Học Viện Kỹ Thuật Quân Sự

**BÁO CÁO ĐỀ TÀI MÔN HỌC**



**ĐỀ TÀI SỐ 22:**

**Xây dựng chương trình soạn thảo, hoặc dạng add-in**

**vào trong chương trình soạn thảo, cho phép tìm kiếm một số thông tin dạng đặc biệt (ví dụ tên tuổi, địa danh, số đt, vvv…)**

**Giảng Viên:**

Hà Chí Trung

**Người làm:**

SV. Ngô Đình Phúc

Lớp: KTPM-k14

MSV: 15150178

NS: 18/08/1997

**Mục Lục**

[1 Đặt vấn đề. 2](#_Toc515026036)

[2 Kiến thức liên quan 3](#_Toc515026037)

[2.1 Automat HỮU HẠN (FA : Finite Automata) 3](#_Toc515026038)

[2.2 Ôtômát hữu hạn đơn định - DFA (Deterministic Finite Automata) 4](#_Toc515026039)

[2.3 Ôtômát hữu hạn không đơn định - NFA (Nondeterministic Finite Automata) 4](#_Toc515026040)

[2.4 Regular Expression là gì? 4](#_Toc515026041)

[2.5 Ví dụ các biểu thức chính qui sử dụng trong chương trình: 4](#_Toc515026042)

[3 Cấu trúc chương trình. 5](#_Toc515026043)

[4 Sơ đồ thực hiện tìm kiếm 6](#_Toc515026044)

[5 Hướng dẫn và kiểm nghiệm. 8](#_Toc515026045)

[5.1 Chọn dữ liệu input 8](#_Toc515026046)

[5.2 Xử lý 8](#_Toc515026047)

[6 Kết luận 9](#_Toc515026048)

[7 Tài liệu tham khảo 9](#_Toc515026049)

# Đặt vấn đề.

Trong văn bản với khối lượng lớn dữ liệu, ta cần bóc tách hoặc tìm kiếm 1 đối tượng cụ thể.

Việc tìm kiếm thông thường sẽ mất rất nhiều thời gian.

Những đối tượng cần tìm kiếm có giá trị không giống nhau, ví dụ tìm kiếm đối tượng là email có rất nhiều giá trị khác nhau như: [abc@gmail.com](mailto:abc@gmail.com), [a\_B\_c@yahoo.com](mailto:a_B_c@yahoo.com)

Bằng cách sử dụng biểu thức chính quy (Regular Expression – RE) ta có thể tìm kiếm tất cả các giá trị dưới dạng cho sẵn bởi RE.

Mục đích của bài tập là nhận dạng được 9 thực thể trong văn bản cho trước: Email, số điện thoại, tên riêng, địa chỉ URL, ngày tháng,tiền tệ,IP, file bằng nhiều loại filter khác nhau

Ngoài ra chương trình có thể highlight từ khóa trong văn bản và đưa ra thống kê.

Nhằm mục đích dễ dàng sử dụng và phát triển, chương trình được xây dựng bằng ngôn ngữ C#.Net trên winsform. Dữ liệu đầu vào là file văn bản nhiều định dạng .rtf, txt, pdf, json, html...

# Kiến thức liên quan

## Automat HỮU HẠN (FA : Finite Automata)

Automat hữu hạn FA là một mô hình tính toán của hệ thống với sự mô tả bởi các input và output. Tại mỗi thời điểm, hệ thống có thể được xác định ở một trong số hữu hạn các cấu hình nội bộ gọi là các trạng thái (states). Mỗi trạng thái của hệ thống thể hiện sự tóm tắt các thông tin liên quan đến những input đã chuyển qua và xác định các phép chuyển kế tiếp trên dãy input tiếp theo.

