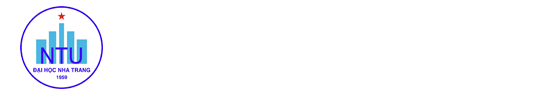
**BỘ GIÁO DỤC VÀ ĐÀO TẠO**



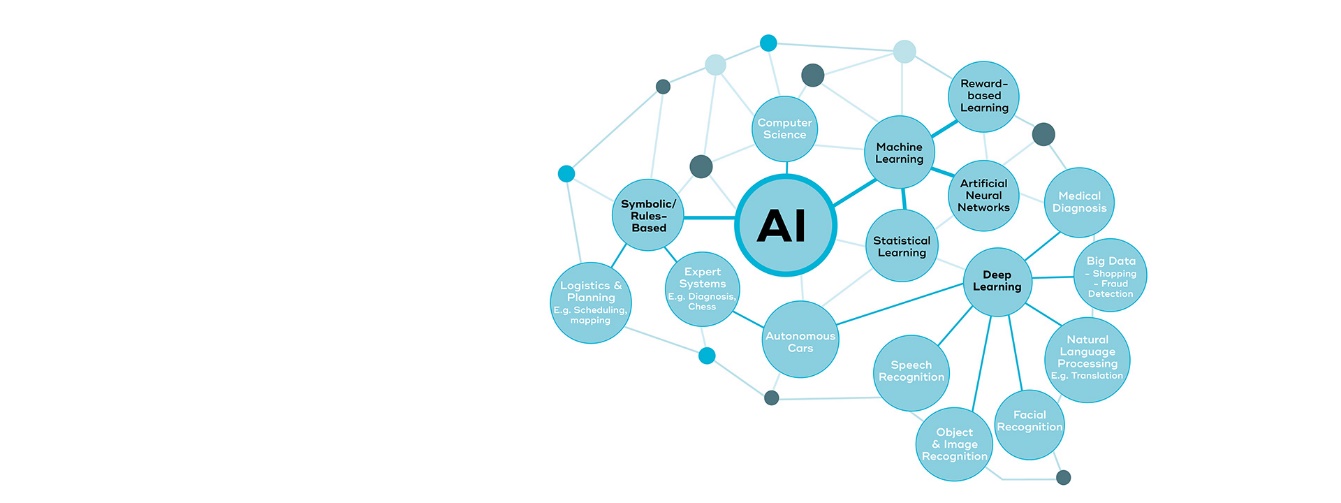
**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO BÀI TẬP HÈ**

**MÔN HỌC: TRÍ TUỆ NHÂN TẠO**



**Giảng viên bộ môn : Nguyễn Đình Cường**

**Sinh viên thực hiện : Nguyễn Văn Hải Long**

**Lớp : 60.CNTT-1**

**MSSV : 60136035**

**Tháng 8 năm 2021**

**PHỤ LỤC**

[**1. Programming Heuristic 3**](#_Toc81259536)

[**1.1. Thuật toán tìm kiếm A\*(A star) 3**](#_Toc81259537)

[**1.2. Chương trình demo: Puzzle 8 sử dụng thuật toán A\* 3**](#_Toc81259538)

[**2. Theory of game (Lý thuyết về trò chơi) 6**](#_Toc81259539)

[**2.1. Thuật toán Minimax 6**](#_Toc81259540)

[**2.2. Tinh chỉnh Alpha-beta (cắt tỉa Alpha- beta) 7**](#_Toc81259541)

[**2.3. Chương trình demo: Cờ vua dùng thuật toán Minimax 7**](#_Toc81259542)

[**3. Logic and Semantic (Logic và ngữ nghĩa) 9**](#_Toc81259543)

[**3.1. Giải thuật Vương Hạo 9**](#_Toc81259544)

[**3.2. Chương trình demo: Mô phỏng thuật toán Vương Hạo trong Prolog 10**](#_Toc81259545)

[**4. Simulation (Mô phỏng) 12**](#_Toc81259546)

[**4.1. Genetic Algorithm applied to optimization ( Thuật toán di truyền áp dụng để tối ưu hoá) 12**](#_Toc81259547)

[**4.2. Chương trình demo: Tìm đường đi sử dụng thuật toán di truyền 12**](#_Toc81259548)

[**5. IoT and Robotics applied in with AI (IoT và Robotics được ứng dụng với AI) 14**](#_Toc81259549)

[**5.1. Sử dụng Arduino để chế tạo cánh tay robot 14**](#_Toc81259550)

[**5.2. Chương trình demo: Cánh tay robot 15**](#_Toc81259551)

**BÁO CÁO BÀI TẬP**

# 1. Programming Heuristic

## 1.1. Thuật toán tìm kiếm A\*(A star)

A\* là một thuật toán tìm kiếm trong đồ thị. Thuật toán sẽ tìm đường từ 1 nút ban đầu đến 1 nút đích cho trước sao cho chi phí là tốt nhất (thấp nhất) và số bước duyệt là ít nhất (mang lại sự tối ưu cho bài toán).

