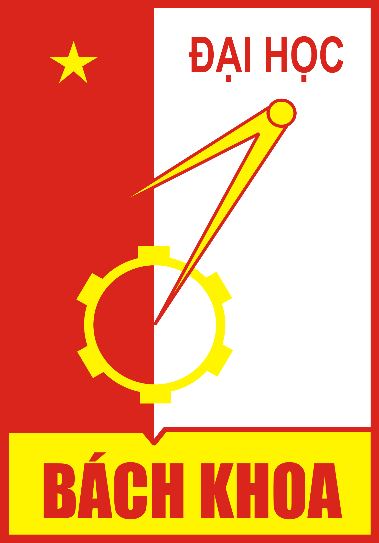
**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**



**VIỆN CÔNG NGHỆ THÔNG TIN – TRUYỀN THÔNG**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

****

**BÁO CÁO MÔN HỌC  
PROJECT II**

**ĐỀ TÀI**

**Tìm hiểu và cài đặt bài toán tìm cặp điểm gần nhất**

**(Closest Pair)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sinh viên thực hiện:** | | | |
| **HỌ VÀ TÊN** | **MSSV** | **LỚP** | **KHÓA** |
| **NGUYỄN TẤT HÒA** | **20131536** | **CNTT2.01** | **K58** |

**Giảng viên hướng dẫn: PGS TS. Nguyễn Đức Nghĩa**

**HÀ NỘI, THÁNG 4/2016**

MỤC LỤC

[**Lời mở đầu** 3](#_Toc449048311)

[**Phần 1: Giới thiệu bài toán** 4](#_Toc449048312)

[**1.** **Phát biểu bài toán** 4](#_Toc449048313)

[**2.** **Lịch sử giải quyết bài toán.** 4](#_Toc449048314)

[**Phần 2: Các thuật toán** 4](#_Toc449048315)

[**1.** **Giải thuật vét cạn.** 4](#_Toc449048316)

[**2.** **Giải thuật chia để trị.** 6](#_Toc449048317)

[**Phần 3: Thực nghiệm.** 10](#_Toc449048318)

[**1. Phương pháp sinh dữ liệu.** 10](#_Toc449048319)

[**2. Tiêu chí đánh giá** 10](#_Toc449048320)

[**3.** **Cấu hình máy thực nghiệm** 11](#_Toc449048321)

[**4.** **Kết quả so sánh.** 11](#_Toc449048322)

[**5.** **Hướng dẫn cài đặt và sử dụng chương trình.** 13](#_Toc449048323)

[**Phần 4: Chương trình nguồn** 15](#_Toc449048324)

[**1.** **Source Code** 15](#_Toc449048325)

[**2.** **Các chương trình con chính.** 15](#_Toc449048326)

[**Phần 5: Tài liệu tham khảo** 17](#_Toc449048327)

# **Lời mở đầu**

Thời đại hiện nay, công nghệ thông tin càng ngày càng phát triển, trong đó Khoa học máy tính là một trong những bộ môn quan trọng trong sự phát triển đó. Khoa học máy tính là nền tảng của mọi lĩnh vực liên quan đến CNTT, trong đó có Công nghệ Phần mềm. Ngành này giúp người học nắm vững kiến thức thuật toán, hiểu biết đầy đủ về các lĩnh vực đa dạng của máy tính.

Tìm cặp điểm gần nhất là một bài toán kinh điển trong khoa học máy tính. Hiện nay, có nhiều thuật toán để giải bài toán này. Trong bài báo cáo này sẽ trình bày hai thuật toán cơ bản để giải quyết vần đề đó là phương pháp vét cạn và phương pháp chia để trị, với từng phương pháp, bản báo cáo sẽ trình bày rõ ý tưởng, giải thuật, cài đặt giải thuật chi tiết, đưa ra được thời gian để giải quyết vấn đề trong từng trường hợp cụ thể cũng như chứng minh tính đúng đắn của thuật toán. Cuối cùng đưa ra đánh giá, kết luận về tính hiệu quả và tính tối ưu để giải quyết được bài toán tìm cặp điểm gần nhất.

Do kiến thức còn hạn hẹp, chắc hẳn bản báo cáo vẫn còn những sai sót và hạn chế vì vậy em mong nhận được sự góp ý, hướng dẫn và sửa chữa để bản báo cáo được dần dần hoàn thiện hơn.

