**Lab1b\_Bài tập minh họa\_Câu 2**

Xác định Input và Output

Input: Một số nguyên dương n (người dùng nhập từ bàn phím).

Output: Thông báo n có phải là số chính phương hay không.

Cấu trúc thuật toán

1. Nhập số nguyên dương n từ bàn phím.
2. Kiểm tra điều kiện hợp lệ: Nếu n ≤ 0, yêu cầu nhập lại.
3. Xác định số chính phương:

Tính căn bậc hai của n: canBacHai = sqrt(n).

Nếu canBacHai \* canBacHai == n thì n là số chính phương, ngược lại không phải.

1. Xuất kết quả: In ra màn hình kết quả kiểm tra.

**Lab1b\_Bài tập minh họa\_Câu 4**

Xác định Input và Output

Input:

Một số nguyên n (số lượng phần tử của mảng).

n số nguyên tạo thành mảng a.

Output:

Xuất mảng các số chẵn tìm được.

Kiểm tra và in kết quả về các tính chất của mảng:

Dãy có đan dấu không?

Dãy có đơn điệu không? (Tăng hoặc giảm dần)

Dãy có đối xứng không?

Cấu trúc thuật toán

1. Nhập mảng số nguyên a gồm n phần tử.
2. Tách các số chẵn từ a sang mảng mới aChan.
3. Xuất mảng số chẵn vừa tìm được.
4. Kiểm tra tính chất của mảng a:
   * Đan dấu: Duyệt qua các cặp số liên tiếp, nếu có cặp cùng dấu → Không đan dấu.
   * Đơn điệu: Kiểm tra dãy có tăng dần hoặc giảm dần không.
   * Đối xứng: So sánh phần tử đầu và cuối, tiếp tục so sánh các cặp tiếp theo cho đến giữa mảng.
5. Xuất kết quả kiểm tra tính chất của mảng.
6. Giải phóng bộ nhớ cấp phát động.

**Lab1b\_Bài tập minh họa\_Câu 5**

Xác định Input và Output

Input:

Một số nguyên n (số lượng phần tử của mảng).

n số nguyên tạo thành mảng a.

Output:

Xuất mảng các số chẵn tìm được.

Xuất mảng các số nguyên tố tìm được.

Kiểm tra và in kết quả về các tính chất của mảng:

Dãy có đan dấu không?

Dãy có đơn điệu không? (Tăng hoặc giảm dần)

Dãy có đối xứng không?

Cấu trúc thuật toán

1. Nhập mảng số nguyên a gồm n phần tử.
2. Tách các số chẵn từ a sang mảng mới aChan.
3. Tách các số nguyên tố từ a sang mảng mới aNguyenTo.
4. Xuất hai mảng con aChan và aNguyenTo.
5. Kiểm tra tính chất của mảng a:
   * Đan dấu: Duyệt qua các cặp số liên tiếp, nếu có cặp cùng dấu → Không đan dấu.
   * Đơn điệu: Kiểm tra dãy có tăng dần hoặc giảm dần không.
   * Đối xứng: So sánh phần tử đầu và cuối, tiếp tục so sánh các cặp tiếp theo cho đến giữa mảng.
6. Xuất kết quả kiểm tra tính chất của mảng.
7. Giải phóng bộ nhớ cấp phát động.

**Lab1b\_Bài tập phải làm\_Câu 1**

Xác định Input và Output

Input:

Ba số thực a, b, c (hệ số của phương trình bậc hai).

Output:

Nghiệm của phương trình bậc hai ax² + bx + c = 0:

Vô nghiệm nếu a == 0 và b == 0.

Một nghiệm duy nhất nếu a == 0 và b ≠ 0.

Nghiệm kép nếu delta = 0.

Hai nghiệm phân biệt nếu delta > 0.

Vô nghiệm nếu delta < 0.

Cấu trúc thuật toán

1. Nhập hệ số a, b, c.
2. Kiểm tra trường hợp đặc biệt:

Nếu a == 0:

Nếu b == 0, phương trình vô nghiệm.

