BÁO CÁO

Hình ảnh thực nghiệm với các số ngẫu nhiên.

```
Bạn muốn các thuật toán chạy mấy lần
10
Bạn muốn thực hiện các thuật toán với kiểu dữ liệu nào ?
1. 10000 số ngẫu nhiên
2. 10000 số tăng dần
3. 10000 số giảm dần
1
Test các thuật toán 10000 số ngẫu nhiên
Lần : 1
Thời gian xử lý phép toán của bubble Sort là : 166243100
Thời gian xử lý phép toán của selection Sort là : 69106200
Thời gian xử lý phép toán của insertion Sort là : 38676000
Lần : 2
Thời gian xử lý phép toán của bubble Sort là : 142009800
Thời gian xử lý phép toán của selection Sort là : 54026800
Thời gian xử lý phép toán của insertion Sort là : 26077100
Lần : 3
Thời gian xử lý phép toán của bubble Sort là : 115693700
Thời gian xử lý phép toán của selection Sort là : 33655500
Thời gian xử lý phép toán của insertion Sort là : 18338800
Lần : 4
Thời gian xử lý phép toán của bubble Sort là : 143176400
Thời gian xử lý phép toán của selection Sort là : 49697300
Thời gian xử lý phép toán của insertion Sort là : 22441600
Lần : 5
Thời gian xử lý phép toán của bubble Sort là : 149495200
Thời gian xử lý phép toán của selection Sort là : 48153600
Thời gian xử lý phép toán của insertion Sort là : 15502700
Lần : 6
Thời gian xử lý phép toán của bubble Sort là : 141327400
Thời gian xử lý phép toán của selection Sort là : 48570200
Thời gian xử lý phép toán của insertion Sort là : 19180500
Lần : 7
Thời gian xử lý phép toán của bubble Sort là : 140216100
Thời gian xử lý phép toán của selection Sort là : 45497200
Thời gian xử lý phép toán của insertion Sort là : 19768900
Lần : 8
Thời gian xử lý phép toán của bubble Sort là : 141188000
Thời gian xử lý phép toán của selection Sort là : 45948000
Thời gian xử lý phép toán của insertion Sort là : 18129800
Lần : 9
Thời gian xử lý phép toán của bubble Sort là : 142159700
Thời gian xử lý phép toán của selection Sort là : 29423400
```

Hình ảnh thực nghiệm với cái số được sắp xếp sẵn :

```
Bạn muốn các thuật toán chạy mấy lần
6
Bạn muốn thực hiện các thuật toán với kiểu dữ liệu nào ?
1. 10000 số ngẫu nhiên
2. 10000 số tăng dần
3. 10000 số giảm dần
2
Test các thuật toán 10000 số tăng dần
Lần : 1
Thời gian xử lý phép toán của bubble Sort là : 59389700
Thời gian xử lý phép toán của selection Sort là : 50401800
Thời gian xử lý phép toán của insertion Sort là : 9499100
Lần : 2
Thời gian xử lý phép toán của bubble Sort là : 33642600
Thời gian xử lý phép toán của selection Sort là : 30072700
Thời gian xử lý phép toán của insertion Sort là : 11224500
Lần : 3
Thời gian xử lý phép toán của bubble Sort là : 27114600
Thời gian xử lý phép toán của selection Sort là : 21399400
Thời gian xử lý phép toán của insertion Sort là : 3527500
Lần : 4
Thời gian xử lý phép toán của bubble Sort là : 27573600
Thời gian xử lý phép toán của selection Sort là : 22939000
Thời gian xử lý phép toán của insertion Sort là : 3420600
Lần : 5
Thời gian xử lý phép toán của bubble Sort là : 29607000
Thời gian xử lý phép toán của selection Sort là : 28499300
Thời gian xử lý phép toán của insertion Sort là : 2799600
Lần : 6
Thời gian xử lý phép toán của bubble Sort là : 31178800
Thời gian xử lý phép toán của selection Sort là : 22043800
Thời gian xử lý phép toán của insertion Sort là : 1689800

Process finished with exit code 0
```

Hình ảnh thực nghiệm với dãy số bị đảo ngược:

```
Bạn muốn các thuật toán chạy mấy lần
6
Bạn muốn thực hiện các thuật toán với kiểu dữ liệu nào ?
1. 10000 số ngẫu nhiên
2. 10000 số tăng dần
3. 10000 số giảm dần
3
Test các thuật toán 10000 số giảm dần
Lần : 1
Thời gian xử lý phép toán của bubble Sort là : 99446100
Thời gian xử lý phép toán của selection Sort là : 77269200
Thời gian xử lý phép toán của insertion Sort là : 55399200
Lần : 2
Thời gian xử lý phép toán của bubble Sort là : 71133900
Thời gian xử lý phép toán của selection Sort là : 62088700
Thời gian xử lý phép toán của insertion Sort là : 45936400
Lần : 3
Thời gian xử lý phép toán của bubble Sort là : 64262200
Thời gian xử lý phép toán của selection Sort là : 41875300
Thời gian xử lý phép toán của insertion Sort là : 31997400
Lần : 4
Thời gian xử lý phép toán của bubble Sort là : 63600600
Thời gian xử lý phép toán của selection Sort là : 59978100
Thời gian xử lý phép toán của insertion Sort là : 34268200
Lần : 5
Thời gian xử lý phép toán của bubble Sort là : 59185700
Thời gian xử lý phép toán của selection Sort là : 55294700
Thời gian xử lý phép toán của insertion Sort là : 35336700
Lần : 6
Thời gian xử lý phép toán của bubble Sort là : 61685200
Thời gian xử lý phép toán của selection Sort là : 61276300
Thời gian xử lý phép toán của insertion Sort là : 31596300

Process finished with exit code 0
```

I. Nhận xét chung :

Cho dù đối với kiểu dữ liệu nào thì :

Bubble sort vẫn tốn nhiều thời gian thực hiện nhất.