Trong khoa học máy tính, ta có thể tìm thấy nhiều ví dụ về hệ thống trạng thái hữu hạn, và lý thuyết về ôtômát hữu hạn là một công cụ thiết kế hữu ích cho các hệ thống này. Chẳng hạn, một hệ chuyển mạch như bộ điều khiển (Control Unit) trong máy tính. Một chuyển mạch thì bao gồm một số hữu hạn các cổng (gate) input, mỗi cổng có 2 giá trị 0 hoặc 1. Các giá trị đầu vào này sẽ xác định 2 mức điện thế khác nhau ở cổng output. Mỗi trạng thái của một mạng chuyển mạch với n cổng bất kỳ sẽ là một trường hợp trong 2n phép gán của 0 và 1 đối với các cổng khác nhau. Các chuyển mạch thì được thiết kế theo cách này, vì thế chúng có thể được xem như hệ thống trạng thái hữu hạn. Các chương trình sử dụng thông thường, chẳng hạn trình sọan thảo văn bản hay bộ phân tích từ vựng trong trình biên dịch máy tính cũng được thiết kế như các hệ thống trạng thái hữu hạn. Ví dụ bộ phân tích từ vựng sẽ quét qua tất cả các dòng ký tự của chương trình máy tính để tìm nhóm các chuỗi ký tự tương ứng với một tên biến, hằng số, từ khóa, …Trong quá trình xử lý này, bộ phân tích từ vựng cần phải nhớ một số hữu hạn thông tin như các ký tự bắt đầu hình thành những chuỗi từ khóa. Lý thuyết về Automat hữu hạn thường được dùng đến nhiều cho việc thiết kế các công cụ xử lý chuỗi hiệu quả.

## Ôtômát hữu hạn đơn định - DFA (Deterministic Finite Automata)

Một ôtômát hữu hạn đơn định (DFA) - gọi tắt là FA -gồm một tập hữu hạn cáctrạng thái và một tập các phép chuyển từ trạng thái này tới trạng thái khác trên các ký hiệu nhập (input symbols) được chọn từ một bộ chữ cái Σ nào đó. Mỗi ký hiệu nhập có đúng một phép chuyển khỏi mỗi trạng thái (có thể chuyển trở về chính nó). Một trạng thái, thường ký hiệu là q0, gọi là trạng thái bắt đầu (trạng thái ôtômát bắt đầu). Một số trạng thái được thiết kế như là các trạng thái kết thúc hay trạng thái chấp nhận.

## Ôtômát hữu hạn không đơn định - NFA (Nondeterministic Finite Automata)

Xét một dạng sửa đổi mô hình DFA để chấp nhận không, một hoặc nhiều hơn một phép chuyển từ một trạng thái trên cùng một ký hiệu nhập. Mô hình mới này gọi là ôtômát hữu hạn không đơn định (NFA).

Một chuỗi ký hiệu nhập a1 a2 ... an được chấp nhận bởi một NFA nếu có tồn tại một chuỗi các phép chuyển, tương ứng với chuỗi nhập, từ trạng thái bắt đầu đến trạng thái kết thúc. Chẳng hạn, chuỗi 01001 được chấp nhận bởi ôtômát trong hình dưới đây vì có chuỗi phép chuyển qua các trạng thái q0, q0, q0, q3, q4, q4 có nhãn tương ứng là 0, 1, 0, 0, 1. NFA này chấp nhận tất cả các chuỗi có hai số 0 liên tiếp hoặc hai số 1 liên tiếp.

## Regular Expression là gì?

Regular Expression hay còn gọi là biểu thức chính quy được dùng để xử lý chuỗi nâng cao thông qua biểu thức riêng của nó, những biểu thức này sẽ có những nguyên tắc riêng và bạn phải tuân theo nguyên tắc đó thì biểu thức của bạn mới hoạt động được. Ngoài cái tên gọi Regular Expression ra thì nó còn có thể viết tắt thành RegEx.

