**Bài 27: Kiểm tra một số vừa nhập có phải số nguyên tố**

Giả sử kiểm tra x có phải số nguyên tố hay không?

Phương pháp thông thường: chia x cho các số từ 2 đến x-1, xem x có chia hết hay không 🡪 Tốn thời gian và vòng lặp.

**Lý do:**

Nếu x không phải số nguyên tố: với a và b là số nguyên dương khác 1 và x.

Giả sử gây mâu thuẫn với ban đầu x = a x b => ít nhất một trong hai số a hoặc b phải nhỏ hơn hoặc bằng căn x.

Ví dụ: x = 36



1 x 36

2 x 18

3 x 12

4 x 9

6 x 6

9 x 4

12 x 3

18 x 2

36 x 1

**Bài 29: Bài tập ngày tháng năm**

**Câu b**

Ý tưởng:

* Cộng số ngày các tháng trước đó
* Cộng thêm ngày của tháng hiện tại

**Câu c**

Nếu ngày == 1

Nếu tháng == 1

🡪 ngày = 31, tháng 12, năm = năm – 1

Ngược lại tháng != 1

🡪 tháng = tháng – 1

🡪 ngày = số ngày trong tháng trước

Ngược lại ngày != 1

Ngày = ngày – 1

**Câu d**

Xử lý như sau:

* 14/07/2024 🡪 chưa cuối tháng 🡪 tăng ngày (15/07/2024)
* 30/04/2024 🡪 cuối tháng 4 🡪 tăng tháng (01/05/2024)
* 28/02/2023 🡪 cuối tháng 2 🡪 sang 01/03/2023
* 29/02/2024 🡪 cuối tháng 2 nhuận 🡪 sang 01/03/2024
* 31/12/2024 🡪 cuối năm 🡪 sang 01/01 năm sau (01/01/2025)

**Bài 30: Truyền tham chiếu và tham trị**

Tham trị 🡪 bạn photo giá trị gốc rồi đưa cho hàm 🡪 hàm làm việc trên bản sao, không ảnh hưởng giá trị thật

Tham chiếu 🡪 bạn đưa địa chỉ nhà của biến cho hàm 🡪 khi biến trong hàm thay đổi thì cũng thay đổi giá trị biến trong hàm main.

**Bài 35: Mảng một chiều**

**Mảng là:**

* Một kiểu dữ liệu có cấu trúc do người lập trình định nghĩa.
* Biểu diễn một dãy các biến có cùng kiểu. Ví dụ: dãy các số nguyên, dãy các ký tự…
* Kích thước được xác định ngay khi khai báo và không bao giờ thay đổi (mảng tĩnh).

**Khai báo mảng:**

<kiểu dữ liệu> <tên biến mảng>[<số phần tử>];

- Chúng ta có thể khởi tạo giá trị cho mảng một chiều lúc khai báo.

- Gán dữ liệu và truy xuất dữ liệu vào các phần tử bên trong mảng.

**Bài 40: Sắp xếp tăng dần, giảm dần trong mảng**

A screenshot of a game

AI-generated content may be incorrect.**Thuật toán sắp xếp bubble sort**

**Bài 43: Tìm phần tử lớn thứ hai, nhỏ thứ hai trong mảng**

Ý tưởng

1, Duyệt 1 lần để tìm max1 và min1

2, Duyệt lại để tìm max2 và min2. Tức muốn tìm được min2 và max2 phải tìm được max1 và min 1.

**Bài 44: Xoá phần tử theo vị trí, theo điều kiện trong mảng**

Giả sử x[n] = [10, 20, 30, 40, 50]

Muốn xoá phần tử ở vị trí 2 (x[2] = 30)

🡪 vòng for bắt đầu từ vị trí 2 chạy đến n-2 🡪 mỗi lần lặp x[i] = x[i+1]

|  |  |  |
| --- | --- | --- |
| i | câu lệnh | kết quả |
| 2 | x[2] = x[3] | [10, 20, 40, 40, 50] |
| 3 | x[3] = x[4] | [10, 20, 40, 50, 50] |

🡪 sau khi dịch xong phần tử cuối bị trùng 🡪 giảm n.

Kết quả: x[n] = [10, 20, 40, 50]

void xoaMangTheoGiaTri(int x[], int &n){

int gt;

printf("\nnhap gia tri can xoa: ");

scanf("%d", &gt);

for(int i=0; i<n; i++){

if(x[i]==gt){

xoaMangTaiViTri(x, n, i);

i--;

}

}

}

Giải thích i--

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Chỉ số | 0 | 1 | 2 | 3 |
| Giá trị | 1 | 5 | 5 | 7 |

Giả sử nhập gt = 5

Khi i=1:

x[1] = 5 = gt 🡪 xoá 🡪 kết quả x[n] = [1 5 7]

Nếu không có i--

i tăng 1 lên 2🡪 x[2] = 7 nó bỏ qua vị trí vừa dồn lên là 5 dẫn đến kết quả x[n] = [1, 5, 7] 🡪 số 5 vẫn còn.

**Bài 45: Chèn phần tử vào đầu mảng, cuối mảng hoặc vào vị trí k của mảng**

// them phan tu vao dau mang

void themVaoDauMang(int x[], int &n, int m){

n++;

for(int i=n-1; i>0; i--){

x[i] = x[i-1]

}

x[0] = m;

}

Ví dụ: x[n] = [1 2 3 4 5] => n = 5

* i = n – 1 🡪 vị trí cuối mảng

|  |  |  |  |
| --- | --- | --- | --- |
| Lần lặp | i | Thao tác | Kết quả mảng |
| 1 | 5 | x[5] = x[4] | [1 2 3 4 5 5] |
| 2 | 4 | x[4] = x[3] | [1 2 3 4 4 5] |
| 3 | 3 | x[3] = x[2] | [1 2 3 3 4 5] |
| 4 | 2 | x[2] = x[1] | [1 2 2 3 4 5] |
| 5 | 1 | x[1] = x[0] | [1 1 2 3 4 5] |