Ngược lại, phương trình có một nghiệm x = -c/b.

1. Tính delta:

delta = b² - 4ac

Nếu delta < 0: phương trình vô nghiệm.

Nếu delta == 0: phương trình có nghiệm kép x = -b / (2a).

Nếu delta > 0: phương trình có hai nghiệm x1 = (-b + sqrt(delta)) / (2a) và x2 = (-b - sqrt(delta)) / (2a).

1. Xuất kết quả nghiệm của phương trình.

**Lab1b\_Bài tập phải làm\_Câu 2**

Xác định Input và Output

Input:

n: số nguyên dương, là số lượng phần tử của mảng (0 < n ≤ 100).

Mảng a gồm n số nguyên.

Output:

Thông báo liệu mảng có tính chẵn lẻ xen kẽ hay không.

Thông báo liệu mảng có toàn số chẵn hay không.

Cấu trúc thuật toán

1. Nhập số lượng phần tử n, kiểm tra n hợp lệ (0 < n ≤ 100).
2. Nhập các phần tử của mảng a.
3. Kiểm tra tính chẵn lẻ xen kẽ:

Duyệt từ phần tử thứ hai (i = 1) đến phần tử cuối cùng (i = n-1).

Nếu hai phần tử liên tiếp có cùng tính chẵn/lẻ (a[i] % 2 == a[i - 1] 2), kết luận mảng không xen kẽ, trả về false.

Nếu không có cặp nào vi phạm, mảng có tính chẵn lẻ xen kẽ, trả về true.

1. Kiểm tra tính toàn chẵn:

Duyệt toàn bộ mảng a.

Nếu có bất kỳ phần tử nào lẻ (a[i] % 2 != 0), kết luận mảng không toàn số chẵn, trả về false.

Nếu tất cả đều chẵn, kết luận mảng toàn số chẵn, trả về true.

1. Xuất kết quả:

Nếu mảng có tính chẵn lẻ xen kẽ, in "Mảng có tính chẵn lẻ xen kẽ.", ngược lại in "Mảng không có tính chẵn lẻ xen kẽ."

Nếu mảng toàn số chẵn, in "Mảng chứa toàn số chẵn.", ngược lại in "Mảng không chứa toàn số chẵn."

**Lab1b\_Bài tập phải làm\_Câu 3**

Xác định Input và Output

Input:

1. Mảng số nguyên:

n: số lượng phần tử (0 < n ≤ 100).

Mảng a gồm n số nguyên.

1. Mảng ký tự:

n: số lượng ký tự (0 < n ≤ 100).

Mảng c gồm n ký tự.

Output:

1. Kết quả kiểm tra mảng số nguyên:

Mảng có tính chẵn lẻ xen kẽ hay không.

Mảng có toàn số chẵn hay không.

1. Mảng nguyên âm từ mảng ký tự c.

Tóm tắt thuật toán

1. Xử lý mảng số nguyên:

Nhập số lượng phần tử n và kiểm tra hợp lệ (0 < n ≤ 100).

Nhập n phần tử vào mảng a.

Kiểm tra tính chẵn lẻ xen kẽ bằng cách duyệt mảng a, nếu có hai phần tử liên tiếp cùng tính chẵn/lẻ thì trả về false, ngược lại trả về true.

Kiểm tra tính toàn chẵn, nếu có phần tử lẻ thì trả về false, ngược lại trả về true.

Xuất kết quả.

1. Xử lý mảng ký tự:

Nhập số lượng ký tự n và kiểm tra hợp lệ.

Nhập n ký tự vào mảng c.

Lọc nguyên âm bằng cách duyệt từng ký tự trong c, nếu là nguyên âm (a, e, i, o, u không phân biệt hoa thường) thì đưa vào mảng b.

Xuất mảng nguyên âm.

**Lab1b\_Bài tập phải làm\_Câu 4**

Xác định Input và Output

Input:

1. Mảng số nguyên a:

n: số lượng phần tử (0 < n ≤ 100).