Nguyên tắc hoạt động của biểu thức RegEx là so khớp dựa vào khuôn mẫu, khuôn mẫu được xây dựng từ các quy tắc căn bản của biểu thức RegEx

## Ví dụ các biểu thức chính qui sử dụng trong chương trình:

* Tìm email:

\b[A-Z0-9.\_+-][+@[A-Z0-9.-]+\.[A-Z]{2,4}\b](mailto:+@[A-Z0-9.-]+\.%5bA-Z%5d%7b2,4%7d\b)

Khớp với: [abc@abc.com](mailto:abc@abc.com), [abc-abc@abc.com](mailto:abc-abc@abc.com)

Không khớp: [abc^abc@abc.com](mailto:abc%5eabc@abc.com) …

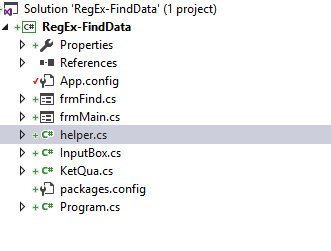
* Tìm số điện thoại:

(\(\+84\)\s\d{9,10})|((09)[0-9]{8})|((0)[0-9]{1}[ .][0-9]{3}[ .][0-9]{3}[ .][0-9]{2})|((09)[0-9]{2}[ .][0-9]{3}[ .][0-9]{3})|((09)[0-9]{1}[ .][0-9]{3}[ .][0-9]{4})|((012)[0-9]{8})|((012)[0-9]{1}[ .][0-9]{3}[ .][0-9]{4})|((016)[2-9]{1}[0-9]{7})|((016)[2-9]{1}[ .][0-9]{3}[ .][0-9]{4})

Khớp với: (+84) 973642632 01667234302 0168.893.4270

Không khớp: 016672343025 (+84) 97364263266

# Cấu trúc chương trình.



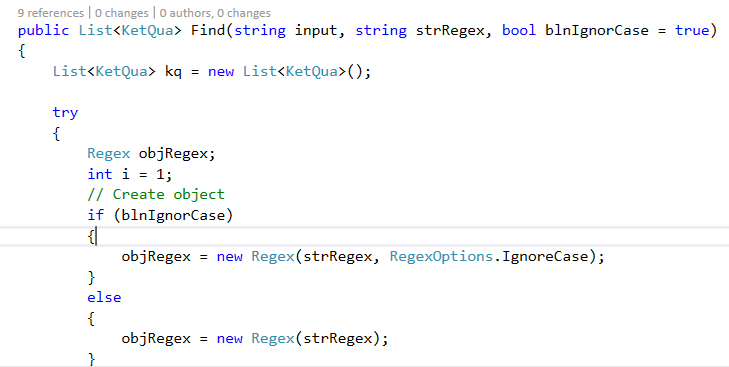
+2 class định nghĩa ra đối tượng trả về

+ class helper.cs hỗ trợ các phương thức mở file

Hiển thị, xuất file excel

+ frmMan Chứa rich textBox giúp thao tác chỉnh sửa text trên đó

+ frmFind Cửa sổ khi ấn vào tìm kiếm hoặc ấn vào Ctrl+F giúp thực hiện thao tác chính của chương trình là tìm kiếm chuỗi khớp với điều kiện được chọn



Hàm Find:

Tham số đưa vào gồm

+ string input Là chuỗi dữ liệu ban đầu

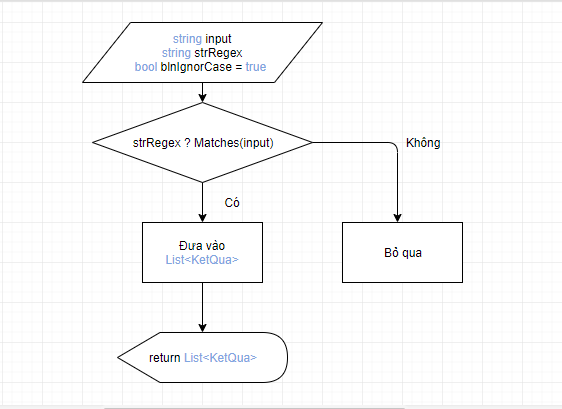
+ string strRegex là pattern regex

+ bool blnIgnorCase = true xuất hiện có hoặc không, nếu trả về true   
là không phân biệt kiểu của 2 tham số,

