# Contents

BÁO CÁO THỰC HÀNH TUẦN 04 – TUẦN 14

Bài thực hành số 4

[Contents 1](#_Toc124261819)

[Bài thực hành số 04 – Tuần 14 3](#_Toc124261820)

[**Phần 1: Bài tập thực hành** 3](#_Toc124261821)

[**Bài 4.1:**Đảo ngược một danh sách liên kết đơn. 3](#_Toc124261822)

[Hình 1 - Code bài 4.1 5](#_Toc124261823)

[Hình 2 – Kết quả bài 4.1 5](#_Toc124261824)

[**Bài 4.2.** Một điểm trong không gian 2 chiều được biểu diễn bằng pair. Hãy viết hàm tính diện tích tam giác theo tọa độ 3 đỉnh. 5](#_Toc124261825)

[Hình 3 – Code bài 4.2 7](#_Toc124261826)

[Hình 4 – Kết quả bài 4.2 7](#_Toc124261827)

[**Bài 4.3.** Một vector trong không gian 3 chiều được biểu diễn bằng tuple<double, double, double>. Hãy viết hàm tính tích có hướng của 2 vector. 7](#_Toc124261828)

[Hình 5 – Code bài 4.3 9](#_Toc124261829)

[Hình 6 - Kết quả bài 4.3 9](#_Toc124261830)

[**Bài 4.4** Cho hai std::vector, hãy xóa hết các phần tử chẵn, sắp xếp giảm dần các số trong cả 2 vector và trộn lại thành một vector cũng được sắp xếp giảm dần. 9](#_Toc124261831)

[Hình 7 – Code bài 4.4 11](#_Toc124261832)

[Hình 8 – Kết quả bài 4.4 11](#_Toc124261833)

[**Bài 4.5**. Viết hàm void dfs(vector< list<int> > adj) thực hiện thuật toán DFS không sử dụng đệ quy trên đồ thị biểu diễn bằng danh sách kề. Đồ thị có n đỉnh được đánh số từ 1 đến n. Thuật toán DFS xuất phát từ đỉnh 1. Các đỉnh được thăm theo thứ tự ưu tiên từ trái sang phải trong danh sách kề. Yêu cầu hàm trả ra thứ tự các đỉnh được thăm (những đỉnh không thể thăm từ đỉnh 1 thì không phải in ra). 11](#_Toc124261834)

[Hình 9 – Code bài 4.5 13](#_Toc124261835)

[Hình 10 – Kết quả bài 4.5 13](#_Toc124261836)

[**Bài 4.6.** Viết hàm void bfs(vector< list<int> > adj) thực hiện thuật toán BFS không sử dụng đệ quy trên đồ thị biểu diễn bằng danh sách kề. Đồ thị có n đỉnh được đánh số từ 1 đến n. Thuật toán BFS xuất phát từ đỉnh 1. Các đỉnh được thăm theo thứ tự ưu tiên từ trái sang phải trong danh sách kề. Yêu cầu hàm trả ra thứ tự các đỉnh được thăm (những đỉnh không thể thăm từ đỉnh 1 thì không phải in ra). 13](#_Toc124261837)

[Hình 11 – Code bài 4.6 15](#_Toc124261838)

[Hình 12 – Kết quả bài 4.6 15](#_Toc124261839)

[**Bài 4.7.** Viết các hàm thực hiện các phép giao và hợp của hai tập hợp được biểu diễn bằng set 15](#_Toc124261840)

[template<class T> 15](#_Toc124261841)

[Hình 13 – Code bài 4.7 17](#_Toc124261842)

[Hình 14 – Kết quả bài 4.7 17](#_Toc124261843)

[**Bài 4.8.**Viết các hàm thực hiện các phép giao và hợp của hai tập hợp mờ được biểu diễn bằng map. 17](#_Toc124261844)

[Hình 15 - Code bài 4.8 19](#_Toc124261845)

[Hình 16 - Kết quả bài 4.8 20](#_Toc124261846)

[**Bài 4.9**. Cài đặt thuật toán Dijkstra trên đồ thị vô hướng được biểu diễn bằng danh sách kề sử dụng std::priority\_queue 20](#_Toc124261847)