A\* là thuật toán cải thiện hiệu năng từ thuật toán Greedy Best-First Search. Khi Greedy Best-First Search chỉ tìm đường dựa trên hàm ước lượng từ điểm kế tiếp đến đích mà không tính độ dài quãng đường mà nó đã đi vì có thể ước lượng là nhỏ nhưng đường đã đi lạimang giá trị quá lớn.

Ý tưởng : Tránh việc xét các nhánh tìm kiếm đã xác định cho đến thời điểm hiện tại là có chi phí cao.

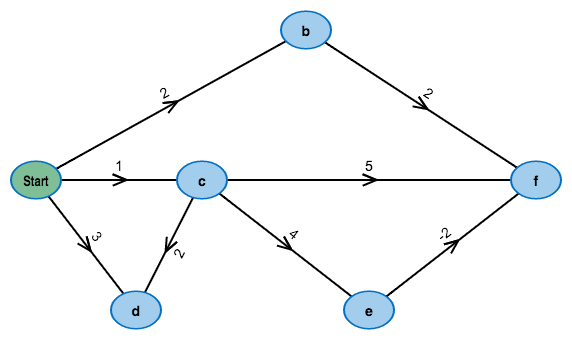
Sử dụng hàm đánh giá f(x) = h(x) + g(x).

h(x) : là ước lượng từ nút hiện tại tới nút đích

g(x) : chi phí từ nút gốc tới nút hiện tại n

f(x) : là chi phí tổng thể ước lượng của đường đi qua nút hiện tại n đến đích.

Thuật toán A\* sử dụng f(x) làm tiêu chí chọn đỉnh tiếp theo trong đồ thị.



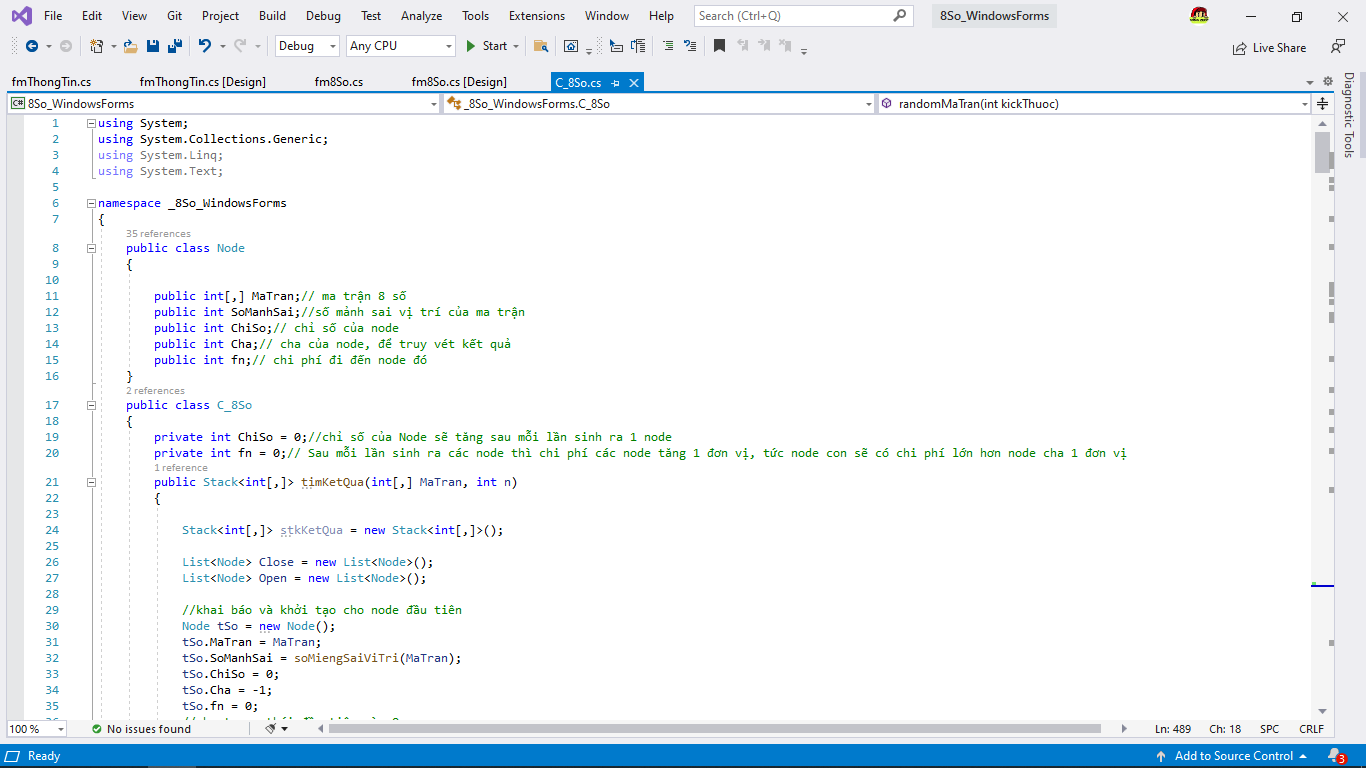
## 1.2. Chương trình demo: Puzzle 8 sử dụng thuật toán A\*

