**DAA – HOMEWORK 4 – DEVIDE AND CONQUER**

PHẦN 1: LÝ THUYẾT PHƯƠNG PHÁP

Ý tưởng của phương pháp chia để trị là:

1. Chia bài toán ban đầu thành các bài toán nhỏ hơn, giải quyết các bài toán này độc lập.
2. Kết hợp các kết quả đã thu được từ các bài toán nhỏ để thu được kết quả cuối cùng cho bài toán ban đầu.

Lược đồ tổng quát:

1. Nếu bài toán đạt được điều kiện dừng (trivial case), giải quyết trực tiếp bài toán này và trả về kết quả.
2. Chia bài toán ban đầu thành các bài toán nhỏ hơn và giải quyết bằng cách đệ quy.
3. Kết hợp các kết quả từ các bài toán nhỏ để thu được kết quả cuối cùng cho bài toán ban đầu.

Ví dụ: Thuật toán merge sort

Ý tưởng của thuật toán Merge Sort

1. Chia đôi mảng ban đầu, rồi tiếp tục chia đôi các nửa đó cho đến khi không thể chia được nữa (nửa đó chỉ có một phần tử) (Devide)
2. Ghép các nửa đó lại với nhau theo thứ tự tăng dần để tạo ra một mảng đã được sắp xếp (Combine)

Lược đồ cho thuật toán Merge Sort:

1. Bài toán cơ sở: nếu mảng có một phần tử, trả về mảng đó
2. Chia mảng thành 2 phần, đệ quy sắp xếp cho từng mảng
3. Kết hợp các mảng con lại với nhau để tạo ra mảng được sắp xếp.

Ví dụ: A = [5, 1, 2, 6, 7]

Kết hợp [1] với [5] ta được [1, 5]

Kết hợp [5, 1] với [2] ta được mảng: [1, 2, 5]

Kết hợp [6] với [7] ta được mảng: [6, 7]

Kết hợp [6, 7] với [1, 2, 5] ta được [1, 2, 5, 6, 7]

PHẦN 2: BÀI TẬP TƯ DUY

Bài 1: Bài toán giá cổ phiếu

- Một phương án để giải quyết bài toán là: dùng tiền mua cổ phiếu ở ngày thấp nhất, sau đó bán tất cả cổ phiếu đó ở ngày giá cổ phiếu cao nhất

- Nhiệm vụ của ta bây giờ là tìm ngày có giá cổ phiếu thấp nhất và ngày có giá cổ phiếu cao nhất

- Bài toán quy về tìm min và max của mảng.

- Sử dụng phương pháp chia để trị để giải quyết bài toán bằng cách:

+ Chia mảng giá cổ phiếu đã cho thành 2 phần

+ Đệ quy tìm min,max cho mỗi phần

+ min = min(phần 1, phần 2), max = max(phần 1, phần 2)

+ Bài toán cơ sở:

+ Nếu n = 1, đây vừa là giá trị min vừa là giá trị max

+ Nếu n = 2, min = min(phần tử 1, phần tử 2), max = phần tử còn lại

Bài 2: Thử nghiệm laptop “unbreakable”

- Nhiệm vụ là tìm ra tầng thấp nhất có thể để laptop có thể vỡ khi thả

- Đây chính là bài toán tìm kiếm, giải pháp đưa ra sẽ tương tự như thuật toán binary search

- B1. Thả laptop từ tầng thứ n/2

- B2. Kiểm tra xem laptop có vỡ hay không

- Nếu vỡ, chứng tỏ tầng cần tìm nằm ở phía dưới tầng n/2 (tức từ 0 🡪 n/2 -1)

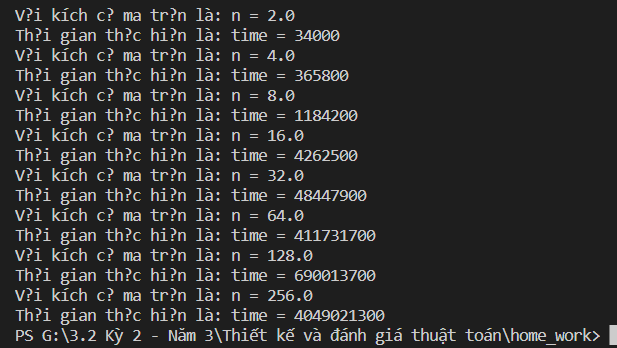
- Nếu không, chứng tỏ tầng cần tìm nằm ở phía trên tầng n/2 (tức n/2 + 1 🡪 n)