Out Put đưa ra là :

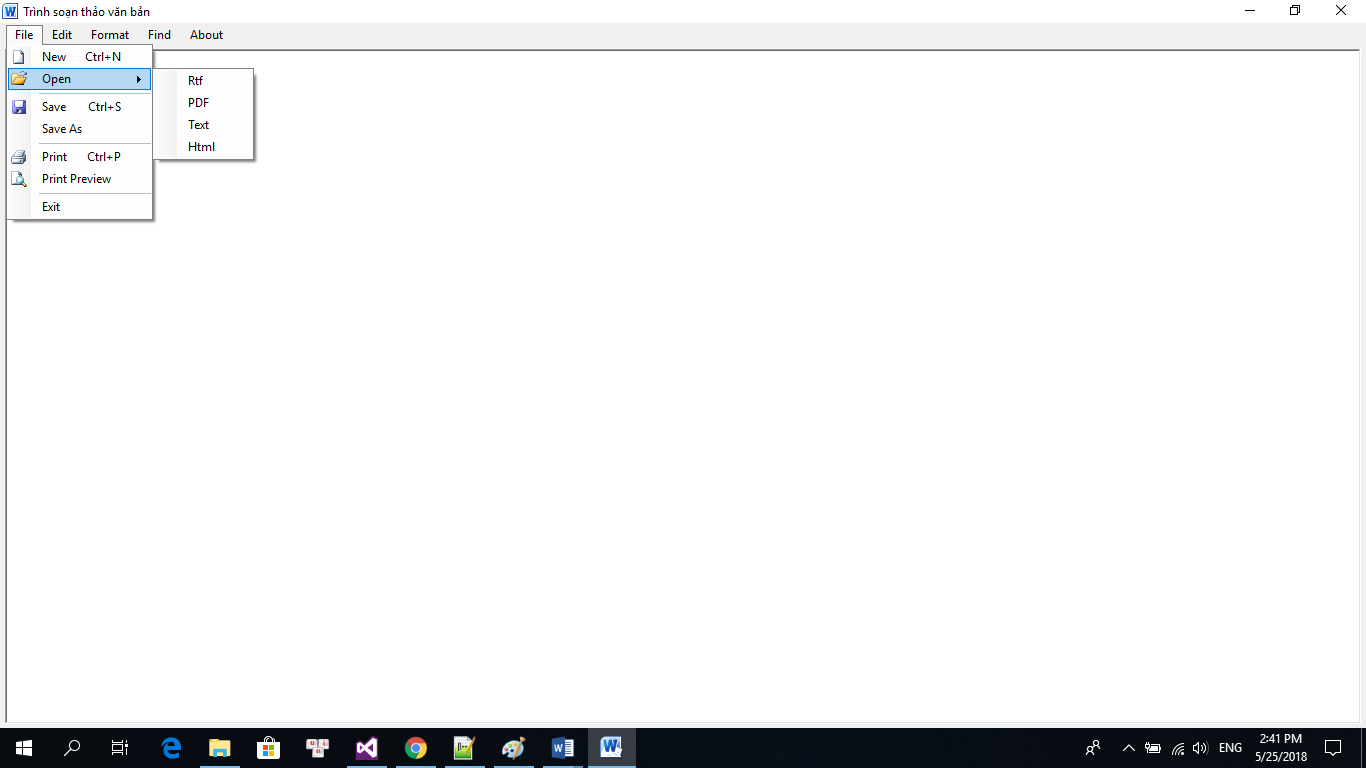
+ List<KetQua>

# Sơ đồ thực hiện tìm kiếm

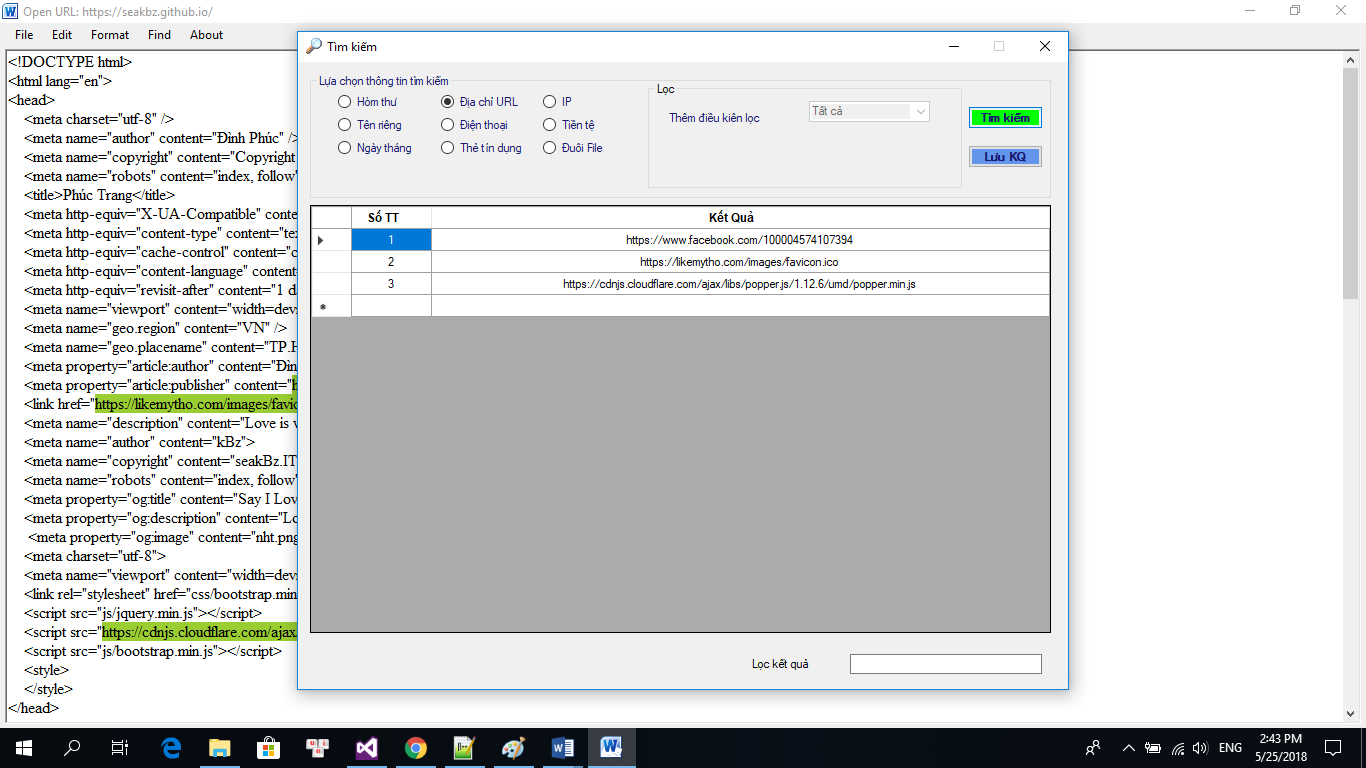


# Hướng dẫn và kiểm nghiệm.

## Chọn dữ liệu input



## Xử lý



Chọn radio button và nếu có điều kiện thì select ở combobox-> ấn nút tìm kiếm

+ Xuất file: excel

+Lọc danh sách kết quả

# Kết luận

Trong quá trình làm em đã tìm hiểu và học thêm được rất nhiều kiến thức Em cảm ơn thầy với đề tài thực tế và có tính ứng dụng cao

Nhược điểm: vì thời gian hơi hạn hẹp nên em lấy giao điện bài mẫu thầy cho và kế thừa lại những cái tốt của chương trình khắc phục và phát truyển thành kiến thức cũng như bài tập và sự tìm tòi học hỏi của bản thân

# Tài liệu tham khảo

<http://regex101.com>

<http://regexlib.com>

<http://stackoverflow.com>

http://google.com/

THE END.