Ngôn ngữ sử dụng: C#

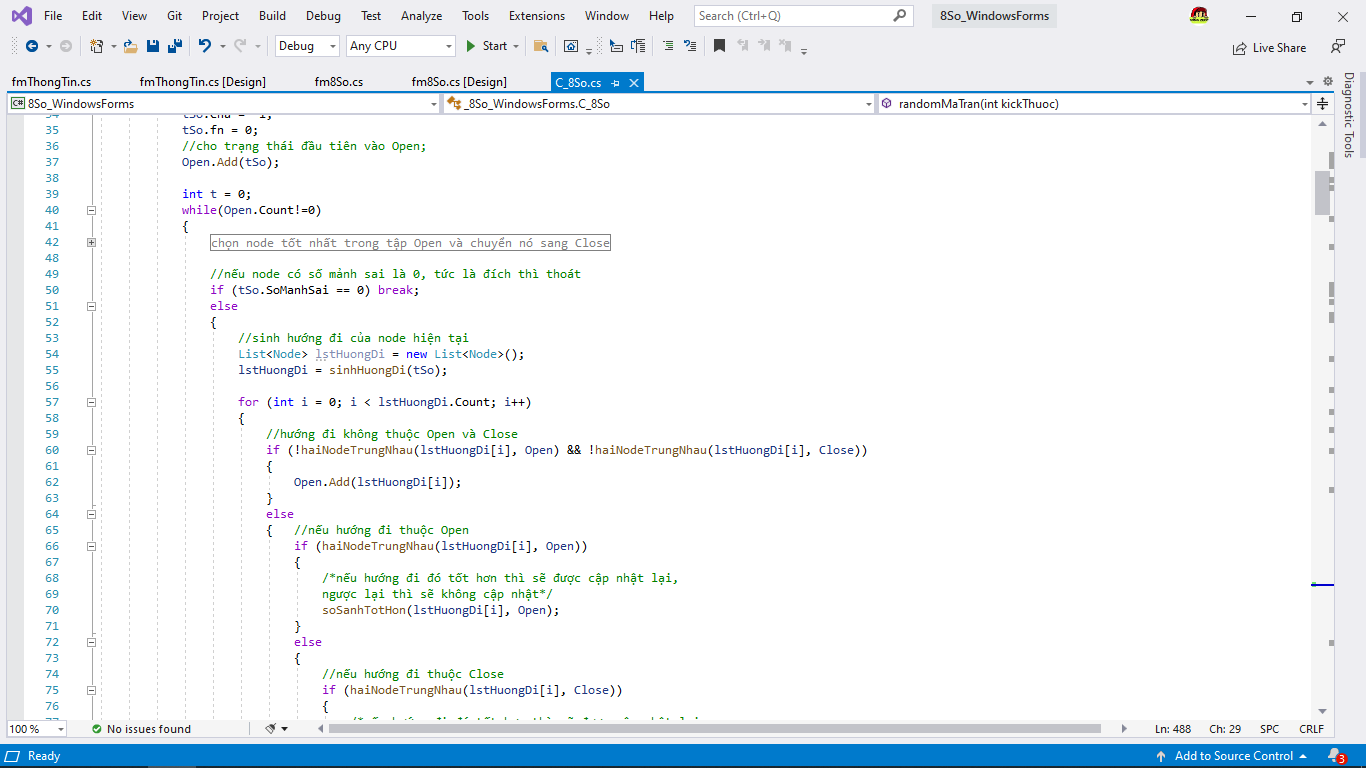
Giao diện: Winform

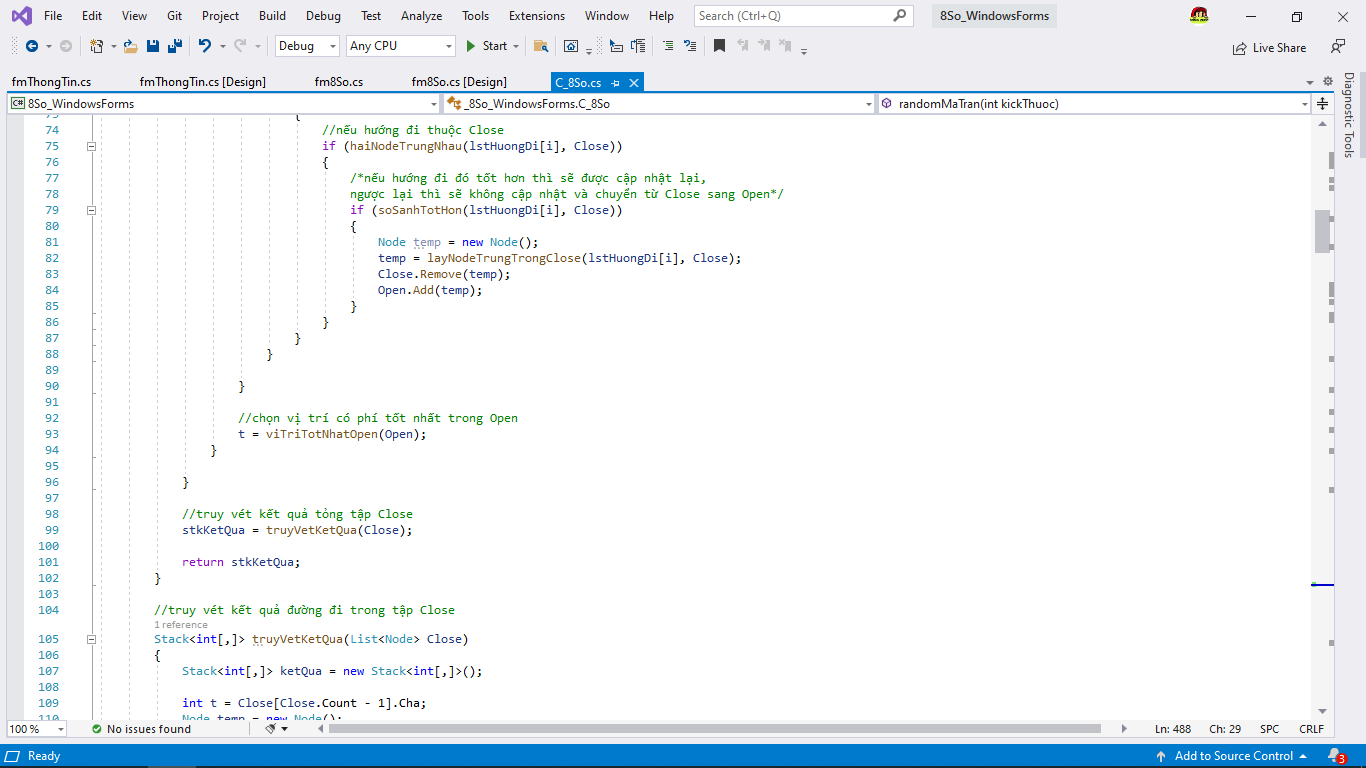