Em xin chân thành cảm ơn PGS TS Nguyễn Đức Nghĩa đã giúp em có được những kiến thức cơ bản trong quá trình hướng dẫn để hoàn thành được bản báo cáo này.

Hà Nội, ngày 1 tháng 4 năm 2016.

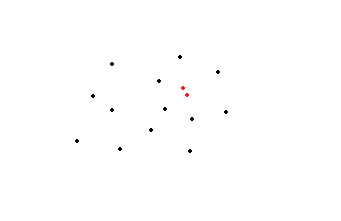
Sinh viên thực hiện:

Nguyễn Tất Hòa

## **Phần 1: Giới thiệu bài toán**

1. **Phát biểu bài toán**

Các cặp điểm gần nhất là một vấn đề của hình học tính toán và là một trong những bài toán kinh điển của bộ môn khoa học máy tính: Cho tập P gồm n điểm trong một không gian d chiều, tìm một cặp điểm thuộc n điểm đó sao cho khoảng cách giữa hai điểm đó là ngắn nhất. Trong bản báo cáo này sẽ tập trung phát biểu và cài đặt thuật toán trên không gian hai chiều.

1. **Lịch sử giải quyết bài toán.**
   * Giải thuật vét cạn: giải thuật này không có tác giả. Độ phức tạp của giải thuật này là O(n2).
   * Giải thuật chia để trị: giải thuật này do Bentley và Shamos trình bày năm 1976. Độ phức tạp của giải thuật là O(*n*log*n*).
   * Giải thuật cải tiến từ Chia để trị của Bentley và Shamos: giải thuật này do Mohammad Zaidul Karim và Nargis Akter đề xuất. Giải thuật này chia bài toán thành n phần thay vì 2 phần. Giải thuật này trong nhiều trường hợp có độ phức tạp nhỏ hơn O(n log n).

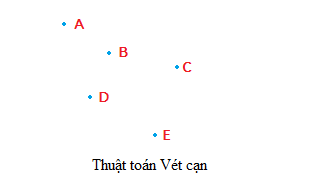
## **Phần 2: Các thuật toán**

1. **Giải thuật vét cạn.**
   1. **Ý tưởng.**

Ý tưởng giải thuật vét cạn gồm các bước cụ thể như sau:

* + - Tính tất cả các khoảng cách giữa các điểm trong tập P.
    - Xác định khoảng cách có giá trị nhỏ nhất trong số các khoảng cách vừa tìm được.
    - Cặp điểm có khoảng cách ngắn nhất là cặp điểm gần nhất.

Ví dụ:



Trong hình trên gồm một tập 5 điểm, chúng ta cần tính tất cả khoảng cách có thế có giữa năm điểm ấy. Bắt đầu từ A có 4 khoảng cách với 4 điểm B, C, D, E. Từ điểm B có 3 khoảng cách tới điểm C, D, E do khoảng cách tới điểm A đã tính trước đó. Tương tự như vậy với các điểm C, D, E vậy có tất cả 4 + 3 + 2 + 1 = 10 khoảng cách cần tính toán. Tổng quát với n điểm thì tổng số phép tính khoảng cách giữa các điểm đó là

* 1. **Giải thuật**

Thuật toán tìm cặp điểm gần nhất bằng vét cạn:

1. **procedure** **BruteForce**(P, n)

2. *min* = ∞

3. **for** i = 1 to n − 1 **do**

4. **for** j = i + 1 to n **do**

5. d = **dist**(Pi, Pj)

6. **if** (d < *min*) **then**

7. *min* = d

8. **closestPair** = (Pi, Pj)

9. **end if**

10. **end for**

11. **end for**

12. **return** closestPair

13. **end procedure**

* 1. **Độ phức tạp.**

Độ phức tạp của thuật toán được tính như sau:

* + - Độ phức tạp của chương trình con dist (phép tính căn và bình phương) là O(1).
    - Độ phức tạp của các lệnh 2, 4, 6, 7, 8 là O(1).
    - Độ phức tạp của lệnh 4 là O(n-i).
    - Độ phức tạp của lệnh 3 sẽ là O(n­­­2­­­) như đã tính toán phía trên

