            return i;
    }
    return -1;
}

void InThongTinDienThoai(DanhSachDienThoai ds, char *maDT) {
    int index = TimDienThoai(ds, maDT);
    if (index == -1) {
        cout << "Khong tim thay dien thoai voi ma \"" << maDT << "\".\n";
    } else {
        DienThoai dt = ds.ds[index];
        cout << "Nhan hieu: " << dt.NhanHieu << ", Gia: " << dt.Gia << "\n";
    }
}

void CapNhatGia(DanhSachDienThoai &ds, char *maDT, int giaMoi) {
    int index = TimDienThoai(ds, maDT);
    if (index == -1) {
        cout << "Khong tim thay dien thoai voi ma \"" << maDT << "\".\n";
    } else {
        ds.ds[index].Gia = giaMoi;
        cout << "Cap nhat gia thanh cong.\n";
    }
}

void XoaDienThoai(DanhSachDienThoai &ds, char *maDT) {
    int index = TimDienThoai(ds, maDT);
    if (index == -1) {
        cout << "Khong tim thay dien thoai voi ma \"" << maDT << "\".\n";
    } else {
        for (int i = index; i < ds.SoLuong - 1; i++) {
            ds.ds[i] = ds.ds[i + 1];
        }
        ds.SoLuong--;
        cout << "Xoa dien thoai thanh cong.\n";
    }
}

void XuatDanhSach(DanhSachDienThoai ds) {
    if (ds.SoLuong == 0) {
        cout << "Danh sach dien thoai rong.\n";
        return;
    }
    for (int i = 0; i < ds.SoLuong; i++) {
        DienThoai dt = ds.ds[i];
        cout << "Ma: " << dt.MaDT << ", Nhan hieu: " << dt.NhanHieu << ", Gia: " <<
dt.Gia << "\n";
```

```
    }
}

void Menu() {
    DanhSachDienThoai ds = { {}, 0 };
    int choice;
    do {
        cout << "\n--- MENU ---\n";
        cout << "1. Them dien thoai\n";
        cout << "2. In thong tin dien thoai theo ma\n";
        cout << "3. Cap nhat gia dien thoai\n";
        cout << "4. Xoa dien thoai\n";
        cout << "5. Xuat danh sach dien thoai\n";
        cout << "0. Thoat\n";
        cout << "Lua chon cua ban: ";
        cin >> choice;

        switch (choice) {
            case 1: {
                DienThoai dt;
                cout << "Nhap ma dien thoai (toi da 10 ky tu): ";
                cin.ignore();
                cin.getline(dt.MaDT, 11);
                cout << "Nhap nhan hieu (toi da 20 ky tu): ";
                cin.getline(dt.NhanHieu, 21);
                cout << "Nhap gia: ";
                cin >> dt.Gia;
                ThemDienThoai(ds, dt);
                break;
            }
            case 2: {
                char maDT[11];
                cout << "Nhap ma dien thoai: ";
                cin.ignore();
                cin.getline(maDT, 11);
                InThongTinDienThoai(ds, maDT);
                break;
            }
            case 3: {
                char maDT[11];
                int giaMoi;
                cout << "Nhap ma dien thoai: ";
                cin.ignore();
                cin.getline(maDT, 11);
                cout << "Nhap gia moi: ";
                cin >> giaMoi;
                CapNhatGia(ds, maDT, giaMoi);
```

```
                break;
            }
            case 4: {
                char maDT[11];
                cout << "Nhap ma dien thoai: ";
                cin.ignore();
                cin.getline(maDT, 11);
                XoaDienThoai(ds, maDT);
                break;
            }
            case 5: {
                XuatDanhSach(ds);
                break;
            }
            case 0:
                cout << "Thoat chuong trinh.\n";
                break;
            default:
                cout << "Lua chon khong hop le.\n";
        }
    } while (choice != 0);
}

