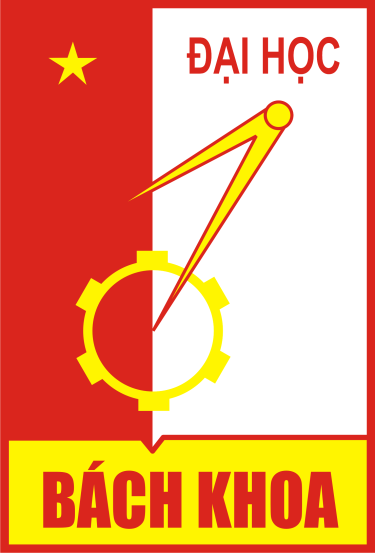
**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

──────── \* ───────



**BÁO CÁO KĨ THUẬT LẬP TRÌNH**

***ĐỀ TÀI:* BÀI TOÁN ĐẢO NGỮ**

**GVHD: ThS. Bùi Thị Mai Anh**

**Nhóm thực hiện: Nhóm 28**

**Các thành viên**:

|  |  |  |  |
| --- | --- | --- | --- |
| **1.** | **ĐỒNG QUANG LINH** | **20162385** | **CNTT1.01 - K61** |
| **2.** | **ĐỖ MINH HIẾU** | **20161505** | **CNTT1.01 - K61** |
| **3.** | **NGUYỄN TIẾN QUANG** | **20163312** | **CNTT1.01 - K61** |

MỤC LỤC

[**PHẦN MỞ ĐẦU** 2](#_Toc512692998)

[**I.** **MỤC ĐÍCH CỦA ĐỀ TÀI :** 2](#_Toc512692999)

[**II.** **SỰ ĐÓNG GÓP CỦA CÁC THÀNH VIÊN** 2](#_Toc512693000)

[**GIỚI THIỆU BÀI TOÁN** 3](#_Toc512693001)

[**PHÂN TÍCH BÀI TOÁN** 3](#_Toc512693002)

[ **GIẢI THUẬT** 3](#_Toc512693003)

[**a.** **Giải thuật checking off** 4](#_Toc512693004)

[**b.** **Giải thuật sắp xếp và so sánh** 4](#_Toc512693005)

[**c.** **Giải thuật đếm và so sánh** 4](#_Toc512693006)

[**d.** **Giải thuật vét cạn** 4](#_Toc512693007)

[**THIẾT KẾ CHƯƠNG TRÌNH** 4](#_Toc512693008)

[**1.** **Checking off** 5](#_Toc512693009)

[**2.** **SẮP XẾP VÀ SO SÁNH** 6](#_Toc512693010)

[**3.** **VÉT CẠN** 7](#_Toc512693011)

[**4.** **ĐẾM VÀ SO SÁNH** 8](#_Toc512693012)

[**ĐÁNH GIÁ** 9](#_Toc512693013)

[**I.** **So sánh** 9](#_Toc512693014)

[**II.** **Phân tích** 9](#_Toc512693015)

[**1.** **Checkoff** 10](#_Toc512693016)

[**2.** **Sắp xếp và so sánh** 10](#_Toc512693017)

[**3.** **Vét cạn** 10](#_Toc512693018)

[**4.** **Đếm và so sánh** 11](#_Toc512693019)

[**KẾT LUẬN** 11](#_Toc512693020)

# **PHẦN MỞ ĐẦU**

## **MỤC ĐÍCH CỦA ĐỀ TÀI :**

* Nghiên cứu độ hiệu quả của các thuật toán trong bài toán xâu chữ đảo ( thời gian xử lý, bộ nhớ sử dụng). Từ đó đưa ra thuật toán tốt nhất để giải quyết bài toán .