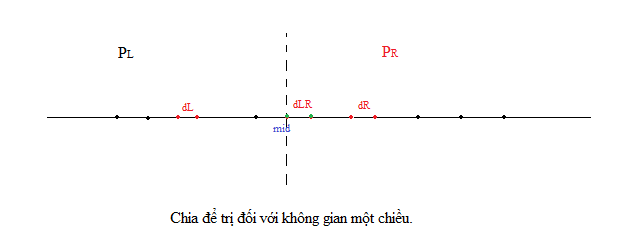
Vậy độ phức tạp của thuật toán vét cạn cho bài toán tìm cặp điểm gần nhất là T(n) = O(n­­2). Vì thuật toán chỉ đơn giản là xét duyệt tất cả các điểm và khoảng cách nên việc cài đặt là như nhau đối với mọi không gian.

1. **Giải thuật chia để trị.**
   1. **Đối với không gian một chiều**

Ý tưởng của giải thuật chia để trị gồm các bước như sau:

* + - Chia tập P thành 2 tập con có số lượng phần tử bằng nhau. Điểm ở giữa là pmid
    - Tìm cặp điểm có khoảng cách nhỏ nhất trong nửa trái P­­L­ là d­L.
    - Tìm cặp điểm có khoảng cách nhỏ nhất trong nửa phải P­­R­ là d­R.
    - ­Tìm khoảng cách nhỏ nhất đối với các điểm xung quanh vị trí p­mid­ gọi là d­mid­.
    - So sánh d­L­, d­R­, d­mid­ sẽ tìm ra được cặp điểm có khoảng cách nhỏ nhất.
  + Dùng đệ quy để tìm được hai điểm có khoảng cách nhỏ nhất thuộc mỗi tập con bên trái và tập con bên phải. Từ đó ta tìm được d­L­. d­R­.  
     Còn d­mid­ giả thiết rằng d­mid­ là khoảng cách nhỏ nhất cần tìm thì chắc chắn trong hai điểm tìm được một điểm sẽ thuộc P­L­ và một điểm sẽ thuộc P­R­ vì nếu vi phạm điều trên thì mẫu thuẫn với d­L­, d­R­ là khoảng cách nhỏ nhất của mỗi nữa trái, phải. Chính vì vậy nên chỉ cần xét khoảng cách giữa hai điểm gần phải nhất đối với tập P­L­ và gần trái nhất đối với tập P­R­ gọi đó là d­mid­.

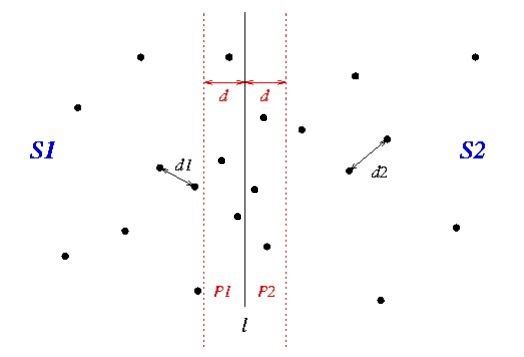
Cuối cùng so sánh d­L­, d­R­, d­mid­ rồi đưa ra hai điểm có khoảng cách nhỏ nhất thuộc tập P. Do trong bản báo cáo này tập trung vào không gian hai chiều nên sẽ không đi sâu vào việc cài đặt thuật toán cũng như so sánh hiệu năng của giải thuật trên không gian một chiều.



* 1. **Đối với không gian hai chiều.**

1. **Ý tưởng.**

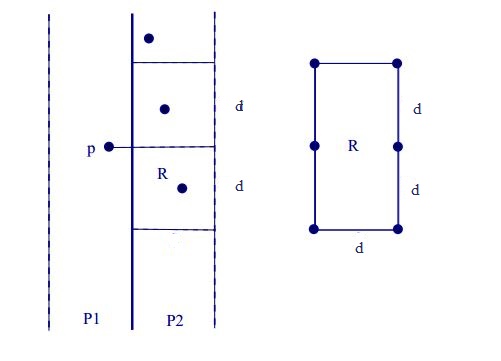
Ý tưởng của thuật toán hoàn toàn tương tự như trong không gian một chiều nhưng việc tìm được khoảng cách d­mid­ lại không hề đơn giản bởi lúc này không phải chỉ có hai điểm cần được xét.