Một số hình ảnh code:

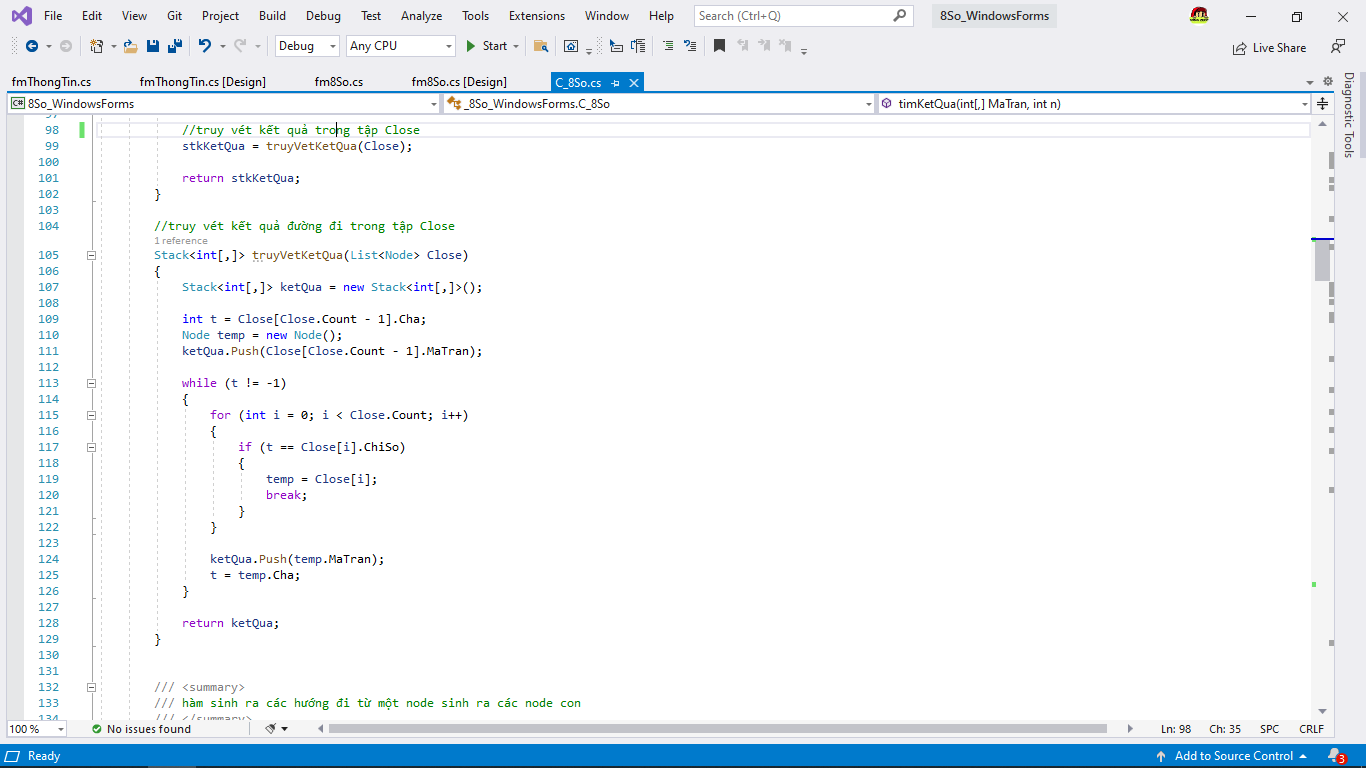
+ Khởi tạo các giá trị đối tượng tính toán:



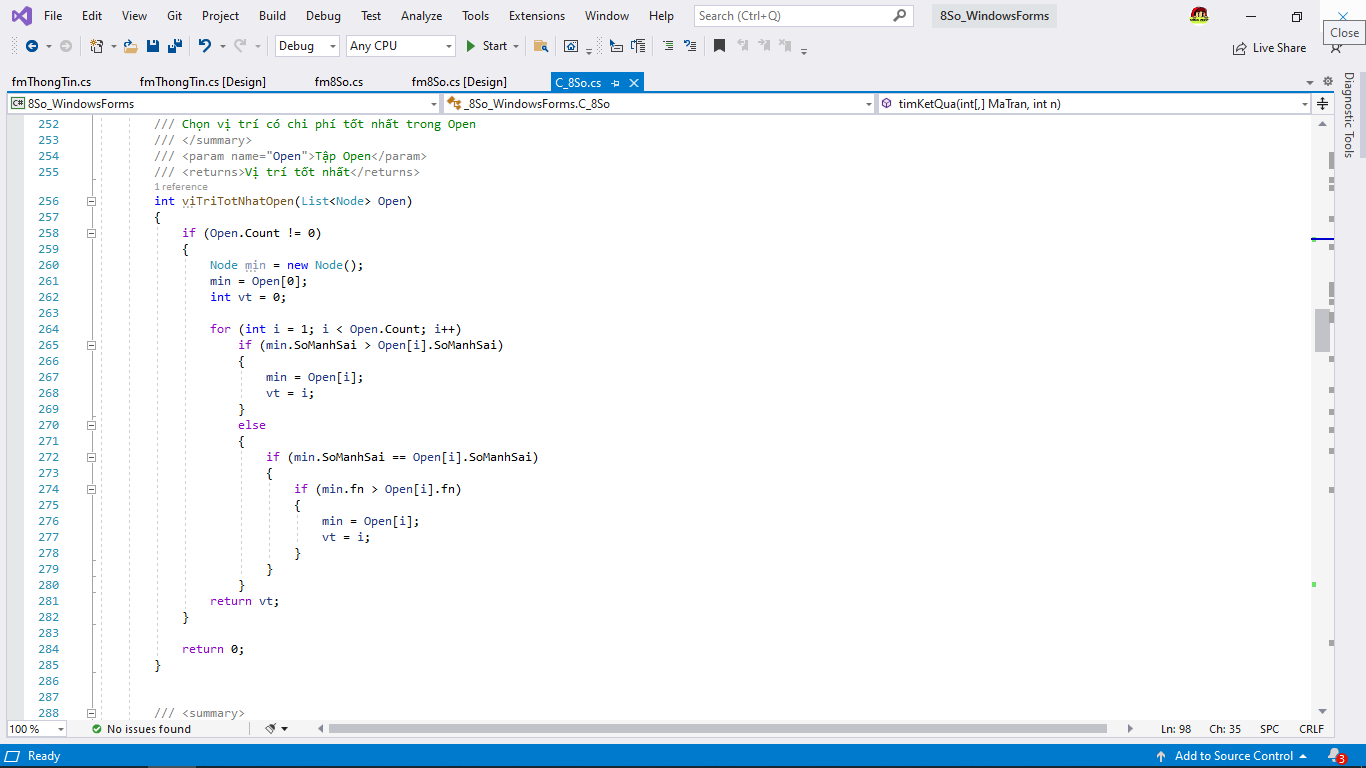
+ Điều chỉnh hướng di chuyển của đối tượng:



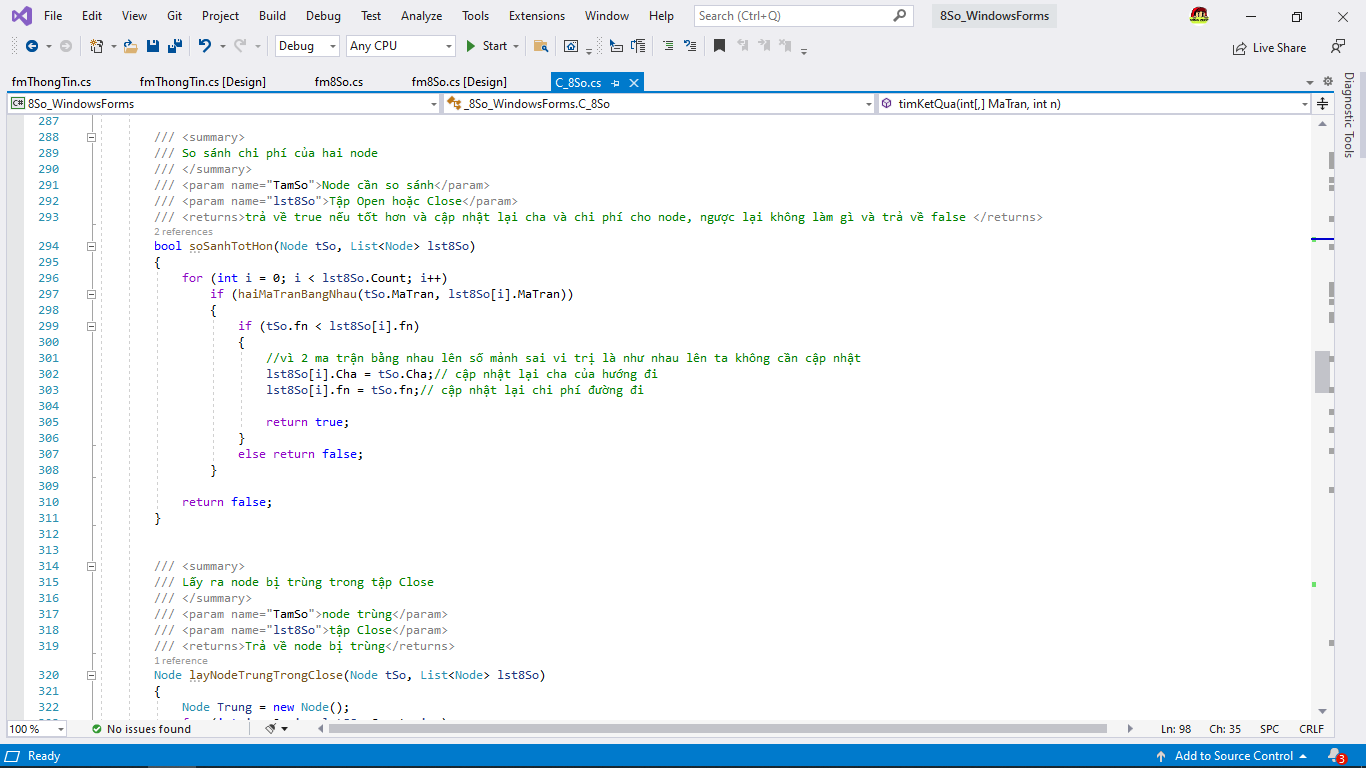




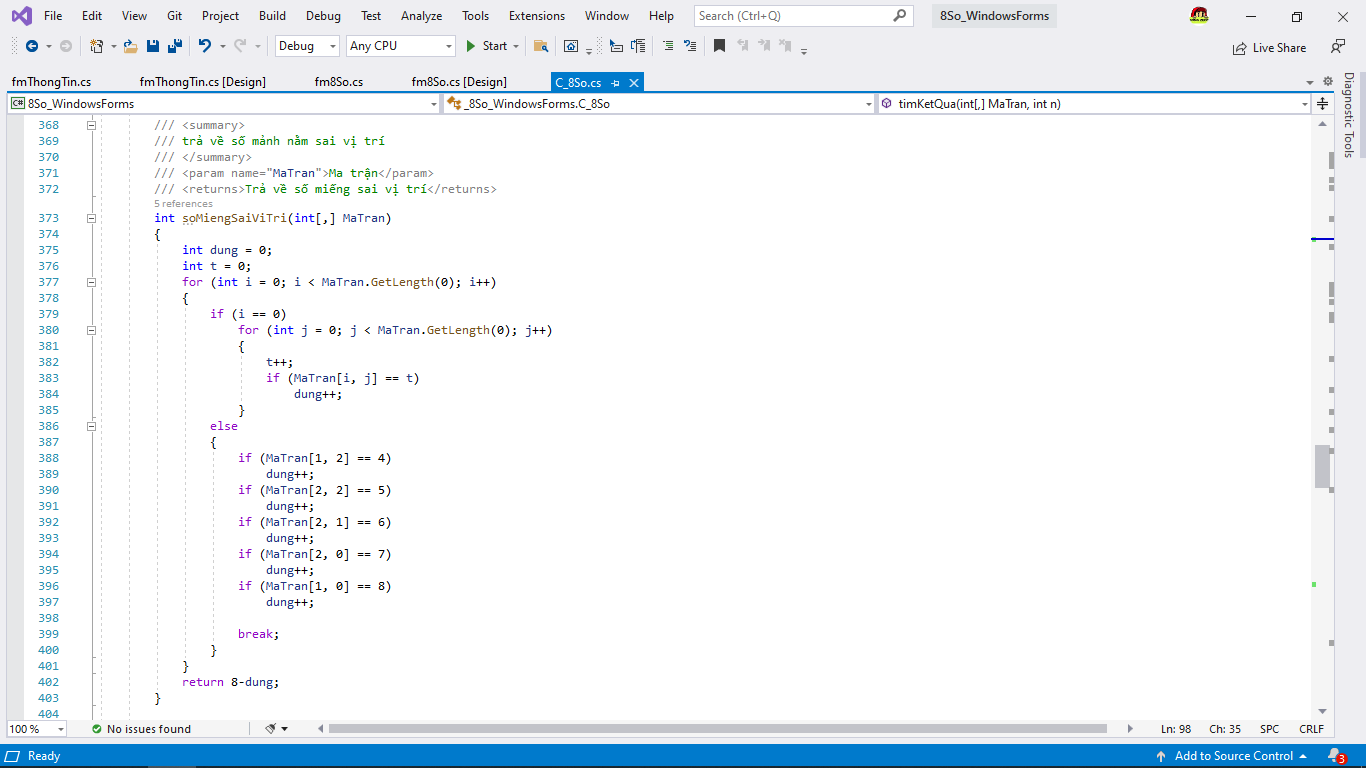
+ Chọn vị trí chi phí tốt nhất:



+ So sánh vị trí Node:



+ Trả về vị trí mảnh nằm sai:



Chú thích đã được thêm vào từng vùng code giảng viên có thể xem trong chương trình Demo.

Hình ảnh Demo:

|  |  |
| --- | --- |
| Màn hình khởi đầu | Màn hình khi hoàn thành sắp xếp |
|  |  |

Có kèm video Demo.

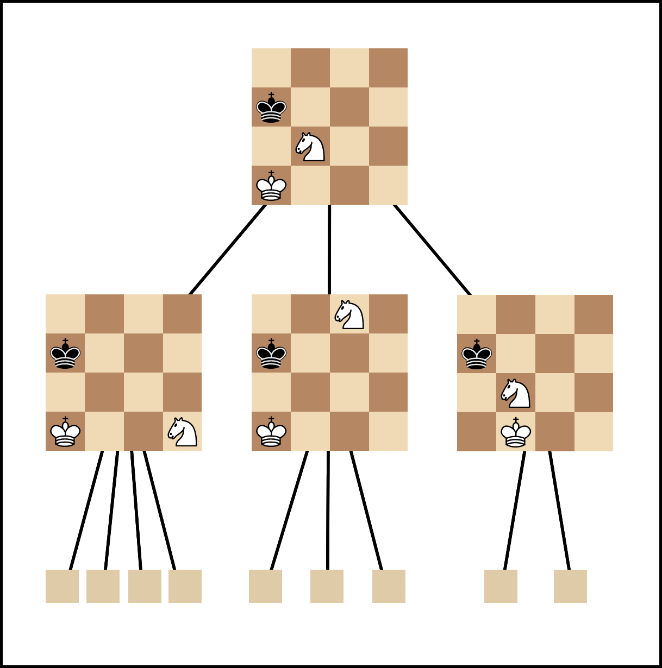
# 2. Theory of game (Lý thuyết về trò chơi)

## 2.1. Thuật toán Minimax

Minimax là giải thuật là một thuật toán đệ quy lựa chọn bước đi kế tiếp trong một trò chơi có hai người bằng cách định giá trị cho các Node trên cây trò chơi sau đó tìm Node có giá trị phù hợp để đi bước tiếp theo.

Quá trình chơi cờ là quá trình quân trắng và đen thay phiên nhau đưa ra quyết định, thực hiện một trong số các nước đi hợp lệ. Trên cây trò chơi, quá trình đó sẽ tạo ra đường đi từ gốc đến lá. Giả sử tới một thời điểm nào đó, đường đi đã dẫn tới đỉnh u. Nếu u là đỉnh trắng (đen) thì trắng (đen) cần chọn đi tới một trong các đỉnh đen (trắng) v là con của u. Tại đỉnh đen (trắng) v mà trắng (đen) vừa chọn, đen (trắng) sẽ phải chọn đi tới một trong các đỉnh trắng (đen) w là con của v. Quá trình trên sẽ dừng lại khi đạt tới một đỉnh là lá của cây.

Giả sử quân trắng cần tìm nước đi tại đỉnh u. Nước đi tối ưu cho quân trắng là nước đi dần tới đỉnh con của v là đỉnh tốt nhất(cho trắng) trong số các đỉnh con của u. Ta cần giả thiết rằng, đến lượt đối thủ chọn nước đi từ v, đen cũng sẽ chọn nước đi tốt nhất cho a ta. Như vậy, để chọn nước đi tối ưu cho trắng tại đỉnh u, ta cần phải xác định giá trị các đỉnh của trò chơi gốc u. Giá trị của các đỉnh lá là giá trị của hàm kết cuộc. Đỉnh có giá trị càng lớn càng tốt cho trắng, đỉnh có giá trị càng nhỏ càng tốt cho đen, Để xác định giá trị các đỉnh của cây trò chơi gốc u, ta đi từ mức thấp nhất lên gốc u. Giả sử v là đỉnh trong của cây và giá trị các đỉnh con của nó đã được xác định. Khi đó nếu v là đỉnh trắng thì giá trị của nó được xác định là giá trị lớn nhất trong các giá trị của các đỉnh con. Còn nếu v là đỉnh đen thì giá trị nhỏ nhất trong các giá trị của các đỉnh con.