## **SỰ ĐÓNG GÓP CỦA CÁC THÀNH VIÊN**

|  |  |  |
| --- | --- | --- |
| Họ Và Tên | MSV | Mức độ đóng góp |
| Đỗ Minh Hiếu | 20161505 | 40.0% |
| Đồng Quang Linh | 20162385 | 40.0% |
| Nguyễn Tiến Quang | 20163312 | 20.0% |

# **GIỚI THIỆU BÀI TOÁN**

* Trò đảo chữ là trò đảo kí tự trong một từ hoặc một câu để tạo thành câu mới, từ mới có nghĩa(trong tiếng Anh). Ví dụ: "heart" và "earth". Hoặc: "O Draconian Devil" là đảo chữ của "LeonardoDa Vincy" (Nguồn "Mật mã Da Vincy").
* Problem: xác định hai chuỗi kí tự input có phải là đảo chữ của nhau hay không?

# **PHÂN TÍCH BÀI TOÁN**

* Với hai xâu dữ liệu đầu vào cần đối chiếu một xâu với xâu còn lại. Nếu tất cả các phần tử đều trùng khớp chứng tỏ 2 xâu là xâu đảo của nhau. Khi đối chiếu cần bỏ qua khoảng trắng và không cần phân biệt chữ hoa và chữ thường.
* *Input* : Hai xâu kí tự.

*Output* : Nếu là xâu đảo thì hiển thị “ là xâu đảo”. Nếu không hiển thị “ không là xâu đảo lên màn hình”.

## **GIẢI THUẬT**

### **Giải thuật checking off**

* Nếu mỗi kí tự trọng chuỗi thứ nhất xuất hiện trong chuỗi thứ 2 thì đánh dấu kí tự đótrong chuỗi thứ 2 (check off). Nếu mọi kí tự trong chuỗi thứ nhất đều được checkoff thì hai chuỗilà đảo chữ của nhau. Có thể bỏ qua các kí tự trắng. Vì chúng ta xét bài toán tổng quát với một câu bất kì.

### **Giải thuật sắp xếp và so sánh**

* Hai chuỗi là đảo chữ của nhau nếu nó chứa cùng tập hợp các kí tự (loại trừ kí tự trắng).Vì vậy chúng ta có thể sắp xếp các kí tự của cả hai chuỗi từ a-z, nếu chúng ta cùng thu được một chuỗi thì hai chuỗi ban đầu là đảo chữ.

### **Giải thuật đếm và so sánh**

* Kĩ thuật vét cạn khi giải quyết một bài toán dựa trên ý tưởng là vét toàn bộ khả năng cóthể xảy ra. Đối với bài toán đảo chữ, ý tưởng vét cạn là với chuỗi s1, chúng ta đảo các kí tự của nó đến khi chúng ta tìm được s2.

### **Giải thuật vét cạn**

* Hai chuỗi là đảo chữ nếu nó có cùng số kí tự a, cùng số kí tự b, etc. Vì chỉ có 26 kí tự trong tiếng Anh, chúng ta có thể tạo ra một list có 26 counter. Với mỗi chuỗi, chúng ta tăng giá trị của 1 counter với kí tự tương ứng. Cuối cùng nếu có 2 list counter giống nhau thì kết luận hai chuỗi ban đầu là đảo chữ.

# **THIẾT KẾ CHƯƠNG TRÌNH**

* **Pseudocode hàm main() :**

*int main(){*

*<Đọc dữ liệu từ file lưu vào mảng>; // Bước tiền xử lý*

*if( <giải thuật>){*

*printf(“ là xâu đảo”);*

*}*

*else{*

*printf(“ không phải xâu đảo”);*

*}*

* **Bước tiền xử lý**

* Việc đọc dữ liệu từ file vào mảng đòi hỏi ta phải cấp phát một bộ nhớ động phù hợp với kích thước của dữ liệu. Do trong dữ liệu đầu vào số khoảng trắng là không lớn nên ta có thể dùng hàm ftell trong thư viện stdio.h ( hàm trả về size của file kể cả khoảng trắng) để cấp một bộ nhớ phù hợp với kích thước dữ liệu đầu vào.
* Xây dựng hàm readData đọc những kí tự là chữ cái trong file đầu vào và lưu ra mảng đồng thời chuyển các kí tự là chữ thường về chữ hoa và trả về số kí tự có nghĩa
* **Pseudocode hàm readData**