[Hình 17 – Code bài 4.9 22](#_Toc124261848)

[Hình 18 – Kết quả bài 4.9 23](#_Toc124261849)

[**Phần 2: Bài tập về nhà** 23](#_Toc124261850)

[**Bài 4.10.** Xây dựng một máy tìm kiếm (search engine) đơn giản. 23](#_Toc124261851)

[Hình 19 – Code bài 4.10 28](#_Toc124261852)

[Hình 20 – Kết quả bài 4.10 28](#_Toc124261853)

[**Bài 4.11.** Bức tường bao quanh một lâu đài nọ được cấu thành từ n đoạn tường được đánh số từ 1 đến n. Quân giặc lên kế hoạch tấn công lâu đài bằng cách gửi ai tên giặc đánh vào đoạn tường thứ i. Để bảo vệ lâu đài có tất cả s lính. 28](#_Toc124261854)

[Hình 21 – Code bài 4.11 30](#_Toc124261855)

[Hình 22 – Kết quả bài 4.11 31](#_Toc124261856)

[**Bài 4.12.** Cho một lược đồ gồm n cột chữ nhật liên tiếp nhau có chiều rộng bằng 1 và chiều cao lần lượt là các số nguyên không âm h1,h2,…,hn . Hãy xác định hình chữ nhật có diện tích lớn nhất có thể tạo thành từ các cột liên tiếp. 31](#_Toc124261857)

[Hình 23 - Code bài 4.12 33](#_Toc124261858)

[Hình 24 - Kết quả bài 4.12 34](#_Toc124261859)

[**Bài 4.13**: Cho một xâu nhị phân độ dài n. Hãy viết chương trình đếm số lượng xâu con chứa số ký tự 0 và số ký tự 1 bằng nhau. 34](#_Toc124261860)

[Hình 25 - Code bài 4.13 35](#_Toc124261861)

[Hình 26 - Kết quả bài 4.13 36](#_Toc124261862)

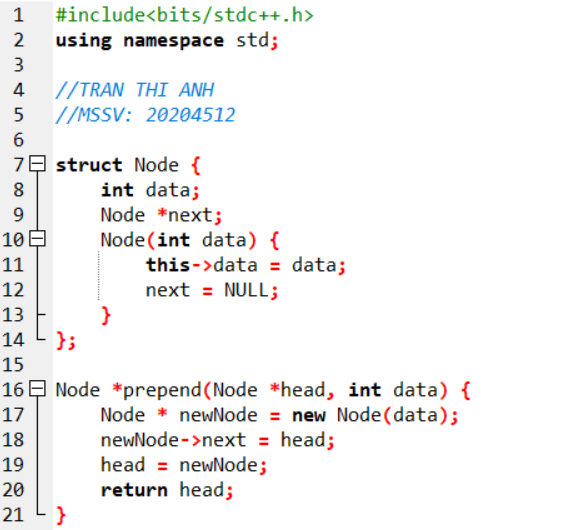
# Bài thực hành số 04 – Tuần 14

# **Phần 1: Bài tập thực hành**

### **Bài 4.1:**Đảo ngược một danh sách liên kết đơn.

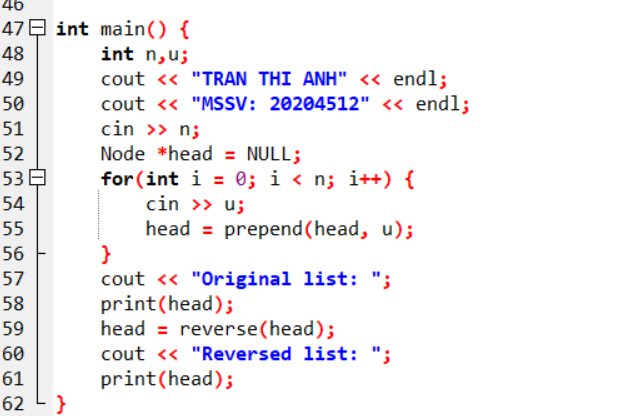
Hãy hoàn thiện các hàm thao tác trên một danh sách liên kết:

* Thêm một phần tử vào đầu danh sách liên kết
* In danh sách
* Đảo ngược danh sách liên kết (yêu cầu độ phức tạp thời gian O(N) và chi phí bộ nhớ dùng thêm O(1))



Text

Description automatically generated



#### Hình 1 - Code bài 4.1

Text

Description automatically generated

#### Hình 2 – Kết quả bài 4.1

### **Bài 4.2.** Một điểm trong không gian 2 chiều được biểu diễn bằng pair. Hãy viết hàm tính diện tích tam giác theo tọa độ 3 đỉnh.

double area(Point a, Point b, Point c) {

    /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    # YOUR CODE HERE #

    \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

trong đó, Point là kiểu được định nghĩa trước trong trình chấm như sau:

using Point = pair<double, double>;

Graphical user interface, text

Description automatically generated

#### Hình 3 – Code bài 4.2

Text

Description automatically generated

#### Hình 4 – Kết quả bài 4.2

### **Bài 4.3.** Một vector trong không gian 3 chiều được biểu diễn bằng tuple<double, double, double>. Hãy viết hàm tính tích có hướng của 2 vector.

Vector cross\_product(Vector a, Vector b) {

    /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    # YOUR CODE HERE #

    \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

trong đó Vector là kiểu được định nghĩa sẵn trong trình chấm như sau:

using Vector = tuple<double, double, double>;

Text

Description automatically generated

#### Hình 5 – Code bài 4.3

Graphical user interface, text, email

Description automatically generated

#### Hình 6 - Kết quả bài 4.3

### **Bài 4.4** Cho hai std::vector, hãy xóa hết các phần tử chẵn, sắp xếp giảm dần các số trong cả 2 vector và trộn lại thành một vector cũng được sắp xếp giảm dần.

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

#### Hình 7 – Code bài 4.4

Text

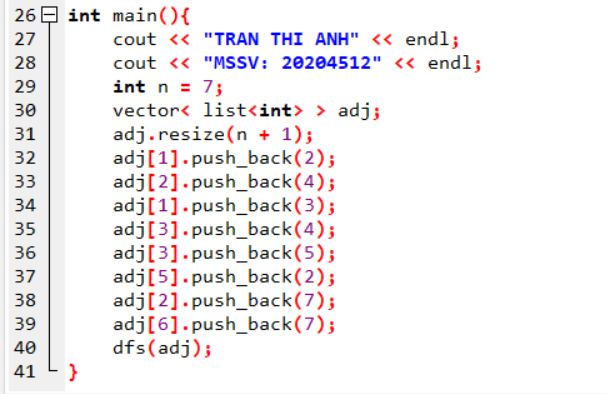
Description automatically generated

#### Hình 8 – Kết quả bài 4.4

### **Bài 4.5**. Viết hàm void dfs(vector< list<int> > adj) thực hiện thuật toán DFS không sử dụng đệ quy trên đồ thị biểu diễn bằng danh sách kề. Đồ thị có n đỉnh được đánh số từ 1 đến n. Thuật toán DFS xuất phát từ đỉnh 1. Các đỉnh được thăm theo thứ tự ưu tiên từ trái sang phải trong danh sách kề. Yêu cầu hàm trả ra thứ tự các đỉnh được thăm (những đỉnh không thể thăm từ đỉnh 1 thì không phải in ra).

Text

Description automatically generated



#### Hình 9 – Code bài 4.5

Text

Description automatically generated

#### Hình 10 – Kết quả bài 4.5

### **Bài 4.6.** Viết hàm void bfs(vector< list<int> > adj) thực hiện thuật toán BFS không sử dụng đệ quy trên đồ thị biểu diễn bằng danh sách kề. Đồ thị có n đỉnh được đánh số từ 1 đến n. Thuật toán BFS xuất phát từ đỉnh 1. Các đỉnh được thăm theo thứ tự ưu tiên từ trái sang phải trong danh sách kề. Yêu cầu hàm trả ra thứ tự các đỉnh được thăm (những đỉnh không thể thăm từ đỉnh 1 thì không phải in ra).