Mảng a gồm n số nguyên.

1. Mảng ký tự c:

n: số lượng ký tự (0 < n ≤ 100).

Mảng c gồm n ký tự.

1. Hai mảng tăng dần x và y để trộn:

px: số lượng phần tử của mảng x.

Mảng x gồm px số nguyên theo thứ tự tăng dần.

py: số lượng phần tử của mảng y.

Mảng y gồm py số nguyên theo thứ tự tăng dần.

Output:

1. Kết quả kiểm tra mảng số nguyên a:

Mảng có tính chẵn lẻ xen kẽ hay không.

Mảng có toàn số chẵn hay không.

1. Mảng nguyên âm b được lọc từ c.
2. Mảng z là kết quả của việc trộn hai mảng x và y theo thứ tự tăng dần.

Cấu trúc thuật toán

1. Xử lý mảng số nguyên a:

Nhập số lượng phần tử n, kiểm tra hợp lệ (0 < n ≤ 100).

Nhập n phần tử vào mảng a.

Kiểm tra tính chẵn lẻ xen kẽ bằng cách duyệt mảng a, nếu có hai phần tử liên tiếp cùng tính chẵn/lẻ thì trả về false, ngược lại trả về true.

Kiểm tra tính toàn chẵn, nếu có phần tử lẻ thì trả về false, ngược lại trả về true.

Xuất kết quả.

1. Xử lý mảng ký tự c:

Nhập số lượng ký tự n, kiểm tra hợp lệ.

Nhập n ký tự vào mảng c.

Lọc nguyên âm bằng cách duyệt từng ký tự trong c, nếu là nguyên âm (a, e, i, o, u không phân biệt hoa thường) thì đưa vào mảng b.

Xuất mảng nguyên âm b.

1. Trộn hai mảng tăng dần x và y:

Nhập số lượng phần tử px, py của hai mảng x và y.

Nhập px số nguyên vào x theo thứ tự tăng dần.

Nhập py số nguyên vào y theo thứ tự tăng dần.

Gộp hai mảng tăng dần bằng cách so sánh từng phần tử, chèn vào mảng z theo đúng thứ tự tăng dần.

Xuất mảng z sau khi trộn.

**Lab1b\_Bài tập phải làm\_Câu 5**

**Xác định Input và Output**

Input:

Mảng số nguyên a:

n: Số lượng phần tử của mảng (0 < n ≤ 100).

a[n]: Mảng chứa n số nguyên.

Output:

Mảng số chẵn b (chứa các số chẵn từ a).

Mảng số lẻ c (chứa các số lẻ từ a).

Cấu trúc thuật toán

1. Nhập mảng số nguyên a:

Nhập n (kiểm tra hợp lệ 0 < n ≤ 100).

Nhập n phần tử vào mảng a.

1. Tách mảng a thành hai mảng b và c:

Duyệt từng phần tử trong a:

Nếu phần tử chẵn, thêm vào b.

Nếu phần tử lẻ, thêm vào c.

Cập nhật số lượng phần tử m (của b) và p (của c).

3.Xuất kết quả:

Xuất mảng số chẵn b.

Xuất mảng số lẻ c.

**Lab1c\_Bài tập minh họa\_Câu 3**

Xác định Input và Output

Input:

Danh sách lớp học (LopHoc) gồm:

soLuong: Số lượng học sinh (0 < soLuong ≤ 50).

Mỗi học sinh (HocSinh) có:

maSo: Mã số học sinh (chuỗi 10 ký tự).

ho: Họ học sinh (chuỗi 10 ký tự).

ten: Tên học sinh (chuỗi 50 ký tự).

phai: Giới tính (0 - Nữ, 1 - Nam).

namSinh: Năm sinh (số nguyên).

Output:

Sĩ số học sinh nam và nữ.