Trên thực tế, trong không gian hai chiều, những điểm xung quanh p­mid­ có thể là tất cả các điểm trong tập hợp đang xét dẫn đến độ phức tạp của bài toán sẽ là O(n2) mà mục tiêu của thuật toán là O(*n*log*n*).

Chú ý rằng với mỗi cặp ứng cử viên (p, q) xung quanh p­mid với p S1 và q S2 thì p, q phải nằm trong khoảng cách *l (l* là giải phân cách đi qua điểm chính giữa p­mid ) một đoạn là d kéo về hai phía với d = min(dL, dR). Như đã nói ở trên thì số điểm cần xét đến có thể là tất cả các điểm.

Và thật may mắn ta có một cách để giải quyết vấn đề này. Xét mỗi điểm p S1, tất cả các điểm thuộc S2 mà cách l một khoảng bằng d phải nằm trong một hình chữ nhật R có kích thước d x 2d. Bởi vì những điểm mà nằm bên ngoài hình chữ nhật R sẽ không thể đạt được khoảng cách tới p mà lại nhỏ hơn d được.



Trong không gian hai chiều thì số điểm thỏa mãn điều kiện đó nhiều nhất là 6 điểm. Do đó chúng ta chỉ cần thực hiện nhiều nhất là 6 x n phép so sánh khoảng cách chứ không còn phải n2. Tất cả những gì chúng ta phải làm là sắp xếp những điểm xung quanh p­mid lại theo chiều tăng của tung độ và kiểm tra các điểm đó theo thứ tự với nhiều nhất sáu điểm láng giềng. Thuật toán sắp xếp cùng cần có độ phức tạp là O(*n*log*n*), trong bài này thuật toán được lựa chọn là Merge Sort.

* 1. **­Giải thuật.**  
     Thuật toán tìm cặp điểm gần nhất bằng cách **chia để trị**:

1**. proceduce ClosestPair**(P, n){

2. **if** n<5 **then**

3. **return** **bruteForce**(P,n)

4. **end** **if**

5. *mid* = n/2; midPoint = P[*mid*];

6. dL = **ClosestPair**(P,mid); dR = **ClosestPair**(P+mid, n-mid);

7. d = **min**(dL, dR);

8. pLR = { P | abs(p[i].x – midPoint.x) <= d};

9. **return** **min**(**stripClosest**(pLR, pLR.count, d), d);

10.**end** **proceduce.**

Thuật toán tìm khoảng cách nhỏ nhất của tập pLR:

1. **proceduce stripClosest**(pLR, k, d){
2. *min* = d;
3. **for** i: 1 to k **do**
4. **for** j: i+1 to k **&&** pLR[j].y - pLR[i].y < min **do**
5. **if** (**dist**(pLR[i],pLR[j]) < min) **then**
6. *min* = **dist**(pLR[i],pLR[j]);
7. **if** (*min* < dmin) **then**
8. dmin = *min*;
9. Closest Pair = (pLR[i], pLR[j])
10. **end if**
11. **end if**
12. **end for**
13. **end for**
14. **return** *min*
15. **end proceduce**
    1. **Độ phức tạp.**

Độ phức tạp của thuật toán chia để trị ở từng bước sẽ là:

* + - Bước 2: O(1)
    - Bước 3: O(25)
    - Bước 5: O(n)
    - Bước 6: O(*n*log*n*)
    - Bước 8: O(n)
    - Bước 9: O(n) (như đã chứng minh phía trên)

Do ở việc sắp xếp ở bước 9 độ phức tạp là O(*n*log*n*) và điều này được lặp lại ở mỗi lần đệ quy nên độ phức tạp của thuật toán sẽ là O(*n*log­2*n*). Ta có thể giải quyết vấn đề này để giải thuật có thể đạt được độ phức tạp O(*n*log*n*) bằng cách sắp xếp theo chiều tăng của tọa độ y ngay từ đầu chương trình thay vì sắp xếp trong hàm **stripClosest** như đã nói ở trên.

Cuối cùng thuật toán đã đạt được độ phức tạp là O(*n*log*n*).

## **Phần 3: Thực nghiệm.**

Theo lí thuyết thì khi giải bài toán tìm cặp điểm gần nhất thì thuật toán vét cạn có độ phức tạp là O(n­­2) còn thuật toán chia để trị thì có độ phức tạp là O(*n*log*n*), tức là thuật toán Chia để trị tốt hơn thuật toán Vét cạn. Như vậy, cần cài đặt hai giải thuật Vét cạn và Chia để trị . Áp dụng hai giải thuật này lên nhiều tập dữ liệu khác nhau với các tiêu chí đánh giá hợp lý nhằm kiểm tra tính đúng đắn của lý thuyết.