- B3. Tiếp tục chia số tầng và thực hiện thả laptop với laptop thứ 2

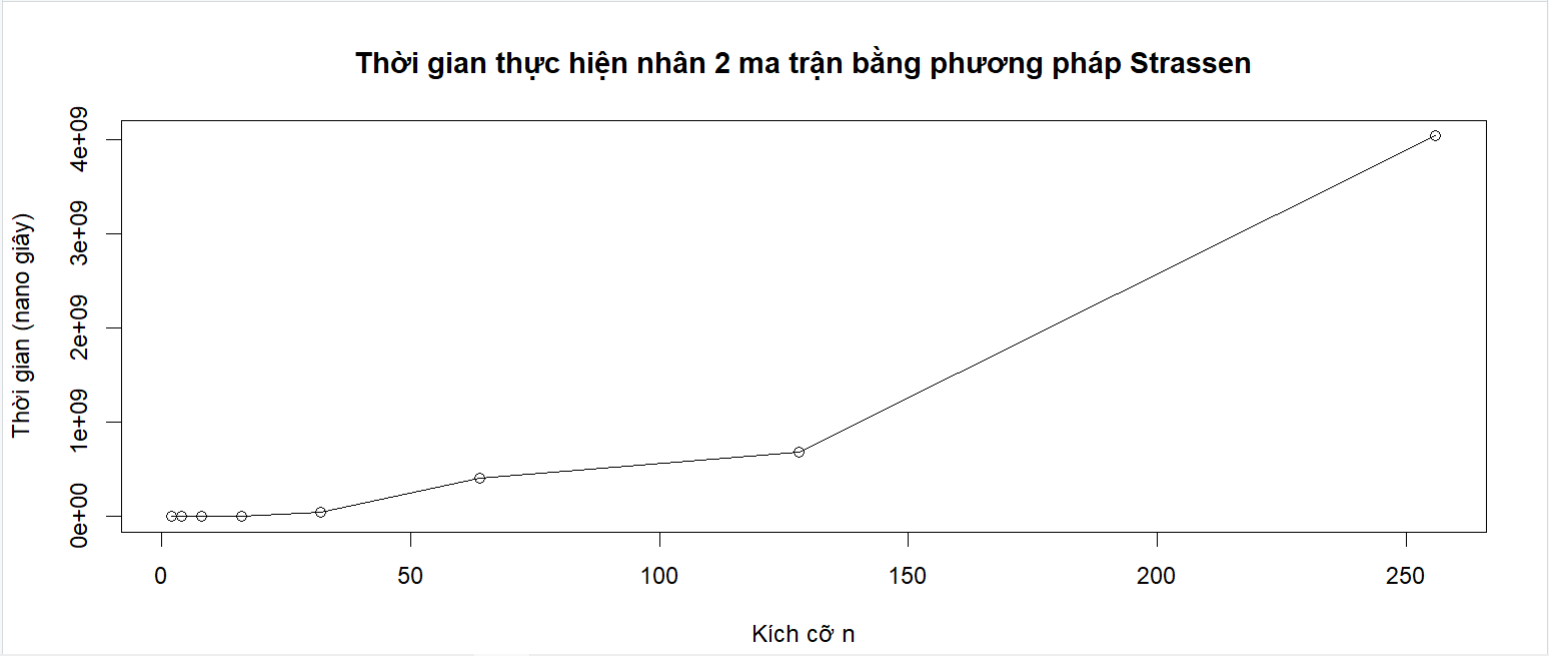
PHẦN 3: BÀI TẬP LẬP TRÌNH

Bài 1: Nhân ma Với bằng phương pháp Strassen

* Sử dụng vòng lặp for để tạo ra 8 cặp ma Với với kích cỡ lần lượt là 2, 4, 8, 16, 32, 64, 128, 256, giá trị của các phần tử trong ma Với được sinh ngẫu nhiên. Ta có bảng kết quả:



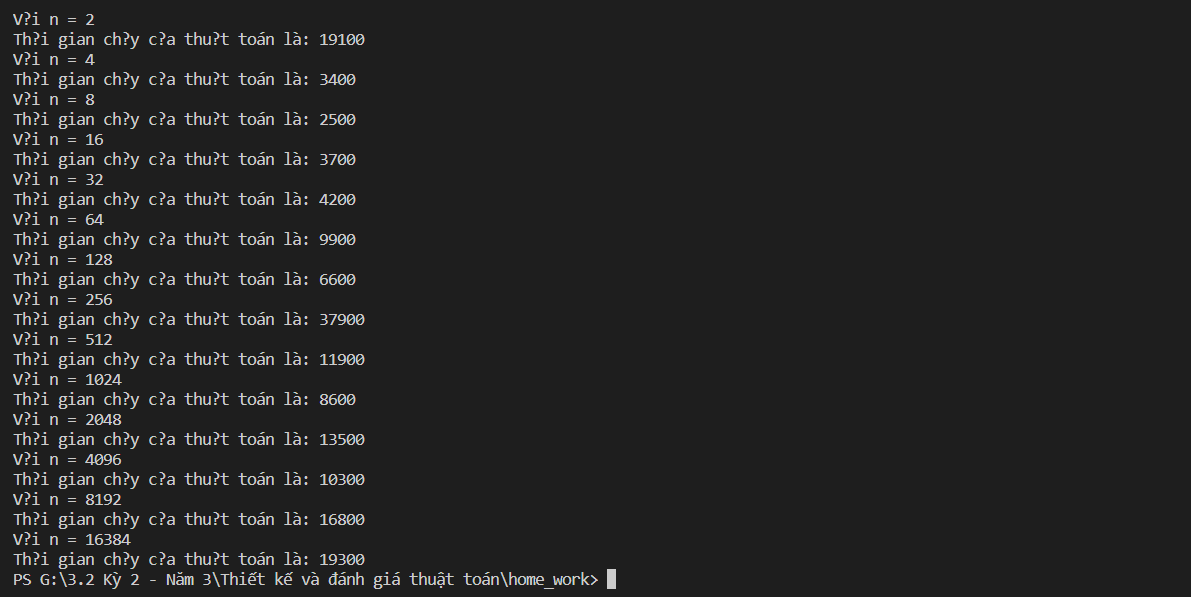
* Sử dụng phần mềm RStudio vẽ biểu đồ, ta có:



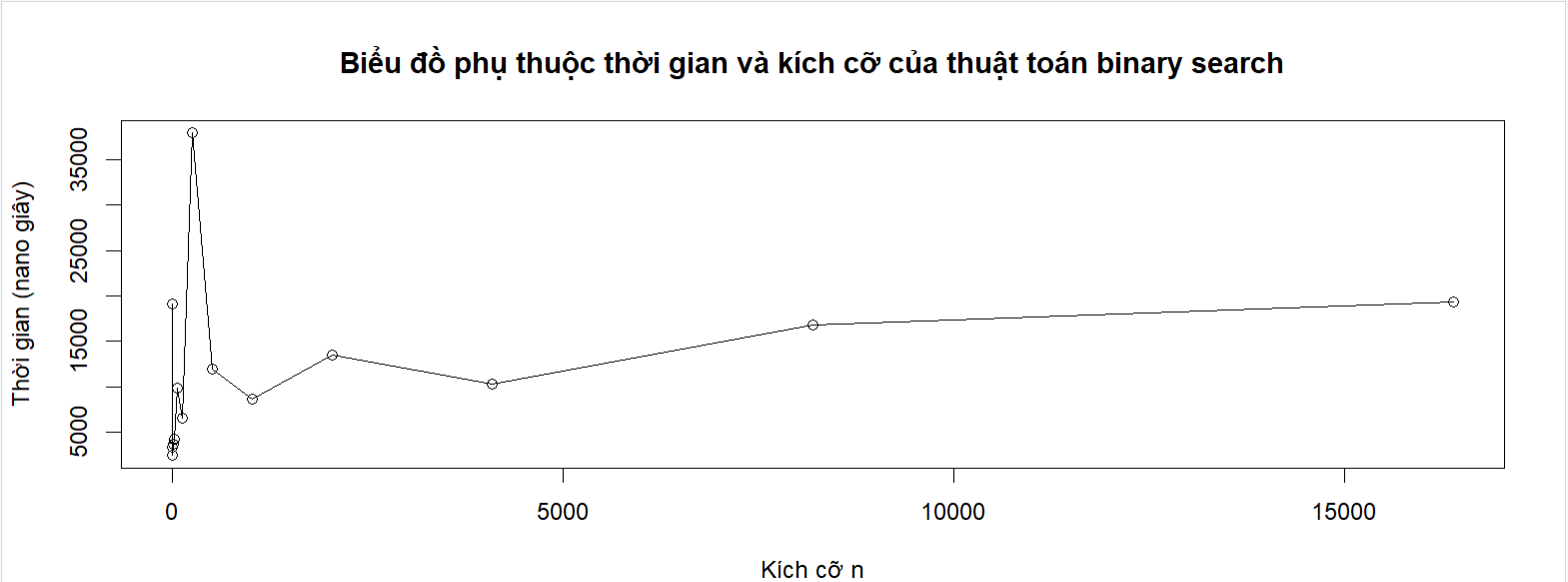
Thời gian thực hiện thuật toán tỉ lệ thuận với kích cỡ, qua biểu đồ ta thấy rằng thuật toán tăng rất nhanh đặc biệt khi n lớn