int main() {
    Menu();
    return 0;
}
```

Bài tập 4.

Cửa hàng tạp hoa Tên file chương trình:CuaHangTapHoa.*

Một cửa hàng tạp hóa cần quản lý danh mục mặt hàng trong cửa hàng. Mỗi mặt hàng gồm các thông tin sau:

• MSMH (Mã số mặt hàng)

• TenMH (Tên mặt hàng)

• SoLuong (Số lượng mặt hàng)

• DonGia (Đơn giá)

Viết chương trình thực hiện các chức năng:

a) Nhập n mặt hàng (n ≤ 100)

b) Cho tên mặt hàng, hãy in ra thông tin đầy đủ của mặt hàng có tên đó (Nếu không tìm thấy thì thông báo “Khong co mat hang dang tim”)

c) Tính tổng số lượng các mặt hàng

- Ý tưởng
    - Cấu trúc dữ liệu:
        - Bài toán yêu cầu quản lý danh mục mặt hàng tại một cửa hàng tạp hóa.
        - Danh sách các mặt hàng sẽ được lưu trong một mảng với tối đa 100 phần tử.
        - Chương trình cần thực hiện ba chức năng chính: nhập danh sách, tìm kiếm mặt hàng theo tên, và tính tổng số lượng mặt hàng.
    - Chức năng chính:
        - Nhập danh sách mặt hàng
            - Dữ liệu cần nhập:
                - MSMH (Mã số mặt hàng): Chuỗi tối đa 10 ký tự.
                - TenMH (Tên mặt hàng): Chuỗi tối đa 20 ký tự.
                - SoLuong (Số lượng mặt hàng): Số nguyên dương.
                - DonGia (Đơn giá): Số nguyên dương.
            - Ràng buộc dữ liệu:
                - Mã số mặt hàng không được trống.
                - Số lượng và đơn giá phải lớn hơn 0.
        - Tìm mặt hàng theo tên
            - Input: Tên mặt hàng cần tìm (TenMH).
            - Quy trình:
                - Duyệt qua danh sách mặt hàng.
                - So sánh tên mặt hàng nhập vào với tên trong danh sách:
                    - Nếu tìm thấy, trả về thông tin đầy đủ của mặt hàng.
                    - Nếu duyệt hết mà không tìm thấy, thông báo "Không có mặt hàng đang tìm".
        - Tính tổng số lượng mặt hàng
            - Khởi tạo biến tổng (tongSoLuong) bằng 0.
            - Duyệt qua danh sách mặt hàng, cộng dồn số lượng từng mặt hàng vào biến tổng.
            - Xuất kết quả tổng số lượng.
        - Hàm main
            - Nhập danh sách mặt hàng.
            - Yêu cầu người dùng nhập tên mặt hàng cần tìm và hiển thị kết quả.
            - Tính và xuất tổng số lượng mặt hàng.
- Testcase kiểm tra:

Testcase	Input	Kết quả mong đợi

1	Nhap so luong mat hang: 3 Nhap mat hang thu 1: Nhap ma so mat hang: MH001 Nhap ten mat hang: Gao Nhap so luong: 50 Nhap don gia: 20000 Nhap mat hang thu 2: Nhap ma so mat hang: MH002 Nhap ten mat hang: Thit Nhap so luong: 30 Nhap don gia: 40000 Nhap mat hang thu 3: Nhap ma so mat hang: MH003 Nhap ten mat hang: Rau Nhap so luong: 40 Nhap don gia: 15000 Nhap ten mat hang muon tim: Gao	Mat hang tim duoc: MSMH: MH001 Ten mat hang: Gao So luong: 50 Don gia: 20000 Tong so luong mat hang: 120
2	Nhap so luong mat hang: 3 Nhap mat hang thu 1: Nhap ma so mat hang: MH001 Nhap ten mat hang: Gao Nhap so luong: 20 Nhap don gia: 20000 Nhap mat hang thu 2: Nhap ma so mat hang: MH002 Nhap ten mat hang: Thit Nhap so luong: 30 Nhap don gia: 40000 Nhap mat hang thu 3: Nhap ma so mat hang: MH003 Nhap ten mat hang: Rau Nhap so luong: 40 Nhap don gia: 14000 Nhap ten mat hang muon tim: Banh	Khong co mat hang dang tim. Tong so luong mat hang: 90

- Mã nguồn

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

#define MAX 100

struct MatHang {
    char MSMH[11];
```

```
    char TenMH[21];
    int SoLuong;
    int DonGia;
};

void NhapMatHang(MatHang &mh) {
    cout << "Nhap ma so mat hang: ";
    cin.ignore();
    cin.getline(mh.MSMH, 11);
    cout << "Nhap ten mat hang: ";
    cin.getline(mh.TenMH, 21);
    cout << "Nhap so luong: ";
    cin >> mh.SoLuong;
    cout << "Nhap don gia: ";
    cin >> mh.DonGia;
}

void NhapDanhSach(MatHang ds[], int &n) {
    cout << "Nhap so luong mat hang: ";
    cin >> n;
    for (int i = 0; i < n; i++) {
        cout << "Nhap mat hang thu " << i + 1 << ":\n";
        NhapMatHang(ds[i]);
    }
}

void XuatMatHang(const MatHang &mh) {
    cout << "MSMH: " << mh.MSMH << endl;
    cout << "Ten mat hang: " << mh.TenMH << endl;
    cout << "So luong: " << mh.SoLuong << endl;
    cout << "Don gia: " << mh.DonGia << endl;
}

int TimMatHangTheoTen(MatHang ds[], int n, const char *tenMH, MatHang &ketQua) {
    for (int i = 0; i < n; i++) {
        if (strcmp(ds[i].TenMH, tenMH) == 0) {
            ketQua = ds[i];
            return 1; // Tìm thấy
        }
    }
    return 0; // Không tìm thấy
}

int TinhTongSoLuong(MatHang ds[], int n) {
    int tong = 0;
    for (int i = 0; i < n; i++) {
        tong += ds[i].SoLuong;
```

```
    }
    return tong;
}

int main() {
    MatHang ds[MAX];
    int n;

    // Nhập danh sách mặt hàng
    NhapDanhSach(ds, n);

    // Tìm mặt hàng theo tên
    char tenTimKiem[21];
    cout << "Nhap ten mat hang muon tim: ";
    cin.ignore();
    cin.getline(tenTimKiem, 21);

    MatHang ketQua;
    if (TimMatHangTheoTen(ds, n, tenTimKiem, ketQua)) {
        cout << "Mat hang tim duoc:\n";
        XuatMatHang(ketQua);
    } else {
        cout << "Khong co mat hang dang tim.\n";
    }

    // Tính tổng số lượng mặt hàng
    int tongSoLuong = TinhTongSoLuong(ds, n);
    cout << "Tong so luong mat hang: " << tongSoLuong << endl;

    return 0;
}
```