### **1. Phương pháp sinh dữ liệu.**

* + Sinh ngẫu nhiên tập P
  + Các điểm trong cùng một tập không được trùng nhau
  + Sinh đủ số lượng yêu cầu
  + Các điểm trong mỗi lần sinh ra là khác nhau

Cách sinh dữ liệu để không tồn tại 2 điểm trùng nhau trong tập:

Dùng một mảng đủ lớn (kích thước bằng với kích thước của tập P) để lưu giá trị của tất cả các điểm đã được tạo ra trước đó. Sau mỗi lần sinh ngẫu nhiên một điểm mới kiểm tra xem điểm đó đã tồn tại chưa nếu tồn tại thì tiếp tục sinh ngẫu nhiên cho đến khi không tồn tại thì lưu lại vào tập P. Thuật toán kết thúc khi ghi đủ số lượng phần tử.

Thuật toán:

1. **proceduce** **generatePoint**(a[], n)
2. Open file point.txt
3. Write value n to file point.txt in first line
4. x = *random*(10000); y = *random*(100000);
5. **for** i: 1 to n **do**
6. **while** exist in a[] **do**
7. x = random(10000); y = random(100000);
8. **end** **while**
9. Write x, y to file point in a line;
10. **end** **for**
11. **end** **proceduce**

Do x, y lần lượt là hoành độ và tung độ của điểm trong P và được sinh ngẫu nhiên trong khoảng (0,100000) nên độ lớn của không gian dữ liệu có thể lên tới 1010 điểm, đủ lớn để có thể thấy được sự khác biệt giữa hai thuật toán

### **2. Tiêu chí đánh giá**

* Kết quả của hai thuật toán phải giống nhau trong các trường hợp
* Trong đa số trường hợp thì thuật toán Chia để trị có thời gian chạy máy ít hơn Vét cạn.
* Khi dữ liệu lớn dần thì cả hai thuật toán đều có thời gian chạy máy tăng lên, nhưng thuật toán Vét cạn có thời gian chạy máy tăng rất nhanh (O(n2) tức là cho kết quả chậm khi dữ liệu lớn), còn chia để trị thì tăng không đáng kể (O(*n*log*n*)).

1. **Cấu hình máy thực nghiệm**

Trong quá trình cài đặt và thực nghiệm có sử dụng máy tính với các thông số như sau:

* CPU: Intel, Core i5, 4200U, 1.60 GHz
* RAM: DDR3L 8GB 1600 MHz
* Ổ cứng: HDD, 500GB

### **Kết quả so sánh.**

Sau đây là kết quả thực nghiệm khi chạy chương trình với trục tung có đơn vị là giây(s) và trục hoành có đơn vị là điểm.

Do thời gian giải quyết bài toán của thuật toán Vét cạn tăng rất nhanh theo sự tăng của không gian dữ liệu nên bên dưới sẽ chỉ trình bày biểu đồ thuật toán Chia để trị với không gian dữ liệu lớn.

**Kết luận:**

* Về lý thuyết:
  + Brute Force: O(n­­2).
  + Devide and Conquer: O(*n*log*n*)
* Về thực nghiệm: dựa vào kết quả đồ thị.
  + Khi tập dữ liệu nhỏ thì thời gian chạy của hai thuật toán không chênh lệch, nhưng khi dữ liệu đủ lớn thì giải thuật Chia để trị chạy nhanh hơn giải thuật Vét cạn.
  + Khi tập dữ liệu lớn dần thì cả hai giải thuật đều có thời gian chạy máy tăng nhưng:
    - Vét cạn: tăng rất nhanh.
    - Chia để trị: tăng rất chậm.
    - Các biểu đồ thứ 2 và thứ 3 thể hiện rất rõ điều đó
    - Khi tập điểm là bội số của 10000 thì thuật toán Vét cạn cho thời gian trả về kết quả là rất lâu lên tới tầm bảy phút khi tập P có 100000 điểm, cũng với tập đó thì thuật toán chia để trị trả về kết quả khi chưa đến 0.12(s).