Danh sách học sinh sau khi sắp xếp theo năm sinh tăng dần, gồm:

Mã số

Họ

Tên

Giới tính

Năm sinh

Cấu trúc thuật toán

Nhập danh sách học sinh (NhapLopHoc)

Nhập soLuong (kiểm tra hợp lệ 0 < soLuong ≤ 50).

Nhập thông tin cho từng học sinh: maSo, ho, ten, phai, namSinh.

Đếm sĩ số nam và nữ (DemSiSo)

Khởi tạo soNam = 0, soNu = 0.

Duyệt danh sách, đếm số lượng học sinh nam (phai == 1) và nữ (phai == 0).

Xuất kết quả.

Sắp xếp danh sách theo năm sinh tăng dần (SapXepTangTheoNamSinh)

Dùng thuật toán sắp xếp chọn (Selection Sort) để sắp xếp mảng aHocSinh theo namSinh.

Xuất danh sách học sinh đã sắp xếp (XuatLopHoc)

Duyệt danh sách và in ra thông tin từng học sinh theo thứ tự đã sắp xếp.

**Lab1c\_Bài tập phải làm\_Câu 1**

Xác định Input và Output

Input:

Nhập danh sách nhân viên của phòng ban với các thông tin sau:

Số lượng nhân viên (0 < soLuong ≤ 100).

Mỗi nhân viên có các thông tin:

Mã số (maSo): chuỗi tối đa 10 ký tự.

Họ (ho): chuỗi tối đa 10 ký tự.

Tên (ten): chuỗi tối đa 50 ký tự.

Giới tính (phai): 0 là nữ, 1 là nam.

Thâm niên (thamNien): số năm làm việc, không âm.

Output:

Xuất danh sách nhân viên gồm: mã số, họ, tên, giới tính và thâm niên.

Số lượng nhân viên nam và nữ trong danh sách.

Danh sách nhân viên sau khi sắp xếp theo thâm niên tăng dần.

Cấu trúc thuật toán

1.Nhập danh sách nhân viên (NhapPhongBan)

Nhập số lượng nhân viên và kiểm tra hợp lệ.

Nhập thông tin từng nhân viên: mã số, họ, tên, giới tính, thâm niên.

Kiểm tra điều kiện hợp lệ của giới tính và thâm niên.

2.Xuất danh sách nhân viên (XuatPhongBan)

Duyệt danh sách và in ra thông tin từng nhân viên.

3.Đếm số lượng nhân viên nam và nữ (DemSiSo)

Khởi tạo biến đếm số nhân viên nam và nữ.

Duyệt danh sách, tăng biến đếm tương ứng theo giới tính.

Xuất kết quả.

4.Sắp xếp danh sách theo thâm niên tăng dần (SapXepTangTheoThamNien)

Sử dụng thuật toán sắp xếp chọn (Selection Sort) để sắp xếp danh sách theo thâm niên.

5.Xuất danh sách nhân viên sau khi sắp xếp

Gọi lại hàm xuất danh sách để hiển thị kết quả.

**Lab1c\_Bài tập phải làm\_Câu 2**

Xác định Input và Output

Input:

Nhập danh sách các loại hoa với các thông tin:

Số lượng loại hoa (0 < SoLuong ≤ 20).

Mỗi loại hoa có các thông tin:

Tên loại (Ten): chuỗi tối đa 50 ký tự.

Số lượng (SoLuong): số nguyên không âm.

Đơn vị tính (DVT): chuỗi tối đa 10 ký tự.

Đơn giá (DonGia): số thực dương.

Nhập thông tin bán hoa:

Tên loại hoa cần mua (tenloai).

Số lượng cần mua (soluong).

Output:

Xuất danh sách các loại hoa trước khi bán, bao gồm: tên loại, số lượng, đơn vị tính, đơn giá.

Nếu tìm thấy loại hoa cần mua:

Kiểm tra số lượng có đủ để bán không.

Nếu đủ, thực hiện giao dịch và in tổng tiền.

Nếu không đủ, thông báo "Không đủ số lượng để bán".