*int readData(){*

*char ch;*

*int counter = 0;*

*while(1){*

*< đọc một kí tự và lưu vào biến ch>;*

*if ( ch == EOF ) break;*

*if ( ch là kí tự chữ cái ){*

*p[i] = ch;*

*counteri++;*

*}*

*}*

*Return counter;*

*}*

* **CHI TIẾT CÁC GIẢI THUẬT**

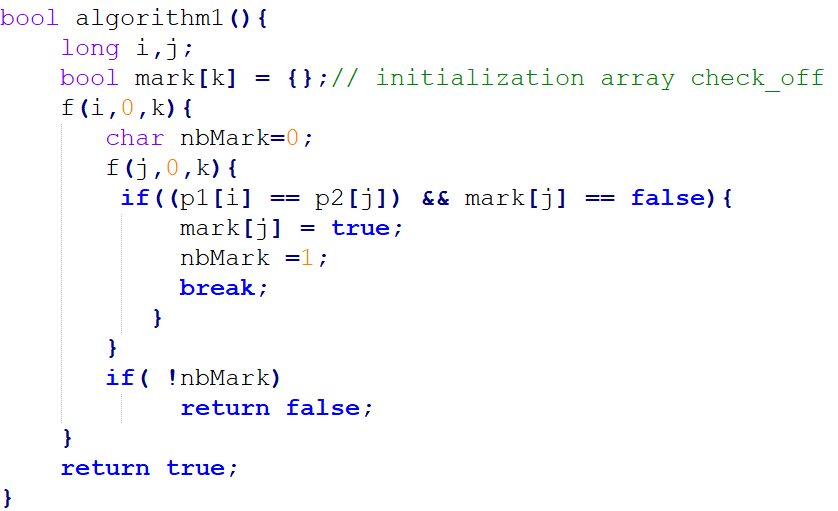
## **Checking off**

1. *Ý tưởng :*

* Nếu mỗi ký tự trong chuỗi thứ nhất xuất hiện trong chuỗi thứ 2 thì đánh dấu kí tự đó trong chuỗi 2. Sau khi duyệt nếu tất cả các kí tự trong chuỗi 2 đều đc đánh dấu thì kết luận hai xâu là xâu đảo của nhau.

1. *Giai pháp :*

* Khởi tạo một mảng đánh dấu mark[] với các phần tử là false ( các phần tử trong chuỗi 2 chưa được đánh dấu).
* Dùng hai vòng lặp for lồng nhau. Duyệt qua từng phần tử của chuỗi 2. Đánh dấu mark[i] = true nếu p[i] là phần tử trong xâu thứ 2 xuất hiện trong xâu thứ nhất.
* Dùng biến flag khởi tạo với giá trị bằng 0 ở mỗi vòng lặp thứ nhất. Nếu trong vòng lặp thứ 2 thấy có kí tự giống kí tự đang xét ở chuỗi 1 và chưa đánh dấu thì biến flag tăng lên 1. Kết thúc vòng lặp thứ 2 kiểm tra biến flag nếu thấy vẫn bằng 0 chứng tỏ không có kí tự nào ở xâu 2 thỏa mãn và trả về false.



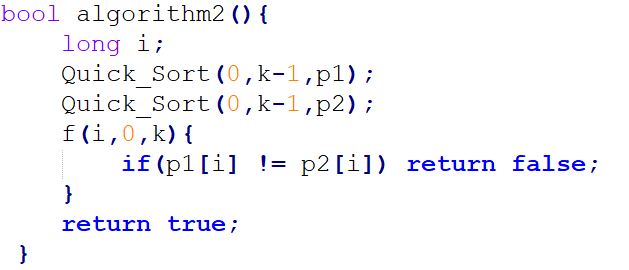
## **SẮP XẾP VÀ SO SÁNH**

1. *Ý tưởng :*

* Sắp xếp hai xâu theo thứ tự tăng dần.
* Duyệt qua các phần tử trong hai xâu nếu tất cả đề trùng nhau thì tức là xâu đảo, ngược lại không là xâu đảo.