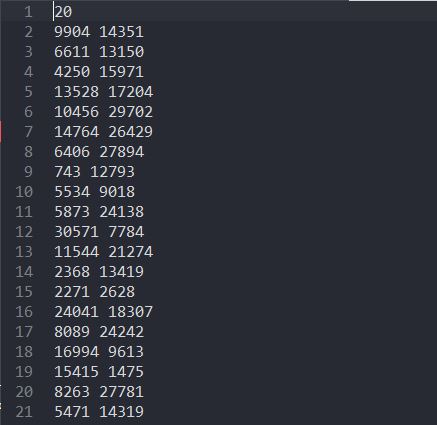
Từ các tính toán trên lý thuyết và thực tế cho thấy để giải quyết bài toán Tìm cặp điểm gần nhất thì thuật toán Chia để trị đạt được hiệu năng cao hơn thuật toán Vét cạn. Trong một không gian dữ liệu đủ lớn việc lựa chọn thuật toán Chia để trị để giải quyết bài toán tìm cặp điểm gần nhất là hợp lí.

1. **Hướng dẫn cài đặt và sử dụng chương trình.**

Chương trình giải quyết bài toán tìm cặp điểm gần nhất được viết bởi ngôn ngữ lập trình C hoàn toàn dể sử dụng đối với những người chưa bao giờ lập trình, để chạy được chương trình chỉ cần bất cứ chương trình có thể biên dịch và chạy ngôn ngữ C.

Đầu vào sẽ được đọc từ file point.txt được lựu trong cùng một thư mục với chương trình, chứa vị trí các điểm trong không gian có cấu trúc như sau:

* + - Dòng đầu là số các điểm có trong tập P
    - Các dòng tiếp theo là vị trí theo tọa độ (x, y) của các điểm trong đó x là hoành độ, y là tung độ.



Khi chạy chương trình sẽ có hai lựa chọn đó là giải quyết bài toán bằng thuật toán Vét cạn hoặc Chia để trị. Chọn một trong hai để thực hiện, kết quả trả về là khoảng cách và hai điểm gần nhất trên màn hình máy tính, thời gian sẽ được lưu trong file result.txt với thứ tự lần lượt là Vét cạn, Chia để trị và không gian dữ liệu.

## **Phần 4: Chương trình nguồn**

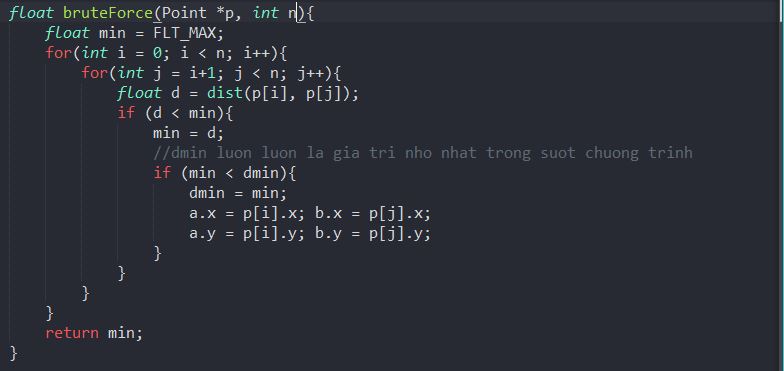
1. **Source Code**

Toàn bộ source code được lưu trên github:

**https://github.com/kevinhoa95/closestpair**

1. **Các chương trình con chính.**
   1. **bruteForce()**

Hàm sử dụng thuật toán vét cạn để tìm ra cặp điểm có khoảng cách nhỏ nhất trong tập không gian dữ liệu.