Nếu không tìm thấy loại hoa, thông báo "Không tìm thấy loại hoa trong danh sách".

Xuất danh sách loại hoa sau khi bán.

Cấu trúc thuật toán

1.Nhập danh sách loại hoa (NhapDanhSach)

Nhập số lượng loại hoa và kiểm tra hợp lệ.

Nhập thông tin từng loại hoa: tên, số lượng, đơn vị tính, đơn giá.

2.Xuất danh sách loại hoa (XuatDanhSach)

Duyệt danh sách và in ra thông tin từng loại hoa.

3.Tìm loại hoa theo tên (TimLoaiHoa)

Duyệt danh sách và so sánh tên từng loại hoa với tên nhập vào.

Nếu tìm thấy, trả về vị trí (index) trong danh sách.

Nếu không tìm thấy, trả về -1.

4.Xử lý bán hoa (XuLyBanHoa)

Tìm vị trí loại hoa cần mua bằng hàm TimLoaiHoa.

Nếu không tìm thấy, in thông báo lỗi.

Nếu số lượng cần mua lớn hơn số lượng có sẵn, in thông báo "Không đủ số lượng để bán".

Nếu số lượng đủ:

Tính tổng tiền = soluong \* DonGia.

Giảm số lượng của loại hoa sau khi bán.

In thông báo bán thành công và số tiền phải trả.

5.Xuất danh sách loại hoa sau khi bán (XuatDanhSach)

Hiển thị danh sách sau khi cập nhật số lượng.

**Lab1d\_Bài tập minh họa\_Câu 2**

Xác định Input và Output

Input:

Một chuỗi ký tự (s) do người dùng nhập vào.

Output:

Nếu chuỗi đối xứng, in: "Chuoi doi xung."

Nếu chuỗi không đối xứng, in: "Chuoi khong doi xung."

Cấu trúc thuật toán

1. Nhập chuỗi từ người dùng (getline(cin, s)).
2. Kiểm tra chuỗi có đối xứng hay không (KiemTraDoiXung):

Duyệt từng ký tự từ đầu đến giữa chuỗi.

So sánh ký tự hiện tại với ký tự tương ứng từ cuối về.

Nếu có sự khác biệt, trả về false.

Nếu không có sự khác biệt, trả về true.

1. Xuất kết quả kiểm tra:

Nếu hàm KiemTraDoiXung trả về true, in "Chuoi doi xung."

Ngược lại, in "Chuoi khong doi xung."

**Lab1d\_Bài tập minh họa\_Câu 3**

Xác định Input và Output

Input:

* Một chuỗi ký tự (s) có độ dài tối đa 1000 ký tự, do người dùng nhập vào.

Outpu:

Số lượng từ trong chuỗi (soTu), in ra dưới dạng:

So tu trong chuoi: <số từ>

Cấu trúc thuật toán

1. Nhập chuỗi từ người dùng (cin.getline(s, 1001)).
2. Đếm số từ trong chuỗi (DemSoTu(s)):

Khởi tạo biến count = 0 để đếm số từ.

Sử dụng biến inWord = false để theo dõi trạng thái đang ở trong một từ hay không.

Duyệt từng ký tự trong chuỗi:

Nếu ký tự không phải khoảng trắng (' ', '\t', '\n') và trước đó không phải trong một từ, tăng count và đặt inWord = true.

Nếu gặp khoảng trắng, đặt inWord = false để đánh dấu kết thúc từ.

1. Xuất số từ đếm được (cout << "So tu trong chuoi: " << soTu << endl;).

**Lab1d\_Bài tập phải làm\_Câu 1**

Xác định Input và Output

Input:

Một chuỗi ký tự (s) có độ dài tối đa 1000 ký tự, nhập từ bàn phím.

Output:

Chuỗi sau khi được sắp xếp các ký tự theo thứ tự tăng dần trong bảng mã ASCII.

Cấu trúc thuật toán

1. Nhập chuỗi từ người dùng (cin.getline(s, 1001)).
2. Sắp xếp các ký tự trong chuỗi (SapXepChuoi(s)):

Tính độ dài chuỗi (strlen(s)).

