**BÁO CÁO TIẾN ĐỘ PROJECT I**

**Sinh viên thực hiện:** Lê Thạch Cương

**GVHD:** Nguyễn Khánh Phương

Tuần 2

Bài 1:

Thuật toán sắp xếp nhanh với 2 phiên bản:

2 way partitioning quicksort:

* Chọn phần tử đầu tiên làm **pivot**
* Chia mảng ra làm 2 phần với **pivot**. Phần 1 không lớn hơn pivot, phần 2 không nhỏ hơn pivot.
* Lặp lại đệ quy với 2 phần vừa chia cho đến khi dãy con có nhỏ hơn hoặc bằng 1 phần tử.
* Cuối cùng tổng hợp lại mảng đã sắp xếp.

3 way partitioning quicksort: Tương tự như 2 way nhưng mảng được chia thành 3 phần với 1 mảng có các phần tử bằng pivot, 1 mảng chứa các phần tử nhỏ hơn pivot và 1 mảng chứa các phần tử lớn hơn pivot.

Với cùng 1 dãy đầu vào chứa các số từ 1 đến 100 thì:

|  |  |  |
| --- | --- | --- |
| Dãy đầu vào/ cách sắp xếp | 2 way(s) | 3 way(s) |
| 10^4 | 0.004 | 0 |
| 10^5 | 0.012 | 0.005 |
| 10^6 | 0.124 | 0.057 |
| 10^7 | 1.272 | 0.579 |

Bài 2:

Phương pháp Horner method sẽ chia đa thức ban đầu thành tích của 1 đa thức nhỏ hơn với x-x0 với x0 là giá trị được nhập từ bàn phím và dư ra một số theo như bảng sau

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | An | An-1 | An-2 | … | A1 | Ao |
| Xo | Bn | Bn-1 | Bn-2 | … | B1 | Bo |

Bo chính là giá trị của đa thức cần tìm với dãy đầu vào và Xo.

Công thức tìm ra hệ số của đa thức sau: B[i]=B[i+1]\*Xo +A[i]

Bài 3:

* Tạo 2 mảng chứa các hệ số của đa thức thứ nhất và đa thức thứ 2.
* Tạo 1 mảng mới có bậc bằng bậc của đa thức 1 + đa thức thứ 2.
* Sử dụng 2 vòng for để tính các phần tử của mảng mới.
* Hệ số của đa thức cần tìm chính là các phần tử của mảng mới.

Bài 4:

* Thuật toán Kruskal tìm cây khung nhỏ nhất: Từ danh sách cạnh đầu vào, sắp xếp danh sách cạnh theo thứ tự trọng số tăng dần. Chọn lần lượt chúng vào trong cây khung sao cho không tạo thành chu trình. Ban đầu mỗi đỉnh sẽ được đại diện bằng chính nó, sau đó đỉnh đại diện sẽ được thay bởi 1 đỉnh khác. Cây khung sẽ hoàn thành khi có đủ n-1 cạnh với n là số đỉnh của cây ban đầu. Đối với việc sử dụng cấu trúc DSU thì cần khởi tạo 2 hàm Find và Unify, hàm Find có nhiệm vụ tìm đỉnh đại diện mới của các đỉnh và hàm Unify có nhiệm vụ hợp nhất các tập hợp.
* Thuật toán Prim tìm cây khung nhỏ nhất: Từ danh sách cạnh đầu vào, chuyển chúng thành danh sách kề với 2 phần (đỉnh kề , trọng số). Chọn 1 đỉnh bất kì đưa vào cây khung, duyệt toàn bộ danh sách kề của đỉnh đó có trọng số nhỏ nhất sao cho không tạo thành bất cứ chu trình nào. Làm như vậy cho đến khi có đủ n-1 cạnh với n là số đỉnh của cây khung ban đầu. Với thuật toán Prim có sử dụng hàng đợi có ưu tiên thì danh sách kề của đỉnh đã duyệt sẽ được đưa vào 1 hàng đợi. Và khi chọn đỉnh kế tiếp thì chỉ việc chọn trong danh sách này (Đưa vào hàng đợi thì thay đổi thứ tự (trọng số, đỉnh).
* Độ phức tạp thuật toán với từng cách cài đặt:

|  |  |  |  |
| --- | --- | --- | --- |
| Kruskal 1 | Kruskal 2 | Prim 1 | Prim 2 |
| O(N^2logN) | O(N^2logN) | O(N^3) | O(N^2) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Số đỉnh n | Kruskal 1 | Kruskal 2 | Prim 1 | Prim 2 |
| 10^2 | 0.907 | 0.142 | 0.07 | 0 |
| 10^3 | 1.888 | 1.197 | 6.762 | 0.013 |
| 10^4 | 10.691 | 9.898 | 120 (minute) | 1.076 |

(Thời gian tính theo giây trừ Prim 1 – 10^4)