1. *Giai pháp :*

* Dùng thuật toán quick sort để sắp xếp.
* Dùng một vòng lặp for để kiểm tra.



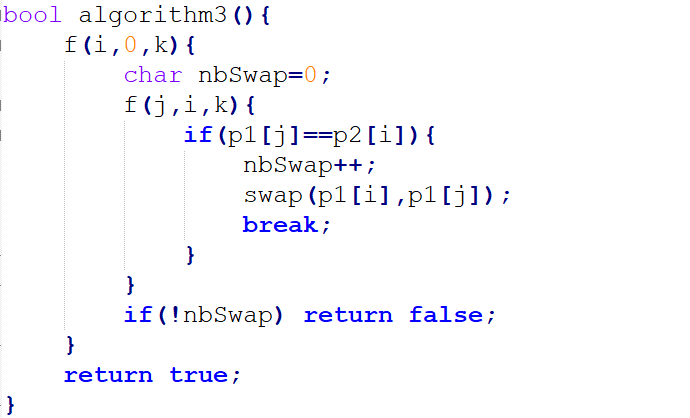
## **VÉT CẠN**

1. *Ý tưởng*

* Đảo các kí tự của xâu 1 cho tới khi nào tìm được xâu 2

1. *Giải pháp*

* Duyệt tuần tự từng phần tử mảng thư 2. Với phần tử thứ i của xâu 2, duyệt xâu thứ nhất từ phần tử i đến cuối cho tới khi nào thấy phần tử thứ k giống với phần tử thứ i của xâu thứ 2. Đổi chỗ phần tử thứ i và phần thứ k trong xâu thứ 1 cho nhau.
* Sử dụng biến nbSwap khởi tạo với giá trị bằng 0 ở đầu vòng lặp thứ nhất. Nếu đổi chỗ thì tăng biến nbSwap lên một đơn vị. Kết thúc mỗi vòng lặp thứ 2 nếu biến nbSwap vẫn không thay đổi giá trị tức hai xâu không phải xâu đảo của nhau.



## **ĐẾM VÀ SO SÁNH**

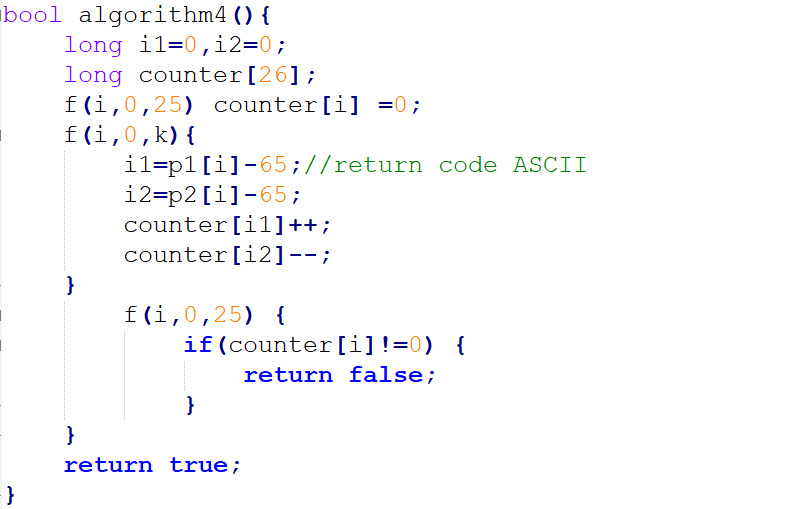
1. *Ý tưởng*

* Đếm số lần xuất hiện của mỗi kí tự trong xâu và lưu vào một mảng counte[].
* So sánh số lần xuất hiện của mỗi kí tự. Nếu giống nhau chứng tỏ hai xâu là xâu đảo của nhau.