Text

Description automatically generated

Text, letter

Description automatically generated

#### Hình 11 – Code bài 4.6

Text

Description automatically generated

#### Hình 12 – Kết quả bài 4.6

### **Bài 4.7.** Viết các hàm thực hiện các phép giao và hợp của hai tập hợp được biểu diễn bằng set

### template<class T>

set<T> set\_union(const set<T> &a, const set<T> &b) {

    /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    # YOUR CODE HERE #

    \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

template<class T>

set<T> set\_intersection(const set<T> &a, const set<T> &b) {

    /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    # YOUR CODE HERE #

    \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

Text

Description automatically generated

Text

Description automatically generated with medium confidence

#### Hình 13 – Code bài 4.7

Text

Description automatically generated

#### Hình 14 – Kết quả bài 4.7

### **Bài 4.8.**Viết các hàm thực hiện các phép giao và hợp của hai tập hợp mờ được biểu diễn bằng map.

Trong đó mỗi phần tử được gán cho một số thực trong đoạn [0..1] biểu thị độ thuộc của phần tử trong tập hợp, với độ thuộc bằng 1 nghĩa là phần tử chắc chắn thuộc vào tập hợp và ngược lại độ thuộc bằng 0 nghĩa là phần tử chắc chắn không thuộc trong tập hợp.

Phép giao và hợp của 2 tập hợp được thực hiện trên các cặp phần tử bằng nhau của 2 tập hợp, với độ thuộc mới được tính bằng phép toán min và max của hai độ thuộc.

template<class T>

map<T, double> fuzzy\_set\_union(const map<T, double> &a, const map<T, double> &b) {

    /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    # YOUR CODE HERE #

    \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

template<class T>

map<T, double> fuzzy\_set\_intersection(const map<T, double> &a, const map<T, double> &b) {

    /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    # YOUR CODE HERE #

    \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}

Text

Description automatically generated with low confidence

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

#### Hình 15 - Code bài 4.8

A picture containing text

Description automatically generated

#### Hình 16 - Kết quả bài 4.8

### **Bài 4.9**. Cài đặt thuật toán Dijkstra trên đồ thị vô hướng được biểu diễn bằng danh sách kề sử dụng std::priority\_queue

Cụ thể, bạn cần cài đặt hàm *vector<int> dijkstra(const vector< vector< pair<int, int> > >&adj)*nhận đầu vào là danh sách kề chứa các cặp *pair<int, int>* biểu diễn đỉnh kề và trọng số tương ứng của cạnh. Đồ thị gồm n đỉnh được đánh số từ 0 tới n-1. Hàm cần trả `vector<int>` chứa n phần tử lần lượt là khoảng cách đường đi ngắn nhất từ đỉnh 0 tới các đỉnh 0, 1, 2, ..., n-1.

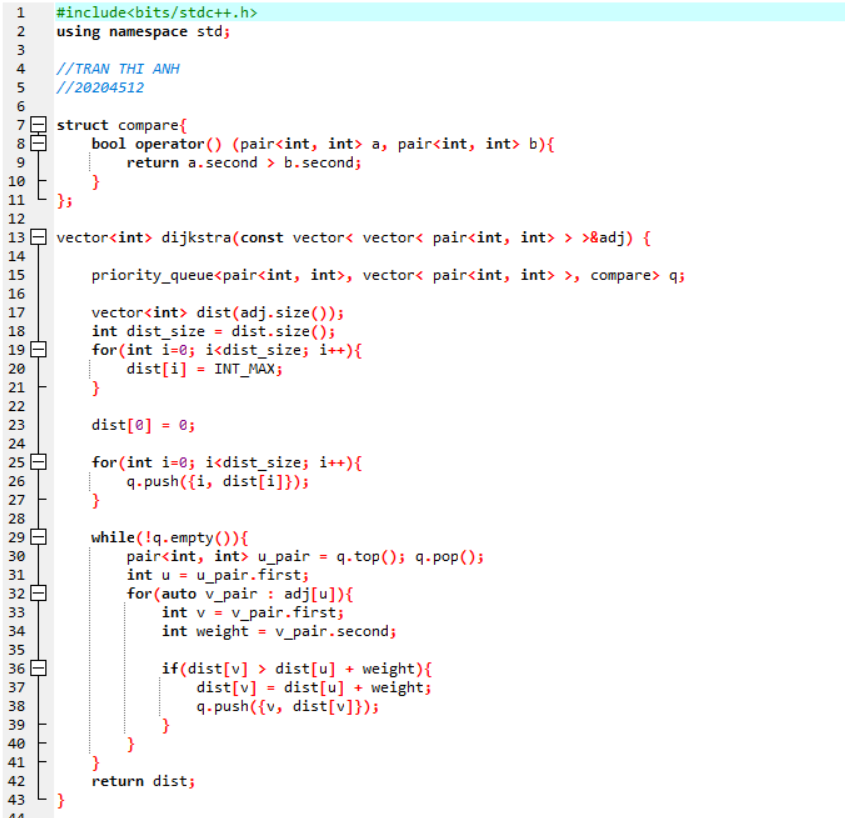
vector<int> dijkstra(const vector< vector< pair<int, int> > >&adj) {