Xử lý chuỗi

Bài tập 1.
Cho chuỗi s (| s | ≤ 1.000). Hãy sắp các ký tự trong chuỗi theo thứ tự tăng dần.

Moi ban nhap chuoi s: abcba

Chuoi “abcba” sau khi sap xep: aabc

- Ý tưởng
  - Cấu trúc dữ liệu:
    - Bài toán yêu cầu sắp xếp các ký tự trong một chuỗi theo thứ tự tăng dần (dựa trên mã ASCII).
    - Độ dài của chuỗi không quá 1.000 ký tự.
    - Sau khi sắp xếp, kết quả là chuỗi có các ký tự được sắp xếp theo thứ tự tăng dần.
  - Chức năng chính:
    - Nhập chuỗi
      - Nhập chuỗi từ bàn phím.
      - Kiểm tra độ dài chuỗi để đảm bảo không vượt quá 1.000 ký tự.
    - Sắp xếp chuỗi
      - Sử dụng hàm sắp xếp (sort) có sẵn trong thư viện <algorithm> của C++ để sắp xếp các ký tự trong chuỗi.
      - Hàm sort sẽ sắp xếp các ký tự theo thứ tự tăng dần dựa trên mã ASCII.
    - Xuất chuỗi
      - In chuỗi đã được sắp xếp ra màn hình.
- Testcase kiểm tra:

Testcase	Input	Kết quả mong đợi
1	Moi ban nhap chuoi s: abcba	Chuoi sau khi sap xep: aabbc
2	Moi ban nhap chuoi s: zxy@12	Chuoi sau khi sap xep: 12@xyz

