

Hướng dẫn thực hành Sắp xếp, tìm kiếm

MỤC TIÊU, TÓM TẮT

Khái niệm:

Sắp xếp là cách thức **đặt các phần tử** của danh sách vào **đúng trật tự thỏa 1 tiêu chí** nào đó dựa trên nội dung của các phần tử.

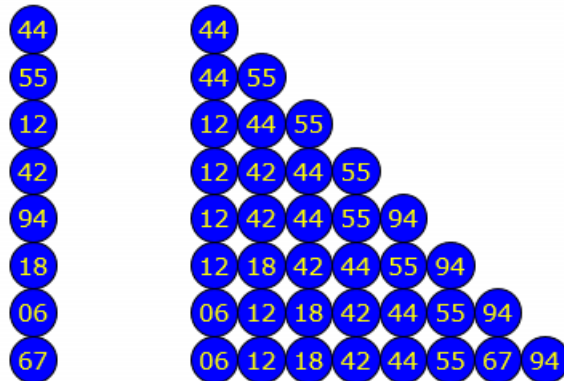
1. Insertion sort

- Ý tưởng: chèn một phần tử vào một đoạn đã sắp xếp sao cho vẫn đảm bảo đúng thứ tự đoạn con đó.
- Với dãy ban đầu, lấy đoạn con là phần tử đầu tiên xem như đã sắp xếp, chèn phần tử a_1 vào đoạn con a_0 , chèn phần tử a_2 vào đoạn con a_0a_1 . Lặp lại cho đến khi không còn phần tử để chèn thì dãy được sắp xếp.
- Độ phức tạp thuật toán $O(n^2)$.

Insertion Sort Execution Example

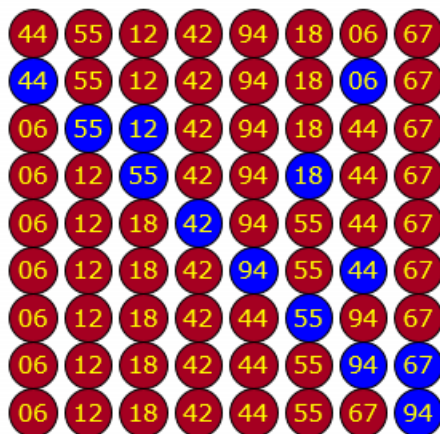


- Ban đầu xem như phần sắp xếp chỉ có 1 phần tử



2. Selection Sort

- ✓ Đưa phần tử nhỏ nhất về vị trí đầu mảng, đưa phần tử nhỏ thứ hai về vị trí thứ hai trong mảng, v.v...
- ✓ Quá trình được lặp lại cho đến khi mảng được sắp xếp.



.....

NỘI DUNG THỰC HÀNH

- sắp xếp danh sách tăng dần theo Insertion sort , Selection Sort,

Làm tương tự cho các thuật toán sx còn lại.

1. INSERTION
2. SELECTION
3. BUBBLE
4. INTERCHANGE
5. QUICK SORT
6. HEAP SORT

7. MERGE SORT

CHƯƠNG TRÌNH MẪU

// Cau 1.2: NHAP DANH SACH

```
void input(int a[], int &n)
{
    cout << "Nhap vao so luong phan tu cua danh sach:";
    cin >> n;
    cout << "Nhap vao cac phan tu cua danh sach:" << endl;
    for (int i = 0; i < n; i++) {
        cout << "a[" << i << "]=";
        cin >> a[i];
    }
}
```

//Cau 1.3: XUAT DANH SACH

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

// Cau 1.4 Insertion Sort

```
void Insertion(int a[],int n)
{
    for(int i= 1;i<n;i++)
    {
        int x=a[i]; //phan tu muon chen vao doan con
        int pos = i-1;
        /* Di chuyen cac ptu co gtri lon hon gtri key
        ve sau 1 vtri so voi vtri ban dau cua no */
        while(pos >= 0 && a[pos] > x)
        {
            a[pos+1]=a[pos];
            pos--;
        }
        a[pos+1] = x;
    }
}
```

```

void Selection(int a[],int n)
{
    // Di chuyển ranh giới của mảng đã sắp xếp và chưa sx
    for(int k=0;k<n-1;k++)
    {
        // tìm ptu nhỏ nhất trong mảng chưa sx
        int vt_min = k;
        for(int i= k+1;i<n;i++)
            if(a[i]<a[vt_min]) vt_min =i;
        // đổi cho ptu nhỏ nhất với ptu đầu tiên
        hoanVi(a[vt_min],a[k]);
    }
}

```

.....

// SV TỰ BỔ SUNG ĐỂ CHO RA MÀN HÌNH NHƯ YÊU CẦU BÊN DƯỚI

Bài 1:

Làm như hướng dẫn bên trên và xuất ra được màn hình như bên dưới

```

int a[] = { 3,5,9,7,3,20};
int n = sizeof(a)/sizeof(a[0]);

```

D:\code_CTDL_DHM\CTDL_1\SapXep\Insertion.exe

```

Xep thu tu danh sach bang INSERTION SORT
3      3      5      7      9      20

```

```

int a[] = { 3,-5,9,7,20};
int n = sizeof(a)/sizeof(a[0]);

```

D:\code_CTDL_DHM\CTDL_1\SapXep\SelectionSort.exe

```

Xep thu tu danh sach bang SELECTION SORT
-5      3      7      9      20

```

```

-----
Process exited after 0.06471 seconds with return value 0
Press any key to continue . . .


```

.....

Làm tương tự cho các thuật toán sx còn lại.

1. BUBBLE
2. INTERCHANGE
3. QUICK SORT
4. HEAP SORT
5. MERGE SORT

Bài 2:

 D:\code_CTDL_DHM\CTDL_1\SapXep\sx_all_menu.exe

```
----- CHUONG 3 : XEP THU TU va TIM KIEM -----
0. Khoi tao danh sach ngau nhien
1. Nhap danh sach
2. Xuat danh sach
3. Xep thu tu danh sach bang SELECTION SORT
4. Xep thu tu danh sach bang INSERTION SORT
5. Xep thu tu danh sach bang BUBBLE SORT
6. Xep thu tu danh sach bang INTERCHANGE SORT
7. Xep thu tu danh sach bang QUICK SORT
8. Xep thu tu danh sach bang HEAP SORT
9. Xep thu tu danh sach bang MERGE SORT
10. Tim kiem phan tu x bang TIM KIEM TUAN TU
11. Tim kiem phan tu x bang TIM KIEM NHI PHAN
12. Thoat

Vui long chon so de thuc hien:
```

Bài 3:

Quản lý một danh sách có số phần tử khá lớn, biến động. Mỗi phần tử có kiểu int. Thường có nhu cầu truy xuất phần tử đứng trước và phần tử đứng sau phần tử đang truy xuất. (Dùng cấu trúc danh sách liên kết đôi).

1. Khai báo cấu trúc danh sách.
2. Viết thủ tục khởi tạo danh sách rỗng.
3. Xuất các phần tử trong danh sách
4. Viết thủ tục thêm một phần tử vào đầu danh sách.
5. Viết thủ tục thêm một phần tử vào cuối danh sách.
6. Viết thủ tục xóa phần tử đầu danh sách.
7. Viết thủ tục xóa phần tử cuối danh sách.
8. Viết thủ tục tìm một phần tử trong danh sách. Nếu tìm thấy, xóa phần tử này.
9. Viết thủ tục tìm một phần tử có giá trị bằng với giá trị X hoặc gần nhất và lớn hơn phần tử nhập vào;
10. Thêm một phần tử đứng trước phần tử tìm thấy.