Bài 2: Binary search

* Sử dụng vòng lặp for để tạo ra các mảng với kích cỡ khác nhau, giá trị của các phần tử trong được sinh ngẫu nhiên. Ta có bảng kết quả:



Sử dụng phần mềm Rstudio, ta có biểu đồ sau đây:

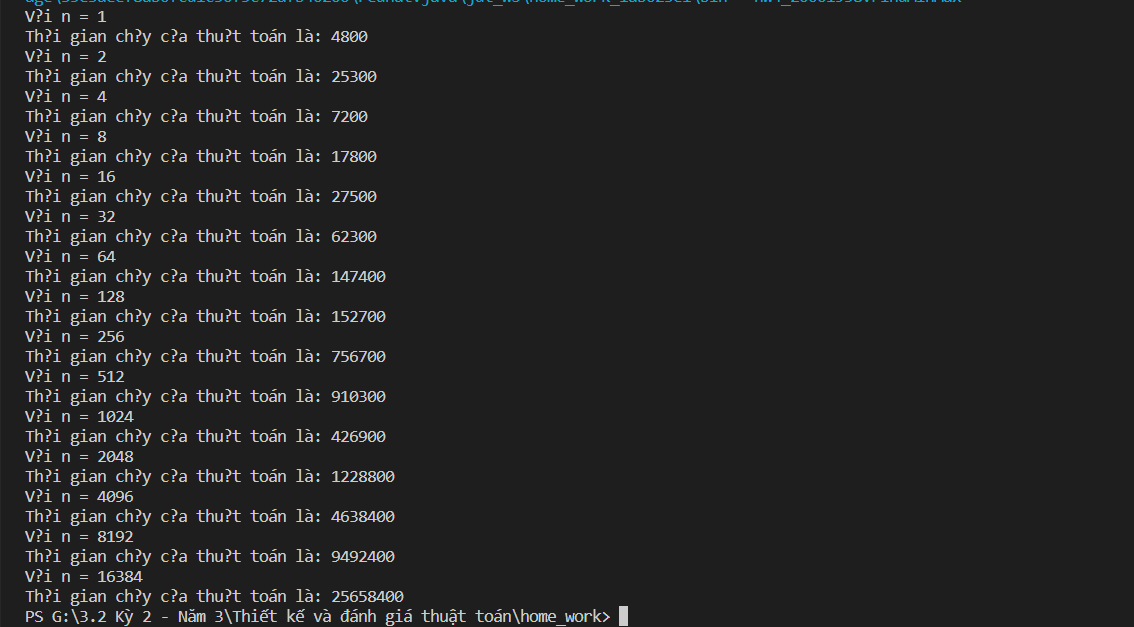


Qua biểu đồ, ta nhận thấy:

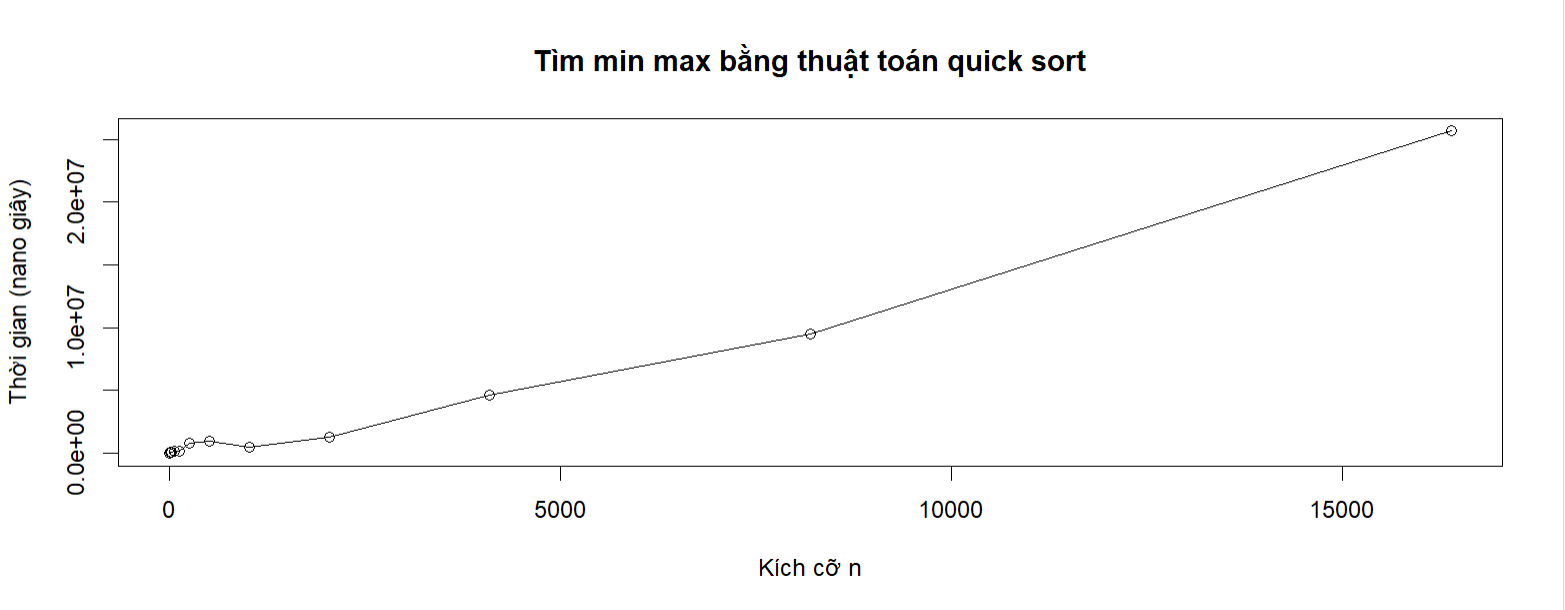
* Với n nhỏ, biểu đồ phụ thuộc rất biến động
* Với n đủ lớn, cỡ > 1000, biểu đồ phụ thuộc ổn định hơn , thậm chí trong một vài trương hợp, thời gian thực hiện thuật toán còn nhanh hơn khi n nhỏ < 1000.
* Thời gian thực hiện khi khi n lớn cỡ 1.000 < n < 16.500 dao động trong khoảng 10000 -> 20000 nano giây.

Bài 3: Tìm min, max bằng quick sort

* Sử dụng vòng lặp for để tạo ra các mảng với kích cỡ khác nhau, giá trị của các phần tử trong được sinh ngẫu nhiên. Ta có bảng kết quả:



* Sử dụng phần mềm **Rstudio** để vẽ biểu đồ ta thu được biểu đồ sau:



- Qua biểu đồ, ta nhận thấy thời gian thực hiện bài toán tăng lên khá đáng kể nếu n lớn

PHẦN 4: ĐẶT BÀI TOÁN, PHÂN TÍCH, THIẾT KẾ THUẬT TOÁN VÀ PHÂN TÍCH THUẬT TOÁN

BÀI TOÁN 1: CHO 1 TẬP HỢP N ĐIỂM TRONG MẶT PHẲNG TOẠ ĐỘ, TÌM CẶP ĐIỂM CÓ KHOẢNG CÁCH NHỎ NHẤT

1. Phân tích bài toán
2. Xác định input và ouput

* Input: Tập hợp n điểm trong mặt phẳng tạo độ Oxy
* Output: Tìm ra cặp điểm có khoảng cách nhỏ nhất