- Mã nguồn

```
#include <iostream>
#include <algorithm>
#include <cstring>
using namespace std;

void SapXepChuoi(char s[]) {
    // Lấy độ dài của chuỗi
    int n = strlen(s);

    // Sắp xếp các ký tự trong chuỗi
```

```
    sort(s, s + n);
}

int main() {
    char s[1001];

    // Nhập chuỗi từ bàn phím
    cout << "Moi ban nhap chuoi s: ";
    cin.getline(s, 1001);

    // Gọi hàm sắp xếp chuỗi
    SapXepChuoi(s);

    // Xuất chuỗi đã sắp xếp
    cout << "Chuoi sau khi sap xep: " << s << endl;

    return 0;
}
```

Bài tập 2.
Cho chuỗi s (| s | ≤ 1.000) và số nguyên k (0 ≤ k ≤ | s | - 1). Hãy viết chương trình xóa ký tự tại vị trí thứ k trong chuỗi.

Moi ban nhap chuoi s: abcba

Moi ban nhap vi tri can xoa: 1

Chuoi “abcba” sau khi xoa ky tu tai vi tri 1: acba

- Ý tưởng
    - Cấu trúc dữ liệu:
        - Bài toán yêu cầu xóa một ký tự tại vị trí k trong chuỗi, với 0 ≤ k ≤ |s| - 1.
        - Kết quả là chuỗi sau khi xóa ký tự tại vị trí chỉ định.
    - Chức năng chính:
        - Nhập chuỗi và chỉ số k
            - Nhập chuỗi s từ bàn phím.
            - Nhập vị trí k cần xóa (đảm bảo k nằm trong khoảng hợp lệ).
        - Xử lý xóa ký tự tại vị trí k
            - Duyệt qua chuỗi và tạo một chuỗi mới, bỏ qua ký tự tại vị trí k.
            - Có thể sử dụng hàm erase của C++ để xóa trực tiếp ký tự tại vị trí k.
        - Xuất chuỗi kết quả
            - In chuỗi sau khi xóa ký tự tại vị trí k.

- Testcase kiểm tra:

Testcase	Input	Kết quả mong đợi
1	Moi ban nhap chuoi s: abcba Moi ban nhap vi tri can xoa: 1	Chuoi sau khi xoa ky tu tai vi tri 1: acba
2	Moi ban nhap chuoi s: abcba Moi ban nhap vi tri can xoa: 5	Vi tri xoa khong hop le! Chuoi sau khi xoa ky tu tai vi tri 5: abcba

- Mã nguồn

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

void XoaKyTu(char s[], int k) {
    int n = strlen(s);

    // Kiểm tra vị trí k có hợp lệ không
    if (k < 0 || k >= n) {
        cout << "Vi tri xoa khong hop le!" << endl;
        return;
    }

    // Dịch các ký tự từ vị trí k+1 về trước
    for (int i = k; i < n - 1; i++) {
        s[i] = s[i + 1];
    }

    // Đặt ký tự cuối thành '\0' để kết thúc chuỗi
    s[n - 1] = '\0';
}

int main() {
    char s[1001];
    int k;

    // Nhập chuỗi
    cout << "Moi ban nhap chuoi s: ";
    cin.getline(s, 1001);

    // Nhập vị trí cần xóa
    cout << "Moi ban nhap vi tri can xoa: ";
    cin >> k;

    // Gọi hàm xóa ký tự tại vị trí k
```

```
    XoaKyTu(s, k);

    // Xuất chuỗi kết quả
    if (k >= 0 && k < strlen(s) + 1) {
        cout << "Chuoi sau khi xoa ky tu tai vi tri " << k << ": " << s << endl;
    }

