ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN



BÁO CÁO ĐỒ ÁN GIỮA KÌ

Môn: Cấu trúc dữ liệu và giải thuật

Họ và tên: Âu Dương Gia Tuấn

MSSV: 1712867

MỤC LỤC

1. Giới thiệu chung

Về việc các *thuật toán sắp xếp* (*Sorting Algorithm*), ta có rất nhiều loại thuật toán khác nhau, ví dụ như: selection sort, insertion sort, interchange sort, binary interchange sort, radix sort, merge sort, heap sort, flash sort, counting sort, shell sort, bubble sort, shaker sort, quick sort, bucket sort, cocktail sort,….

Mỗi thuật toán trên đều có ưu và nhược điểm khác nhau, tùy thuộc vào từng bộ dữ liệu mà ta chọn mộ thuật toán thích hợp nhất để cài đặt.

Nhưng trong bài báo cáo này ta chỉ tìm hiểu về 10 thuật toán khá phổ biến hiện nay, đó là: selection sort, interchange sort, insertion sort, binary insertion sort, shaker sort, quick sort, heap sort, shell sort, merge sort, radix sort.

1. Tìm hiểu về các thuật toán

# Selection sort

## Ý tưởng

Ý tưởng của thuật toán Selection sort chia phần mảng ban đầu thành hai phần: phần mảng đã được sắp xếp và phần mảng chưa được sắp xếp.

Sau đó tiến hành sắp xếp mảng bằng cách liên tục tìm ra *phần tử nhỏ nhất (ví dụ sắp cho xếp tăng dần)* từ phần mảng chưa được sắp xếp và hoán đổi vị trí đó với vị trí đầu. Tiến trình này sẽ dừng lại khi toàn bộ phần tử của mảng chưa được sắp xếp đều được chuyển sang phần mảng đã được sắp xếp.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 10 | 4 | 7 | 3 | 15 | 1 |

Ví dụ:

**Mảng ban đầu:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 10 | 4 | 7 | 3 | 15 | 1 |

Vị trí đầu tiên chính là số 10 và sau khi duyệt mảng thì ta có số 1 là giá trị nhỏ nhất:

Ta tiến hành hoán đổi số 10 và số 1:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 4 | 7 | 3 | 15 | 10 |

Lúc này mảng đã được chia thành hai phần: phần đã sắp xếp và phần chưa sắp xếp

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 4 | 7 | 3 | 15 | 10 |

Tiến trình trên sẽ áp dụng cho phần còn lại của mảng.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 4 | 7 | 3 | 15 | 10 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 7 | 4 | 15 | 10 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 4 | 7 | 15 | 10 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 4 | 7 | 15 | 10 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 4 | 7 | 10 | 15 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 4 | 7 | 10 | 15 |

**Mảng đã sắp xếp:**

## Thuật toán

|  |
| --- |
| void hoanVi(int &a, int &b) |
| {  int temp;  temp = a;  a = b;  b = temp;  } |

|  |
| --- |
| void selectionSort(int\* arr, int soLuongPhanTu) |
| {  int min;  for (int i = 0; i < soLuongPhanTu - 1; i++)  {  min = i;  for (int j = i + 1; j < soLuongPhanTu; j++)  {  if (arr[min] > arr[j])  {  min = j;  }  }  if (i != min)  {  hoanVi(arr[i], arr[min]);  }  }  } |

## Đánh giá

|  |  |
| --- | --- |
| **Ưu điểm** | **Nhược điểm** |
| Thuật toán dễ cài đặt  Sắp xếp nhanh với mảng ít phần tử | Tốn nhiều bộ nhớ  Đối với mảng nhiều phần tử, thuật toán sắp xếp lâu vì độ phức tạp của thuật toàn là O( |

# Interchange sort



## Ý tưởng

Ý tưởng của thuật toán Interchange sort là bắt cặp tất cả phần tử trong mảng và đổi chổ cho nhau nếu hai phần tử đó thỏa điều kiện sắp xếp.

Thuật toán này sẽ dùng hai vòng lặp. Vòng lặp ngoài sẽ chạy từ *vị trí đầu tiên* đến vị trí thứ *n-1* của mảng. Vòng lặp trong sẽ chạy từ *vị trí* *ta đang xét + 1* đến vị trí *n* của mảng. Mỗi lần ta sẽ so sánh hai giá trị trong vòng lặp, nếu chúng thỏa điều kiện, sẽ tiến hành hoán đổi vị trí.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 10 | 4 | 7 | 3 | 15 | 1 |

**Ví dụ:**

**Mảng ban đầu:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 10 | 4 | 7 | 3 | 15 | 1 |

Ta sẽ tiến hành sắp xếp mảng trên tăng dần.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 4 | 10 | 7 | 3 | 15 | 1 |

Vì số 10 và 4 thỏa điều kiện nên ta sẽ hoán đổi vị trí hai số này.

Ta sẽ tiến hành quá trinh này cho đến khi chạy hết vòng lặp.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 4 | 10 | 7 | 3 | 15 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 4 | 10 | 7 | 3 | 15 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 3 | 10 | 7 | 4 | 15 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 3 | 10 | 7 | 4 | 15 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 3 | 10 | 7 | 4 | 15 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 10 | 7 | 4 | 15 | 3 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 10 | 7 | 4 | 15 | 3 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 7 | 10 | 4 | 15 | 3 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 10 | 7 | 4 | 15 | 3 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 4 | 7 | 10 | 15 | 3 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 4 | 7 | 10 | 15 | 3 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 4 | 7 | 10 | 15 | 3 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 7 | 10 | 15 | 4 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 7 | 10 | 15 | 4 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 7 | 10 | 15 | 4 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 7 | 10 | 15 | 4 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 4 | 10 | 15 | 7 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 4 | 10 | 15 | 7 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 4 | 10 | 15 | 7 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 4 | 7 | 15 | 10 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 4 | 7 | 15 | 10 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 4 | 7 | 10 | 15 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 3 | 4 | 7 | 10 | 15 |

**Mảng đã sắp xếp:**

## Thuật toán

|  |
| --- |
| void interchangeSort(int\* arr, int soLuongPhanTu) |
| {  for (int i = 0; i < soLuongPhanTu - 1; i++)  {  for (int j = i + 1; j < soLuongPhanTu; j++)  {  if (arr[i] > arr[j])  {  hoanVi(arr[i], arr[j]);  }  }  }} |

## Đánh giá

|  |  |
| --- | --- |
| **Ưu điểm** | **Nhược điểm** |
| Thuật toán dễ cài đặt  Sắp xếp nhanh với mảng ít phần tử | Tốn nhiều bộ nhớ  Đối với mảng nhiều phần tử, thuật toán sắp xếp lâu vì độ phức tạp của thuật toàn là O( |

# Insertion sort

## Ý tưởng

Ý tưởng của thuật toán insertion sort là