Sử dụng hàm sort(s, s + len) để sắp xếp chuỗi theo thứ tự tăng dần của bảng mã ASCII.

1. Xuất chuỗi sau khi sắp xếp (cout << "Chuoi sau khi sap xep: " << s << endl;).

**Lab1d\_Bài tập phải làm\_Câu 2**

Xác định Input và Output

Input:

Một chuỗi ký tự s có độ dài tối đa 1000 ký tự.

Một số nguyên k (0-based index), là vị trí của ký tự cần xóa trong chuỗi.

Output:

Nếu k không hợp lệ (nhỏ hơn 0 hoặc lớn hơn hoặc bằng độ dài chuỗi), in thông báo:

Vi tri k khong hop le!

Nếu k hợp lệ, in chuỗi sau khi đã xóa ký tự tại vị trí k.

Cấu trúc thuật toán

1. Nhập chuỗi từ người dùng (cin.getline(s, 1001)).
2. Nhập vị trí k cần xóa (cin >> k).
3. Kiểm tra tính hợp lệ của k:

Nếu k < 0 hoặc k >= len(s), in thông báo lỗi và kết thúc.

1. Xóa ký tự tại vị trí k (XoaKyTuTaiViTri(s, k)):

Dịch chuyển các ký tự bên phải s[k] sang trái một vị trí.

Đặt ký tự cuối cùng thành '\0' để kết thúc chuỗi.

1. Xuất chuỗi sau khi xóa (cout << s).

**Lab1d\_Bài tập phải làm\_Câu 3**

**Xác định Input và Output**

Input:

Một chuỗi ký tự s có độ dài tối đa 1001 ký tự.

Một số nguyên k (0-based index), là vị trí cần chèn ký tự vào trong chuỗi.

Một ký tự c cần chèn vào vị trí k.

Output:

Nếu k không hợp lệ (k < 0 hoặc k > len), in thông báo:

Vi tri k khong hop le!

Nếu k hợp lệ, in chuỗi sau khi đã chèn ký tự c vào vị trí k.

Cấu trúc thuật toán

1. Nhập chuỗi s.
2. Nhập vị trí k cần chèn.
3. Nhập ký tự c cần chèn.
4. Kiểm tra k có hợp lệ không:

Nếu k < 0 hoặc k > len, in thông báo lỗi và kết thúc.

1. Dịch chuyển các ký tự trong chuỗi sang phải từ vị trí k.
2. Chèn ký tự c vào vị trí k.
3. Cập nhật độ dài chuỗi.
4. Xuất chuỗi sau khi chèn.

**Lab1e\_Bài tập cơ bản\_Câu 1**

Xác định Input và Output

Input:

Một tập tin có tên "DaySoNguyen.inp", chứa:

Một số nguyên n (số lượng phần tử).

n số nguyên tiếp theo.

Output:

Nếu không thể mở file, in thông báo:

Khong the mo file!

Nếu mở file thành công, xuất ra màn hình:

Số lượng phần tử (n).

Danh sách n số nguyên trong file, cách nhau bằng khoảng trắng.

Cấu trúc thuật toán

1. Mở file "DaySoNguyen.inp" để đọc.
2. Kiểm tra nếu file không mở được → in thông báo lỗi và kết thúc.
3. Đọc số nguyên n từ file.
4. Khai báo mảng có n phần tử để lưu dãy số.
5. Duyệt file và đọc n số nguyên vào mảng.
6. Đóng file sau khi đọc xong.
7. Xuất dữ liệu:

In số lượng phần tử (n).

In các số nguyên trong mảng.

**Lab1e\_Bài tập cơ bản\_Câu 2**

Xác định Input và Output

Input:

Một tập tin có tên "MangSo.inp", chứa:

Hai số nguyên n và m (số dòng và số cột của ma trận).

n × m số nguyên tiếp theo, biểu diễn một ma trận n x m.