    return 0;
}
```

Bài tập 3.
Cho chuỗi s (| s | ≤ 1.000), số nguyên k (0 ≤ k ≤ | s | - 1) và ký tự c. Viết hàm chèn ký tự c vào chuỗi s tại vị trí k.

Moi ban nhap chuoi s: abcba

Moi ban nhap vi tri can chen: 1

Moi ban nhap ky tu chen: d

Chuoi "abcba" sau khi them ky tu "d" vao vi tri 1: adbcba

- Ý tưởng
  - Cấu trúc dữ liệu:
    - Bài toán yêu cầu chèn một ký tự c vào vị trí k trong chuỗi s.
    - Sau khi thực hiện, in ra chuỗi kết quả.
  - Chức năng chính:
    - Nhập chuỗi và các thông tin cần thiết
      - Nhập chuỗi s.
      - Nhập vị trí cần chèn k (đảm bảo $0 \leq k \leq |s|$).
      - Nhập ký tự cần chèn c.
    - Xử lý chèn ký tự
      - Dịch các ký tự từ vị trí k trở về sau lùi một vị trí.
      - Gán ký tự c vào vị trí k.
    - Xuất chuỗi kết quả
      - In chuỗi sau khi chèn ký tự.
- Testcase kiểm tra:

Testcase	Input	Kết quả mong đợi
1	Moi ban nhap chuoi s: abcba Moi ban nhap vi tri can chen: 1 Moi ban nhap ky tu chen: d	Chuoi sau khi them ky tu "d" vao vi tri 1: adbcba
2	Moi ban nhap chuoi s: abcba	Vi tri chen khong hop le!

	Moi ban nhap vi tri can chen: 6 Moi ban nhap ky tu chen: d	

- Mã nguồn

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

void ChenKyTu(char s[], int k, char c) {
    int n = strlen(s);

    // Kiểm tra vị trí k có hợp lệ không
    if (k < 0 || k > n) {
        cout << "Vi tri chen khong hop le!" << endl;
        return;
    }

    // Dịch các ký tự từ vị trí k về sau lùi một ô
    for (int i = n; i >= k; i--) {
        s[i + 1] = s[i];
    }

    // Gán ký tự c vào vị trí k
    s[k] = c;

    // Thêm ký tự kết thúc chuỗi
    s[n + 1] = '\0';
}

int main() {
    char s[1001], c;
    int k;

    // Nhập chuỗi
    cout << "Moi ban nhap chuoi s: ";
    cin.getline(s, 1001);

    // Nhập vị trí cần chèn
    cout << "Moi ban nhap vi tri can chen: ";
    cin >> k;

    // Nhập ký tự cần chèn
    cout << "Moi ban nhap ky tu chen: ";
    cin >> c;

    // Gọi hàm chèn ký tự
    ChenKyTu(s, k, c);
```

```
    // Xuất chuỗi kết quả nếu vị trí hợp lệ
    if (k >= 0 && k <= strlen(s)) {
        cout << "Chuoi sau khi them ky tu \"" << c << "\" vao vi tri " << k << ": "
<< s << endl;
    }

    return 0;
}
```

File văn bản

Bài tập 1.
Doc các số nguyên Tên file chương trình: DocCacSoNguyen.*

Viết hàm đọc file chứa n số nguyên. Xuất kết quả đọc được ra màn hình.

Dữ liệu vào: Từ file văn bản DaySoNguyen.inp gồm

- Dòng thứ 1: Chứa số nguyên n là số phần tử của mảng.
- Dòng thứ 2: Chứa n số nguyên (các số cách nhau ít nhất một khoảng trắng).

Ví dụ: file DaySoNguyen.inp

5

4 3 5 3 2

- Ý tưởng
    - Bài toán yêu cầu:
        - Đọc dữ liệu từ file văn bản DaySoNguyen.inp.
            - Dòng đầu tiên chứa số nguyên n (số phần tử của mảng).
            - Dòng thứ hai chứa n số nguyên (cách nhau bởi khoảng trắng).
        - In ra các số đã đọc từ file.
    - Chức năng chính:
        - Đọc file
            - Mở file văn bản DaySoNguyen.inp bằng cách sử dụng fopen để mở file với chế độ "rt" (read, text).
            - Đọc số nguyên đầu tiên (số lượng phần tử n).
            - Duyệt n lần để đọc các số nguyên còn lại và lưu vào mảng a[].
            - Đóng file bằng fclose(fi) sau khi đọc xong.
        - Hàm XuatDaySoNguyen
            - Xuất dãy số từ mảng a[] ra màn hình.
        - Hàm Main
            - Khai báo mảng a[] và số lượng phần tử n.
            - Gọi hàm DocDaySoNguyen để đọc file.
            - Nếu file đọc thành công, xuất các số bằng hàm XuatDaySoNguyen.
- Testcase kiểm tra:

Testcase	Input	Kết quả mong đợi
1	DaySoNguyen.inp 5 4 3 5 3 2	Cac so trong file la: 4 3 5 3 2

2	DaySoNguyen1.inp 5 4 3 5 3 2	Khong the doc file

- Mã nguồn

```c
#include <stdio.h>
#define FI "DaySoNguyen.inp"

void DocDaySoNguyen(int a[], int *n) {
    FILE *fi;
    fi = fopen(FI, "rt"); // Mở file đọc

    if (fi == NULL) { // Kiểm tra lỗi mở file
        printf("Khong the doc file\n");
        return;
    }

    // Đọc số lượng phần tử
    fscanf(fi, "%d", n);

    // Đọc các số nguyên vào mảng
    for (int i = 0; i < *n; i++) {
        fscanf(fi, "%d", &a[i]);
    }

    fclose(fi); // Đóng file
}

void XuatDaySoNguyen(int a[], int n) {
    printf("Cac so trong file la: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", a[i]);
    }
    printf("\n");
}

int main() {
    int a[100]; // Mảng lưu các số nguyên
    int n;      // Số lượng phần tử

    DocDaySoNguyen(a, &n); // Gọi hàm đọc file

    if (n > 0) { // Nếu n hợp lệ thì xuất dãy số
        XuatDaySoNguyen(a, n);
    }
```

```
    return 0;
}
```

Bài tập 2.
Đọc mảng 2 chiều Tên file chương trình: DocMang2CSoNguyen.*

Viết hàm đọc file chứa mảng 2 chiều các số nguyên. Xuất kết quả đọc được ra màn hình.

Dữ liệu vào: Từ file văn bản MangSo.inp gồm

- Dòng thứ 1: Chứa 2 số nguyên n, m là số dòng và số cột của mảng.
- n dòng tiếp theo: Mỗi dòng chứa n số nguyên.
    - (các số cách nhau ít nhất một khoảng trắng)

Ví dụ: file MangSo.inp

3 5

5 3 5 7 2

4 5 5 7 1

5 4 5 3 9

- Ý tưởng
    - Đầu vào:
        - File MangSo.inp chứa:
            - Dòng đầu tiên: Hai số nguyên n (số dòng) và m (số cột) của mảng.
            - n dòng tiếp theo: Mỗi dòng chứa m số nguyên cách nhau bởi khoảng trắng.
    - Xử lý:
        - Mở file để đọc dữ liệu.
        - Đọc số dòng n và số cột m.
        - Sử dụng vòng lặp để đọc từng phần tử vào mảng hai chiều.
        - Đóng file sau khi đọc xong.
    - Đầu ra:
        - Xuất mảng hai chiều ra màn hình dưới dạng bảng.
- Testcase kiểm tra:

Testcase	Input	Kết quả mong đợi
1	MangSo.inp 3 5 5 3 5 7 2 4 5 5 7 1	Mang 2 chieu doc duoc: 5 3 5 7 2 4 5 5 7 1 5 4 5 3 9

	5 4 5 3 9	
2	MangSo1.inp 3 5 5 3 5 7 2 4 5 5 7 1 5 4 5 3 9	Khong the doc file

- Mã nguồn

```c
#include <stdio.h>
#define FI "MangSo.inp"

void DocMang2Chieu(int a[][100], int *n, int *m) {
    FILE *fi;
    fi = fopen(FI, "rt"); // Mở file đọc

    if (fi == NULL) { // Kiểm tra lỗi mở file
        printf("Khong the doc file\n");
        return;
    }

    // Đọc số dòng và số cột
    fscanf(fi, "%d %d", n, m);

    // Đọc từng phần tử vào mảng
    for (int i = 0; i < *n; i++) {
        for (int j = 0; j < *m; j++) {
            fscanf(fi, "%d", &a[i][j]);
        }
    }

    fclose(fi); // Đóng file
}

void XuatMang2Chieu(int a[][100], int n, int m) {
    printf("Mang 2 chieu doc duoc:\n");
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            printf("%5d", a[i][j]);
        }
        printf("\n");
    }
}

int main() {
    int a[100][100]; // Mảng lưu các số nguyên
```

```
    int n, m;         // Số dòng và số cột

    DocMang2Chieu(a, &n, &m); // Gọi hàm đọc file

    if (n > 0 && m > 0) { // Nếu đọc thành công, xuất mảng
        XuatMang2Chieu(a, n, m);
    }

    return 0;
}
```

Bài tập 3.
Tìm số nguyên tố Tên file chương trình: NguyenTo.*

Cho dãy n số nguyên $a_0, a_1, \ldots, a_{n-1}$. Hãy liệt kê các số nguyên tố trong dãy theo thứ tự tăng dần.