1. Giải pháp

* Sử dụng phương pháp chia để trị để giải bài toán
* Khi đó ta cần sắp xếp các điểm theo chiều tăng (nếu chưa được sắp xếp ta cần được sắp xếp lại theo quicksort hoặc merge để độ phức tạp về thời gian chỉ là O(nlogn)), sau đó chia các điểm cần tìm thành 2 phần bằng đường thẳng x = m
* Khi đó cặp điểm (hay khoảng cách) cần tìm chính là khoảng cách nhỏ nhất trong 2 phần và các điểm thuộc lân cận của đường thẳng x = m

1. Phát biểu lại bài toán

Input:

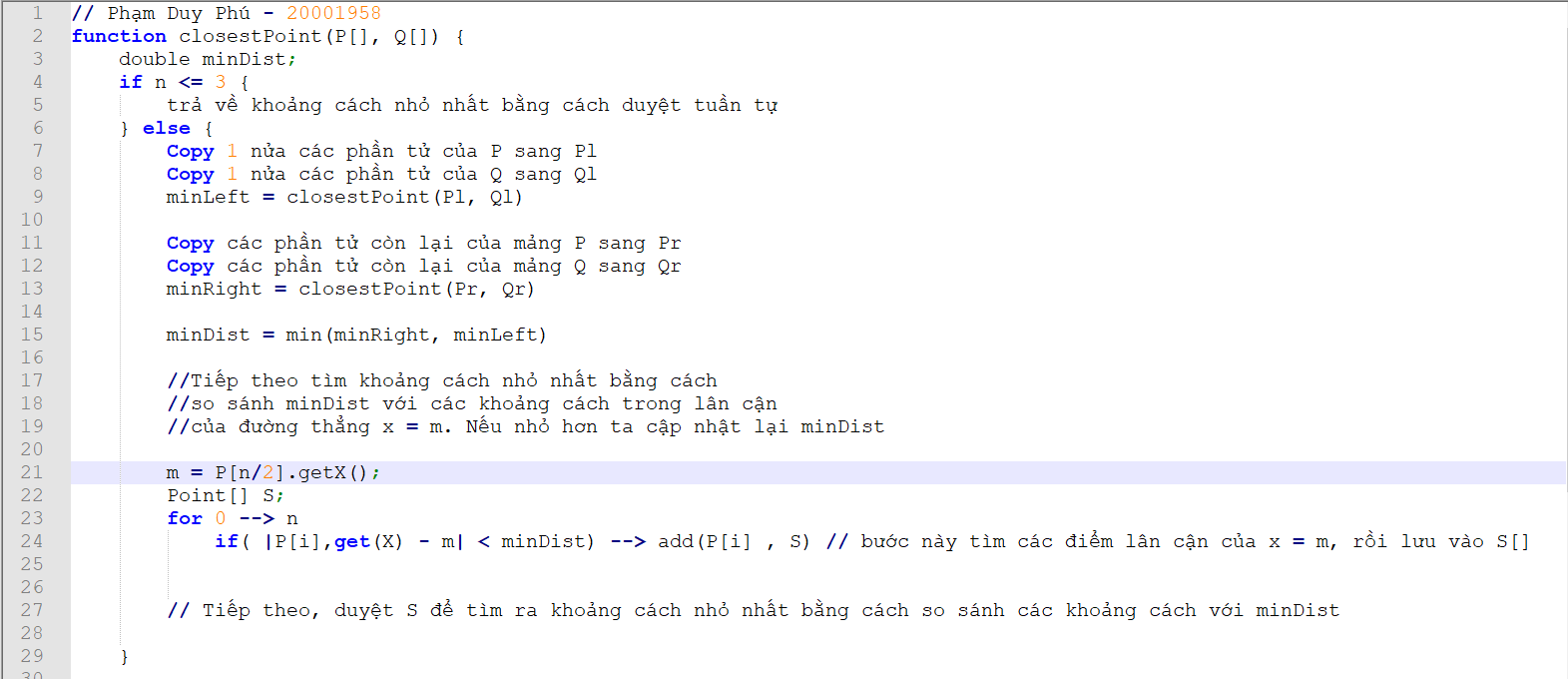
* Mảng P gồm các điểm được sắp xếp theo toạ độ x
* Mảng Q gồm các điểm được sắp xếp theo toạ độ y

Output:

* Khoảng cách ngắn nhất của các cặp điểm

1. Xây dựng thuật toán

Pseudo code:



1. Phân tích thuật toán

Độ phức tạp:

* Bài toán được giải quyết bằng cách chia số điểm ban đầu thành 2 phần, sau đó tìm min trong 2 phần và các điểm lân cận của m.
* Thời gian thực hiện của bài toán phụ thuộc vào thời gian giải 2 bài toán con + thời gian kết hợp các lời giải
* Thời gian thực hiện 2 bài toán con: 2T(n/2), thời gian kết hợp các lời giải: f(n)

Áp dụng công thức Master:

BÀI TOÁN 2: CHO N ĐỒNG XU, TRONG ĐÓ CÓ 1 ĐỒNG XU LÀ GIẢ. TÌM VỊ TRÍ CỦA ĐỒNG XU GIẢ ĐÓ

1. PHÂN TÍCH BÀI TOÁN
2. Xác định input và output

* Input: N đồng xu, biết rằng có 1 đồng xu là giả
* Output: Tìm vị trí của đồng xu giả

1. Giải pháp

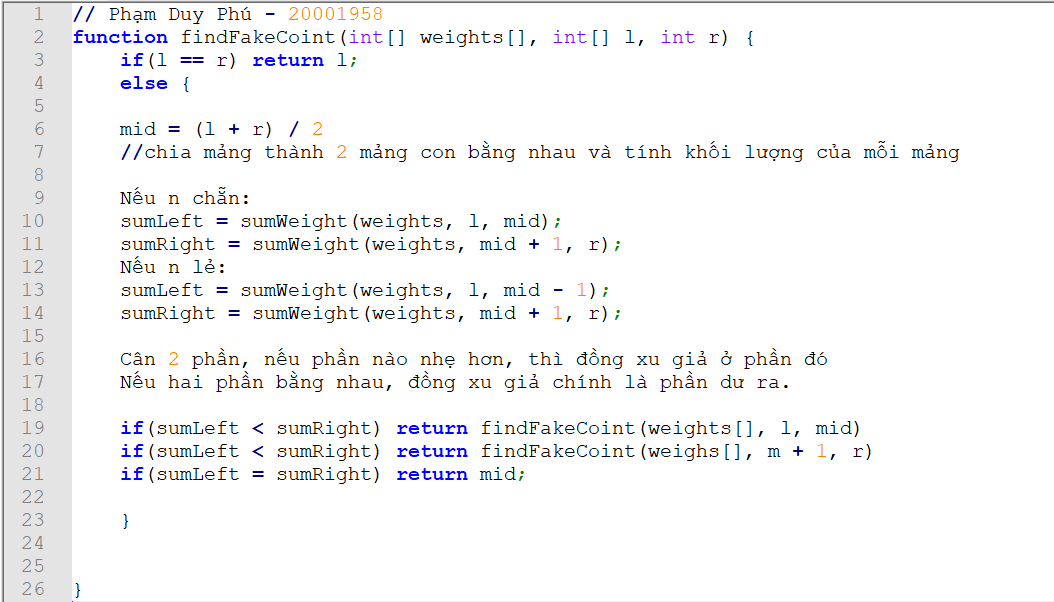
* Sử dụng phương pháp giảm để trị (mỗi lần tìm, ta loại bỏ được 1 nửa các phần tử) để giải quyết bài toán bằng cách chia n đồng xu thành 2 phần bằng nhau
* Cân 2 phần, nếu phần nào nhẹ hơn, tức đồng xu giả nằm ở phần đó, ta tiếp tục chia đôi phần nhẹ hơn để tìm ra đồng xu giả
* Trong trường hợp 2 phần bằng nhau(xảy ra khi n lẻ) thì đồng xu giả chính là đồng xu lẻ ra.

1. Đặt lại bài toán:

* Để thuận tiên, ta biểu diễn n đồng xu là 1 mảng weights gồm n phần tử
* Trong đó, phần tử có giá trị bằng 1 là đồng xu thật, phần tử giá trị bằng 0 là đồng xu giả
* Input: weighs[], weighs[i] = 0, 1, có 1 phần tử bằng 0
* Ouput: Vị trí của phần tử 0

1. Xây dựng thuật toán

* Pseudo code:



1. Phân tích thuật toán

* Thời gian thực hiện thuật toán bằng thời gian thực hiện 1 trong 2 phần của bài toán con + thời gian thực hiện trường hợp cơ sở