Output:

Nếu không thể mở file, in thông báo:

Khong the mo file!

Nếu mở file thành công, xuất ra màn hình:

Dòng đầu tiên:

Ma tran n x m doc tu file:

Tiếp theo là n dòng chứa các phần tử của ma trận, mỗi phần tử cách nhau một khoảng trắng.

Cấu trúc thuật toán

1. Mở file "MangSo.inp" để đọc.
2. Kiểm tra nếu file không mở được → in thông báo lỗi và kết thúc.
3. Đọc hai số nguyên n và m (số dòng và số cột của ma trận).
4. Khai báo mảng hai chiều arr[n][m] để lưu dữ liệu.
5. Duyệt file và đọc n × m số nguyên vào ma trận.
6. Đóng file sau khi đọc xong.
7. Xuất dữ liệu:

In dòng Ma tran n x m doc tu file:

In từng phần tử của ma trận theo hàng và cột.

**Lab1e\_Bài tập cơ bản\_Câu 3**

Xác định Input và Output

Input:

Tệp "NT.INP" chứa:

Số nguyên n (số lượng phần tử).

n số nguyên tiếp theo.

Output:

Tệp "NT.OUT" chứa:

Dòng đầu tiên là số lượng số nguyên tố tìm được.

Dòng thứ hai là danh sách các số nguyên tố theo thứ tự tăng dần, cách nhau một khoảng trắng.

Nếu không thể mở file, in thông báo lỗi:

Khong the mo file!

Nếu không có số nguyên tố nào, ghi 0 vào file.

Cấu trúc thuật toán

1. Mở file "NT.INP" để đọc và "NT.OUT" để ghi.

Nếu không mở được file → In thông báo lỗi và kết thúc.

1. Đọc số nguyên n từ file.
2. Duyệt n số nguyên, kiểm tra số nguyên tố:

Nếu là số nguyên tố → Thêm vào danh sách primeNumbers.

1. Sắp xếp danh sách số nguyên tố theo thứ tự tăng dần.
2. Ghi số lượng số nguyên tố vào "NT.OUT".
3. Ghi danh sách số nguyên tố vào "NT.OUT".
4. Đóng file.
5. Xuất thông báo ra màn hình:
6. Xu ly xong, vui long kiem tra NT.OUT

**Lab1e\_Bài tập cơ bản\_Câu 4**

Xác định Input và Output

Input:

Tệp "CapSo.INP" chứa:

n (số lượng phần tử của mảng).

k (tổng cần tìm).

n số nguyên tiếp theo (các phần tử của mảng).

Output:

Tệp "CapSo.OUT" chứa:

Nếu tìm được cặp số có tổng bằng k:

Ghi mỗi cặp số trên một dòng, theo thứ tự xuất hiện.

Nếu không tìm thấy cặp số nào:

Ghi 0.

Nếu không mở được file, in thông báo lỗi:

Khong the mo file!

Sau khi xử lý xong, in thông báo:

Xu ly xong, vui long kiem tra CapSo.OUT

Cấu trúc thuật toán

1. Mở file "CapSo.INP" để đọc và "CapSo.OUT" để ghi.

Nếu không mở được file → In thông báo lỗi và kết thúc.

1. Đọc n và k từ file.
2. Đọc n số nguyên vào mảng arr.
3. Khởi tạo tập hợp seen để lưu các số đã duyệt.
4. Duyệt qua từng phần tử arr[i]:

Tính complement = k - arr[i].

Nếu complement đã xuất hiện trong seen → Thêm {complement, arr[i]} vào danh sách cặp số.

Thêm arr[i] vào seen.

1. Ghi kết quả ra file "CapSo.OUT":

Nếu danh sách cặp số rỗng → Ghi 0.

Ngược lại, ghi từng cặp số lên mỗi dòng.

1. Đóng file.
2. Xuất thông báo ra màn hình.