    /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    # YOUR CODE HERE #

    \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

}



Text

Description automatically generated

#### Hình 17 – Code bài 4.9

Text

Description automatically generated

#### Hình 18 – Kết quả bài 4.9

# **Phần 2: Bài tập về nhà**

### **Bài 4.10.** Xây dựng một máy tìm kiếm (search engine) đơn giản.

Cho N văn bản và Q truy vấn. Với mỗi truy vấn, cần trả về văn bản khớp với truy vấn đó nhất.

**Input:**

* Dòng đầu tiên chứa số N
* Dòng thứ i trong N dòng tiếp theo thể hiện văn bản i, mỗi dòng là một dãy các từ ngăn cách nhau bởi dấu phẩy
* Dòng tiếp theo chứa số Q
* Dòng thứ i trong Q dòng tiếp theo thể hiện truy vấn thứ i, mỗi dòng là một dãy các từ ngăn cách nhau bởi dấu phẩy

**Output:** Gồm Q dòng, dòng thứ i là chỉ số của văn bản khớp với truy vấn thứ i nhất. Nếu có nhiều văn bản có điểm số bằng nhau, in ra văn bản có chỉ số nhỏ nhất.

Text

Description automatically generated with medium confidence

Text

Description automatically generated

A picture containing text

Description automatically generated

Text

Description automatically generated

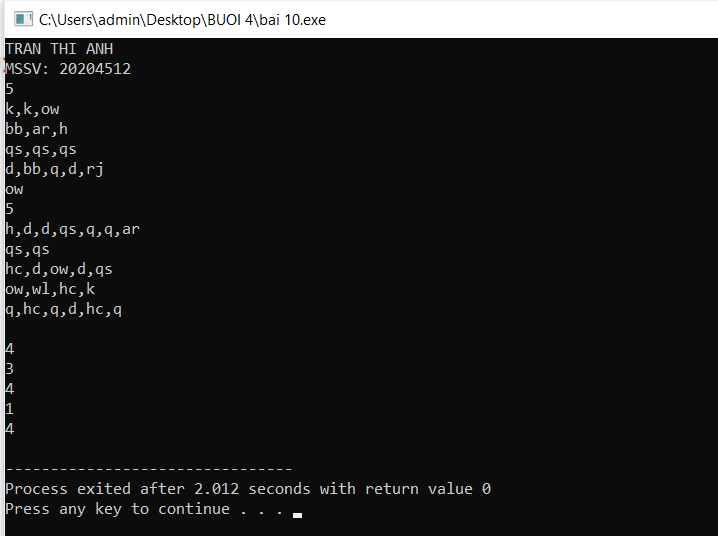
Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

#### Hình 19 – Code bài 4.10



#### Hình 20 – Kết quả bài 4.10

### **Bài 4.11.** Bức tường bao quanh một lâu đài nọ được cấu thành từ n đoạn tường được đánh số từ 1 đến n. Quân giặc lên kế hoạch tấn công lâu đài bằng cách gửi ai tên giặc đánh vào đoạn tường thứ i. Để bảo vệ lâu đài có tất cả s lính.