Dữ liệu vào: Từ file văn bản NT.INP gồm có

- Dòng đầu là n ($1 < n \leq 10.000$)
- n dòng tiếp theo, dòng thứ i ghi số a_i

Kết quả: Ghi ra file văn bản NT.OUT

- Dòng đầu ghi số m là số lượng số nguyên tố (Nếu không có số nguyên tố nào thì ghi số 0)
- Dòng thứ 2 ghi m số nguyen tố tìm được (các số cách nhau ít nhất một khoảng trắng)

Ví dụ:

NT.INP
6
7
5
6
1
19
4

NT.OUT
3
5 7 19

- Ý tưởng
  - Đầu vào:
    - File NT.INP chứa:
      - Dòng đầu: Số nguyên n là số lượng phần tử trong dãy.
      - n dòng tiếp theo: Các số nguyên a0, a1, ..., an-1.
  - Xử lý:
    - Đọc dữ liệu từ file NT.INP.

    - Kiểm tra từng số trong dãy có phải số nguyên tố hay không.
        - Số nguyên tố là số lớn hơn 1, chỉ chia hết cho 1 và chính nó.
    - Lưu các số nguyên tố vào danh sách và sắp xếp chúng theo thứ tự tăng dần.
- Đầu ra:
    - Ghi vào file NT.OUT:
        - Dòng đầu: Số lượng số nguyên tố tìm được.
        - Dòng thứ hai: Các số nguyên tố, cách nhau bởi khoảng trắng.

- Testcase kiểm tra:

Testcase	Input	Kết quả mong đợi
1	NT.INP 6 7 5 6 1 19 4	Xu ly xong! Ket qua da duoc ghi vao file NT.OUT NT.OUT 3 5 7 19
2	NT1.INP 6 7 5 6 1 19 4	Khong the doc file NT.INP

- Mã nguồn

```
#include <stdio.h>
#include <math.h>
#include <vector>
#include <algorithm>
#define INPUT_FILE "NT.INP"
#define OUTPUT_FILE "NT.OUT"

// Hàm kiểm tra số nguyên tố
bool KiemTraNguyenTo(int x) {
    if (x <= 1) return false; // Không phải số nguyên tố
    for (int i = 2; i <= sqrt(x); i++) {
        if (x % i == 0) return false; // Không phải số nguyên tố
    }
```

```
    return true; // Là số nguyên tố
}

void DocDuLieuVaTimNguyenTo() {
    FILE *fi = fopen(INPUT_FILE, "rt");
    if (fi == NULL) {
        printf("Khong the doc file %s\n", INPUT_FILE);
        return;
    }

    // Đọc số lượng phần tử
    int n;
    fscanf(fi, "%d", &n);

    // Đọc các phần tử và kiểm tra số nguyên tố
    std::vector<int> nguyenTo;
    for (int i = 0; i < n; i++) {
        int x;
        fscanf(fi, "%d", &x);
        if (KiemTraNguyenTo(x)) {
            nguyenTo.push_back(x);
        }
    }
    fclose(fi);

    // Sắp xếp danh sách số nguyên tố tăng dần
    std::sort(nguyenTo.begin(), nguyenTo.end());

    // Ghi kết quả vào file
    FILE *fo = fopen(OUTPUT_FILE, "wt");
    if (fo == NULL) {
        printf("Khong the ghi file %s\n", OUTPUT_FILE);
        return;
    }

    fprintf(fo, "%lu\n", nguyenTo.size()); // Ghi số lượng số nguyên tố
    for (int i = 0; i < nguyenTo.size(); i++) {
        fprintf(fo, "%d ", nguyenTo[i]); // Ghi các số nguyên tố
    }
    fclose(fo);
}

int main() {
    DocDuLieuVaTimNguyenTo();
    printf("Xu ly xong! Ket qua da duoc ghi vao file NT.OUT\n");
    return 0;
}
```

Bài tập 4.