1. *Giải pháp*

* Dùng 1 mảng counter gồm 26 phần tử : counter[0] ứng với số kí tự A ,tương tự với counter[1],… đến counter[26] ứng với số kí tự Z
* Duyêt mảng 1 ứng với kí tự nào tăng phần tử counter[] ứng với kí tự đó thêm 1.
* Duyêt mảng 2 ứng với kí tự nào giảm phần tử counter[] ứng với kí tự đó đi 1.

***=>*** 2 xâu đảo nhau khi mọi phần tử trong mang counter =0 (nghĩa là số phần tử giống nhau của 2 mang là bằng nhau).



# **ĐÁNH GIÁ**

## **So sánh**

|  |  |  |  |
| --- | --- | --- | --- |
|  | 100kb | 500kb | 1000kb |
| Checkoff | 16.481 | +00 | +00 |
| Sắp xếp và so sánh | 1.374 | 28.966 | 114.327 |
| Vét cạn | 17.421 | +00 | +00 |
| Đếm và so sánh | 0.001 | 0.005 | 0.008 |

Bảng đánh giá thời gian chạy (s)

* Giai thuật 4 chạy tốn ít thời gian nhất
* Giai thuật 2 tốn thời gian nhất

## **Phân tích**

### **Checkoff**

* Thời gian thực hiên

|  |  |
| --- | --- |
|  | Thời Gian |
| Tồi Nhất | O(n^2) |
| Tốt Nhất | O(n^2) |

* Bộ nhớ

Dùng 2 mảng một chiều để lưu hai xâu kí tự và một mảng đánh dấu mark[].

* Đánh giá: giải thuật checkoff với dữ liệu lớn mất nhiều thời gian chạy và tốn bộ nhớ . Do phải dùng 2 vòng lặp và tốn thêm một bộ nhớ dành cho mảng đánh dấu mark[].

### **Sắp xếp và so sánh**

* Thời gian thực hiện
* Thời gian thực hiện phụ thuộc vào thời gian sắp xếp

|  |  |
| --- | --- |
|  | Thời gian |
| Tồi nhất | O(nlogn) |
| Tốt nhất | O(nlogn) |

* Bộ nhớ
* Chỉ tốn bộ nhớ để lưu trữ 2 xâu kí tự
* Đánh giá :
  + - thời gian thực hiện nhanh hơn và ít tốn bộ nhớ hơn giải thuật checkoff nhưng hạn chế là nếu hai xâu một và hai giống nhau thì vẫn mất thời gian sắp xếp. do đó thời gian khác nhau giữa trường hợp tồi nhất và tốt nhất là không đáng kể.

### **Vét cạn**

* Thời gian thực hiện

|  |  |
| --- | --- |
|  | Thời Gian |
| Tồi nhất | O(n^2) |
| Tốt nhất | O(n) |

* Bộ nhớ
* Chỉ tốn bộ nhớ để lưu trữ 2 xâu kí tự
* Đánh giá: Trong trường hợp tổng quát thì thời gian thực hiện tương tự giải thuật checkoff nhưng không tốn bộ nhớ dành cho mảng đánh dấu mark[] -> Không tốn bộ nhớ.

### **Đếm và so sánh**

* Thời gian thực hiện

|  |  |
| --- | --- |
|  | Thời gian |
| Tồi nhất | O(n) |
| Tốt nhất | O(n) |

* Bộ nhớ
* Tốn bộ nhớ dành để lưu trữ 2 xâu kí tự và mảng counter[].
* Đánh giá: thuật toán đơn giản và cho thời gian thực hiện nhanh -> tối ưu

# **KẾT LUẬN**

* Xét về thời gian :
* GIẢI thuật tối ưu nhất : 4
* Giai thuật kém nhất : 1 và 3
* Giai thuật ổn định : 2
* Xét về bộ nhớ
* Giai thuật 1 tốn bộ nhớ nhiều hơn so với các giải thuật còn lại

=>Nếu phải đưa ra lựa chọn thì nên chọn giải thuật 4 vì giải thuật 4 tốn ít thời gian và bộ nhớ nhất