Do các đoạn tường có chất lượng khác nhau nên khả năng bảo vệ tại các đoạn tường cũng khác nhau. Cụ thể tại đoạn tường thứ i, mỗi lính có thể đẩy lùi tấn công của ki tên giặc. Giả sử đoạn tường thứ i có xi lính. Khi đó nếu số tên giặc không vượt quá xi thì không có tên giặc nào lọt vào được qua đoạn tường này. Ngược lại sẽ có ai - xi \* ki tên giặc lọt vào lâu đài qua đoạn tường này.

Yêu cầu hãy viết chương trình phân bố lính đứng ở các đoạn tường sao cho tổng số lính là s và tổng số lượng tên giặc lọt vào lâu đài là nhỏ nhất.

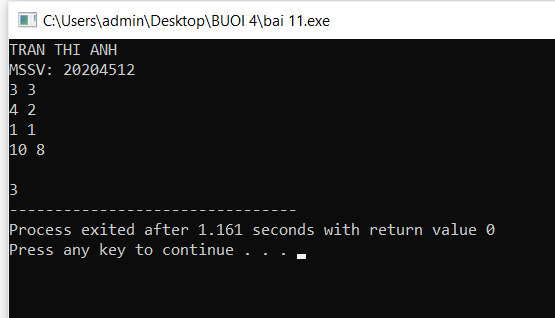
Text

Description automatically generated

Text

Description automatically generated

#### Hình 21 – Code bài 4.11

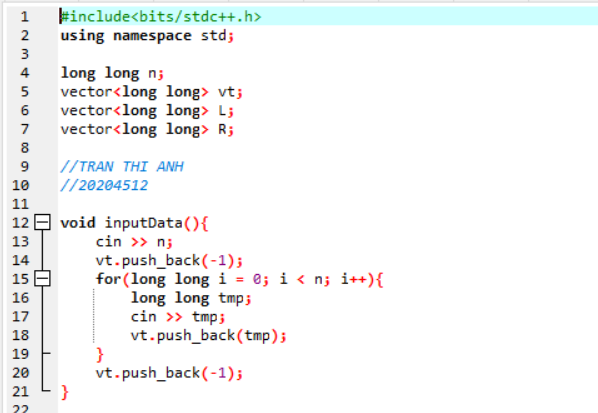


#### Hình 22 – Kết quả bài 4.11

### **Bài 4.12.** Cho một lược đồ gồm n cột chữ nhật liên tiếp nhau có chiều rộng bằng 1 và chiều cao lần lượt là các số nguyên không âm h1,h2,…,hn . Hãy xác định hình chữ nhật có diện tích lớn nhất có thể tạo thành từ các cột liên tiếp.

**Dữ liệu vào:** Dòng thứ nhất chứa số nguyên dương n (1 ≤ n ≤ 106). Dòng thứ hai chứa n số nguyên không âm h1,h2,…,hn cách nhau bởi dấu cách (0 ≤ hi ≤ 109).

**Kết quả:** In ra số nguyên duy nhất là diện tích hình chữ nhật lớn nhất có thể tạo thành từ các cột liên tiếp của lược đồ.



A picture containing text

Description automatically generated

#### Hình 23 - Code bài 4.12

Text

Description automatically generated

#### Hình 24 - Kết quả bài 4.12

### **Bài 4.13**: Cho một xâu nhị phân độ dài n. Hãy viết chương trình đếm số lượng xâu con chứa số ký tự 0 và số ký tự 1 bằng nhau.

**Dữ liệu vào:** Một dòng duy nhất chứa một xâu nhị phân độ dài n ( 1 ≤ n ≤ 106 ).

**Kết quả:** Ghi ra một số nguyên duy nhất là số lượng xâu con có số ký tự 0 và số ký tự 1 bằng nhau.

Text

Description automatically generated with medium confidence

Text

Description automatically generated

#### Hình 25 - Code bài 4.13

Text

Description automatically generated

#### Hình 26 - Kết quả bài 4.13