**Lab1c\_Phần 2\_Bài tập cơ bản\_Câu 1**

Xác định Input và Output

Input:

Người dùng nhập các lựa chọn trong menu (1-5) và dữ liệu liên quan:

Thêm điện thoại: Nhập mã, nhãn hiệu, giá.

Tìm điện thoại: Nhập mã cần tìm.

Cập nhật giá: Nhập mã cần cập nhật và giá mới.

Xóa điện thoại: Nhập mã cần xóa.

Xuất danh sách: Hiển thị tất cả điện thoại.

Lặp lại cho đến khi chọn 0 để thoát.

Output:

Thông báo kết quả của từng thao tác:

Thêm điện thoại: Thành công hoặc lỗi (mã trùng, danh sách đầy).

Tìm điện thoại: Thông tin điện thoại hoặc không tìm thấy.

Cập nhật giá: Thành công hoặc không tìm thấy.

Xóa điện thoại: Thành công hoặc không tìm thấy.

Xuất danh sách: Hiển thị danh sách hoặc báo rỗng.

Xuất menu mỗi lần sau khi thực hiện thao tác.

Cấu trúc thuật toán

1. Khởi tạo đối tượng QuanLyDienThoai chứa danh sách điện thoại.
2. Lặp lại đến khi người dùng chọn "0" để thoát:

Hiển thị menu và nhận lựa chọn.

Xử lý lựa chọn:

1: Thêm điện thoại.

Nhập dữ liệu, kiểm tra mã trùng, thêm vào danh sách.

2: Tìm điện thoại theo mã.

Nhập mã, tìm kiếm, hiển thị kết quả.

3: Cập nhật giá.

Nhập mã, tìm kiếm, cập nhật giá nếu tìm thấy.

4: Xóa điện thoại theo mã.

Nhập mã, tìm kiếm, xóa nếu tồn tại.

5: Xuất danh sách điện thoại.

Duyệt danh sách, hiển thị thông tin.

0: Thoát chương trình.

Lựa chọn không hợp lệ: Hiển thị thông báo lỗi.

1. Kết thúc chương trình với thông báo.

**Lab1c\_Phần 2\_Bài tập cơ bản\_Câu 2**

Xác định Input, Output và Cấu trúc thuật toán

1. Input và Output

Input:

Người dùng nhập từ bàn phím:

Thêm mặt hàng: Mã, tên, số lượng, đơn giá.

Tìm mặt hàng: Nhập tên cần tìm.

Tính tổng số lượng: Không cần nhập.

Xuất danh sách: Không cần nhập.

Lựa chọn menu: Nhập số từ 0 đến 4 để chọn thao tác.

Output:

Thêm mặt hàng: Xác nhận thêm thành công hoặc báo lỗi nếu danh sách đầy.

Tìm mặt hàng: Hiển thị thông tin hoặc thông báo không tìm thấy.

Tính tổng số lượng: Hiển thị tổng số lượng mặt hàng.

Xuất danh sách: Hiển thị toàn bộ mặt hàng.

Lỗi nhập liệu: Nếu nhập sai lựa chọn menu, thông báo lỗi.

2. Cấu trúc thuật toán

1. Khởi tạo đối tượng QuanLyMatHang chứa danh sách mặt hàng.
2. Lặp lại đến khi người dùng chọn "0" để thoát:

Hiển thị menu và nhận lựa chọn từ bàn phím.

Xử lý theo lựa chọn:

1: Thêm mặt hàng vào danh sách.

Kiểm tra danh sách có đầy chưa.

Nhập mã, tên, số lượng, đơn giá.

Thêm vào danh sách.

2: Tìm mặt hàng theo tên.

Nhập tên, tìm kiếm, hiển thị kết quả.

3: Tính tổng số lượng mặt hàng.

Duyệt danh sách và tính tổng.

4: Xuất danh sách mặt hàng.

Hiển thị thông tin từng mặt hàng.

0: Thoát chương trình.

Lựa chọn không hợp lệ: Hiển thị lỗi.

1. Kết thúc chương trình với thông báo.