Insertion sort là thuật toán nhanh nhất

Selection sort là thuật toán có thời gian giải quyết trung bình.

Với kiểu dữ liệu ngẫu nhiên thì Bubble sort có thời gian giải quyết lâu hơn gấp rất nhiều lần 2 thuật toán còn lại.

Với kiểu dữ liệu được sắp xếp sẵn hay đảo ngược thì giữa bubble sort và selection sort đều có thời gian xử lý gần bằng nhau, và insertion sort vẫn nhanh gấp nhiều lần .

II. Giải thích kết quả thực nghiệm

1. Thuật toán

1.1 Bubble sort

1.1.1 Worst case :

Trường hợp xấu nhất xảy ra khi các phần tử bị sắp xếp ngược lại so với thứ tự cần sắp xếp. Lúc này tại mỗi vòng lập, ta cần hoán đổi vị trí của các phần tử từ đầu đền phần tử cuối cùng chưa được sắp xếp.

Tổng quát hơn, cho n là số lượng phần tử có trong mảng. Số lần hoán đổi cần thiết là $n*(n-1)/2$. Với công thức này, ta thấy cấp số cao nhất là n^2. Vậy BigO của Bubble Sort trong trường hợp xấu nhất là $O(n^2)$.

1.1.2 Average case:

Trong trường hợp trung bình, ta có thể tạm ước lượng được số lần hoán đổi sẽ bằng một nửa số lần hoán đổi trong trường hợp xấu nhất. Vậy số lần hoán đổi cần thiết là $n*(n-1)/4$. Với công thức này, ta thấy cấp số cao nhất là n^2. Vậy BigO của Bubble Sort trong trường hợp trung bình là $O(n^2)$.

1.1.3 Best case:

Trường hợp tốt nhất xảy ra khi mảng đã đươc sắp xếp trước. Lúc này, thuật toán sẽ xác định được trong lần lập đầu tiên rằng không cần hoán đổi vị trí của bất cứ phần tử nào trong mảng và kết thúc thực thi (phải có thêm 1 đoạn mã để break vòng lặp khi mảng đã được sắp xếp).

Vậy BigO của Bubble Sort trong trường hợp tốt nhất là $O(n)$.

1.2 Selection Sort

Trong mọi trường hợp. Selection Sort cần so sánh & tìm giá trị nhỏ nhất trong tất cả các phần tử chưa được sắp xếp. Cho n là số lượng phần tử có trong mảng. Số lần so sánh cần thiết là $n*(n–1)/2$. Với công thức này, ta thấy cấp số cao nhất là n^2. Vậy BigO của Selection Sort trong mọi trường hợp là $O(n^2)$. Tuy nhiên, Selection Sort thường có thời gian thực hiện nhanh hơn nhiều so với Bubble Sort do không phải thực hiện quá nhiều phép hoán đổi

1.3 Insertion Sort

1.3.1 Worst case :

Trường hợp xấu nhất xảy ra khi các phần tử bị sắp xếp ngược lại so với thứ tự cần sắp xếp. Lúc này tại mỗi vòng lập, ta cần dịch chuyển các phần tử đã sắp xếp sang bên phải 1 vị trí, sau đó chèn phần tử đang sắp xếp vào đầu mảng.

Cho n là số lượng phần tử có trong mảng. Số lần dịch chuyển cần thiết là $n*(n-1)/2$. Với công thức này, ta thấy cấp số cao nhất là n^2. Vậy BigO của Insertion Sort trong trường hợp xấu nhất là $O(n^2)$.

1.3.2 Average case:

Trong trường hợp trung bình, ta có thể tạm ước lượng được số lần dịch chuyển sẽ bằng một nửa số lần dịch chuyển trong trường hợp xấu nhất. Vậy số lần dịch chuyển cần thiết là n * (n - 1) / 4.. Với công thức này, ta thấy cấp số cao nhất là n^2. Vậy BigO của Insertion Sort trong trường hợp trung bình là $O(n^2)$.

1.3.3 Best case:

Trường hợp tốt nhất xảy ra khi mảng đã được sắp xếp sẵn. Lúc này Insertion Sort chỉ cần lập 1 lần duy nhất, các phần tử trong mảng không cần dịch chuyển vị trí. Vậy BigO của Insertion Sort trong trường hợp tốt nhất là O(n)

2. Giải thích

2.1 Mảng sắp xếp ngẫu nhiên

Bubble Sort chạy chậm nhất do nó cần lần lượt so sánh và hoán đổi vị trí của các phần tử trong mảng.

Selection Sort chạy nhanh hơn Bubble Sort do nó chỉ cần so sánh các phần tử trong mảng với nhau, việc hoan đổi chỉ phải thực hiện 1 lần trong mỗi vòng lập.

Insertion Sort chạy nhanh nhất do nó chỉ cần so sánh & dịch chuyển các phần tử đã sắp xếp mà lớn hơn phần tử đang so sánh.

2.2 Mảng sắp xếp theo thứ tự

Thời gian giải quyết phép toán của 3 thuật toán đều được rút ngắn đáng kể vì rơi là trường hợp tốt nhất của thuật toán.

2.3 Mảng sắp xếp theo thứ tự ngược lại

Các trường hợp đều rơi vào worst cast nhưng do cơ chế Branch prediction của CPU nên thời gian xử lý của các thuật toán đều được rút ngắn đáng kể.