Tìm cặp số Tên file chương trình: CapSo.*

Cho dãy n số nguyên $a_0, a_1, \ldots, a_{n-1}$ và số nguyên k. Hãy tìm các cặp (a_i , a_j) ($0 \leq i < j \leq n-1$) sao cho $a_i + a_j = k$.

Dữ liệu vào: Từ file văn bản CapSo.INP gồm có

- Dòng đầu chứa số n ($1 < n \leq 10.000$) và k
- n dòng tiếp theo, dòng thứ i ghi số a_i

Kết quả: Ghi ra file văn bản CapSo.OUT

- Nếu không tìm được cặp số nào ghi số 0. Ngược lại ghi các cặp số tìm được, mỗi cặp nằm trên 1 dòng (Các số cách nhau ít nhất 1 khoảng trắng).

Ví dụ:

CapSo.INP
6 5 2 5 3 1 4 1

CapSo.OUT
2 3 1 4 4 1

- Ý tưởng
  - Đầu vào:
    - File CapSo.INP:
      - Dòng đầu chứa hai số nguyên: n (số lượng phần tử trong dãy) và k (tổng cần tìm).
      - n dòng tiếp theo: Các số nguyên $a_0, a_1, \ldots, a_{n-1}$.
  - Xử lý:
    - Đọc dữ liệu từ file CapSo.INP.
    - Duyệt qua tất cả các cặp chỉ số (i, j) với i < j trong dãy số.
    - Kiểm tra điều kiện: $a_i + a_j == k$.
      - Nếu đúng, lưu cặp (ai, aj) vào danh sách kết quả.
    - Nếu không tìm được cặp số nào, ghi số 0.
  - Đầu ra:
    - Ghi vào file CapSo.OUT:
      - Nếu không tìm được cặp số nào, ghi 0.
      - Ngược lại, ghi từng cặp (ai, aj) trên mỗi dòng.

- Testcase kiểm tra:

Testcase	Input	Kết quả mong đợi
1	CapSo.INP 6 5 2 5 3 1 4 1	Xu ly xong! Ket qua da duoc ghi vao file CapSo.OUT CapSo.OUT 2 3 1 4 4 1
2	CapSo1.INP 6 5 2 5 3 1 4 1	Khong the doc file CapSo.INP

- Mã nguồn

```
#include <stdio.h>
#include <vector>
#define INPUT_FILE "CapSo.INP"
#define OUTPUT_FILE "CapSo.OUT"

// Hàm xử lý tìm các cặp số
void TimCapSo() {
    FILE *fi = fopen(INPUT_FILE, "rt");
    if (fi == NULL) {
        printf("Khong the doc file %s\n", INPUT_FILE);
        return;
    }

    // Đọc số lượng phần tử n và tổng cần tìm k
    int n, k;
    fscanf(fi, "%d %d", &n, &k);

    // Đọc mảng các số nguyên
    std::vector<int> arr(n);
    for (int i = 0; i < n; i++) {
        fscanf(fi, "%d", &arr[i]);
    }
```

```
    fclose(fi);

    // Tìm các cặp thỏa mãn
    std::vector<std::pair<int, int>> result;
    for (int i = 0; i < n; i++) {
        for (int j = i + 1; j < n; j++) {
            if (arr[i] + arr[j] == k) {
                result.push_back({arr[i], arr[j]});
            }
        }
    }

    // Ghi kết quả ra file
    FILE *fo = fopen(OUTPUT_FILE, "wt");
    if (fo == NULL) {
        printf("Khong the ghi file %s\n", OUTPUT_FILE);
        return;
    }

    if (result.empty()) {
        fprintf(fo, "0\n");
    } else {
        for (const auto &p : result) {
            fprintf(fo, "%d %d\n", p.first, p.second);
        }
    }
    fclose(fo);
}

int main() {
    TimCapSo();
    printf("Xu ly xong! Ket qua da duoc ghi vao file %s\n", OUTPUT_FILE);
    return 0;
}
```