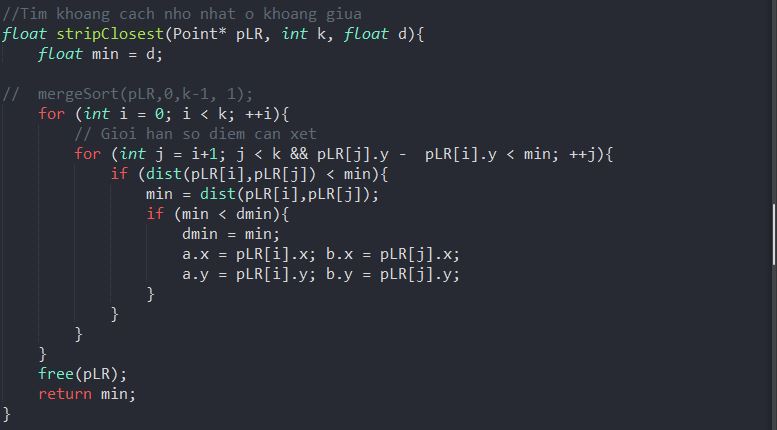
* 1. **generatePoint()**



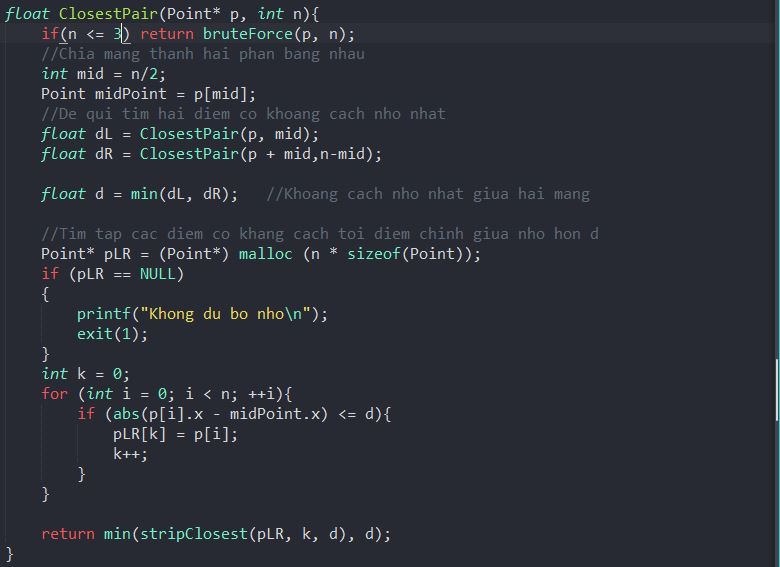
Hàm sinh ngẫu nhiên tập không gian để kiểm chứng hiệu năng của từng thuật toán.

* 1. **stripClosest()**

Hàm tìm và so sánh hai khoảng cách đệ quy với khoảng cách giữa các điểm xung quanh vị trí p­mid­ để phục vụ cho hàm ClosestPair().



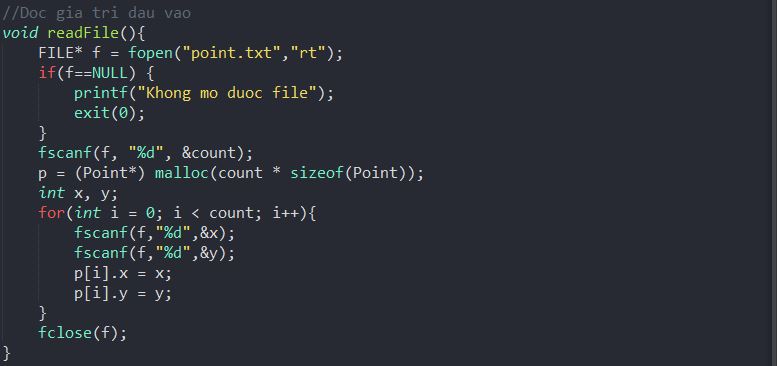
* 1. **ClosestPair()**



Hàm chính để tìm hai cặp điểm có khoảng cách nhỏ nhất theo thuật toán chia để trị.

* 1. **readFile()**

Hàm đọc dữ liệu đầu vào trong chương trình. Thực hiện đọc và gán các điểm cho mảng được cấp phát động.



## **Phần 5: Tài liệu tham khảo**

Trong quá trình tìm hiểu và cài đặt thuật toán em có tham khảo một số tài liệu sau:

* Tài liệu của PGS TS Nguyễn Đức Nghĩa.
* Slide bài giảng môn học “Cấu trúc dữ liệu và giải thuật” của thầy Phạm Quang Dũng trường DHBKHN.
* Cuốn sách “Algorithm Design” của Jon KleinBerg.

Và một số tài liệu trên Internet như:

* <https://www.cs.ucsb.edu/~suri/cs235/ClosestPair.pdf>
* <http://www.geeksforgeeks.org/closest-pair-of-points>
* <https://rosettacode.org/wiki/Closest-pair_problem>