Để cài đặt kĩ thuật Minimax, việc gán giá trị cho các đỉnh được thực hiện bởi các hàm đệ quy Maxval và Minval. Hàm MaxVal xác định giá trị cho các đỉnh trắng, hàm MinVal xác định giá trị cho các đỉnh đen.

## 2.2. Tinh chỉnh Alpha-beta (cắt tỉa Alpha- beta)

Việc tinh chỉnh Alpha-beta là một phương pháp tối ưu hóa thuật toán Minimax cho phép chúng ta bỏ qua một số hướng trong tất cả hướng đi có thể. Điều này giúp ta dự đoán hướng bằng minimax hiệu quả, trong khi sử dụng cùng một thuật toán.

Việc giảm thiểu alpha-beta dựa trên tình huống mà chúng ta có thể ngừng đưa ra hướng đi nếu chúng ta thấy hướng đi đó dẫn đến một kết quả không mong đợi. Việc điều chỉnh alpha-beta làm cho thuật toán nhanh hơn.

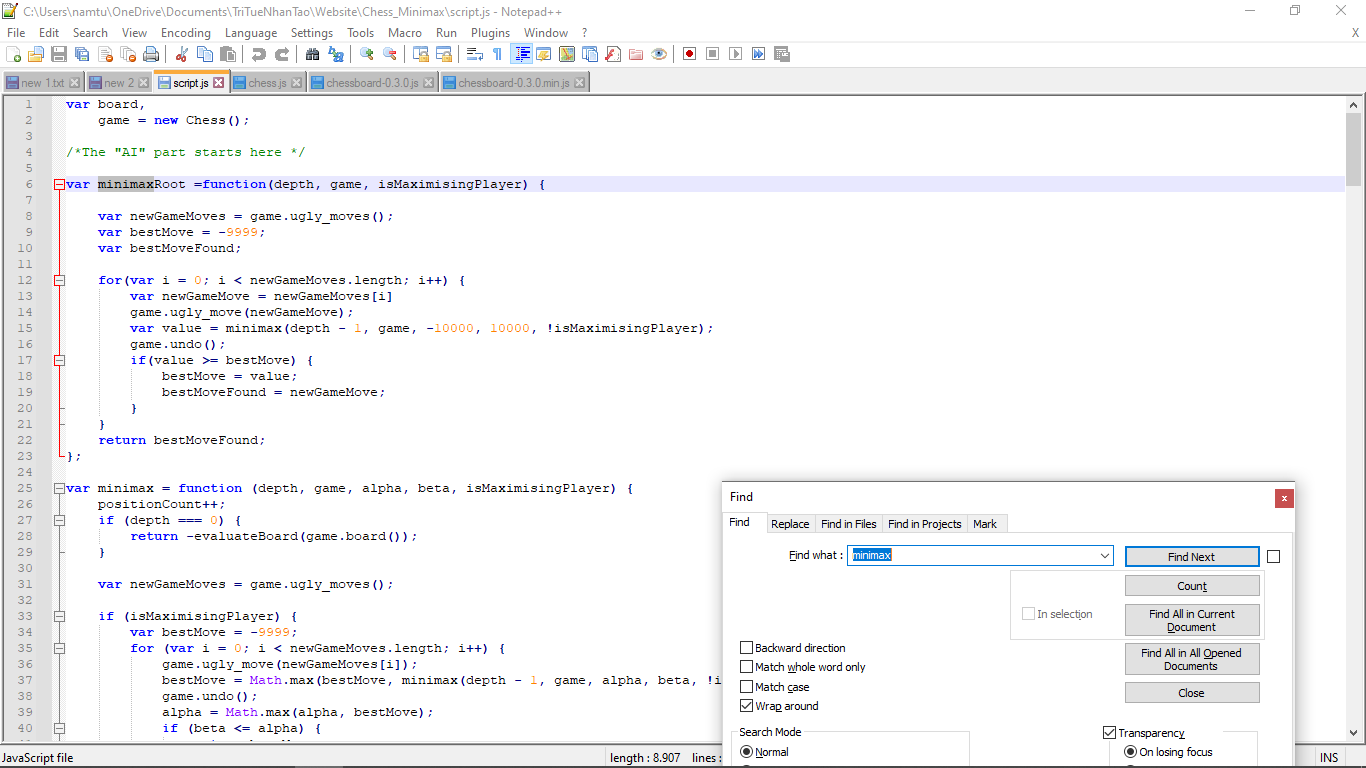
## 2.3. Chương trình demo: Cờ vua dùng thuật toán Minimax

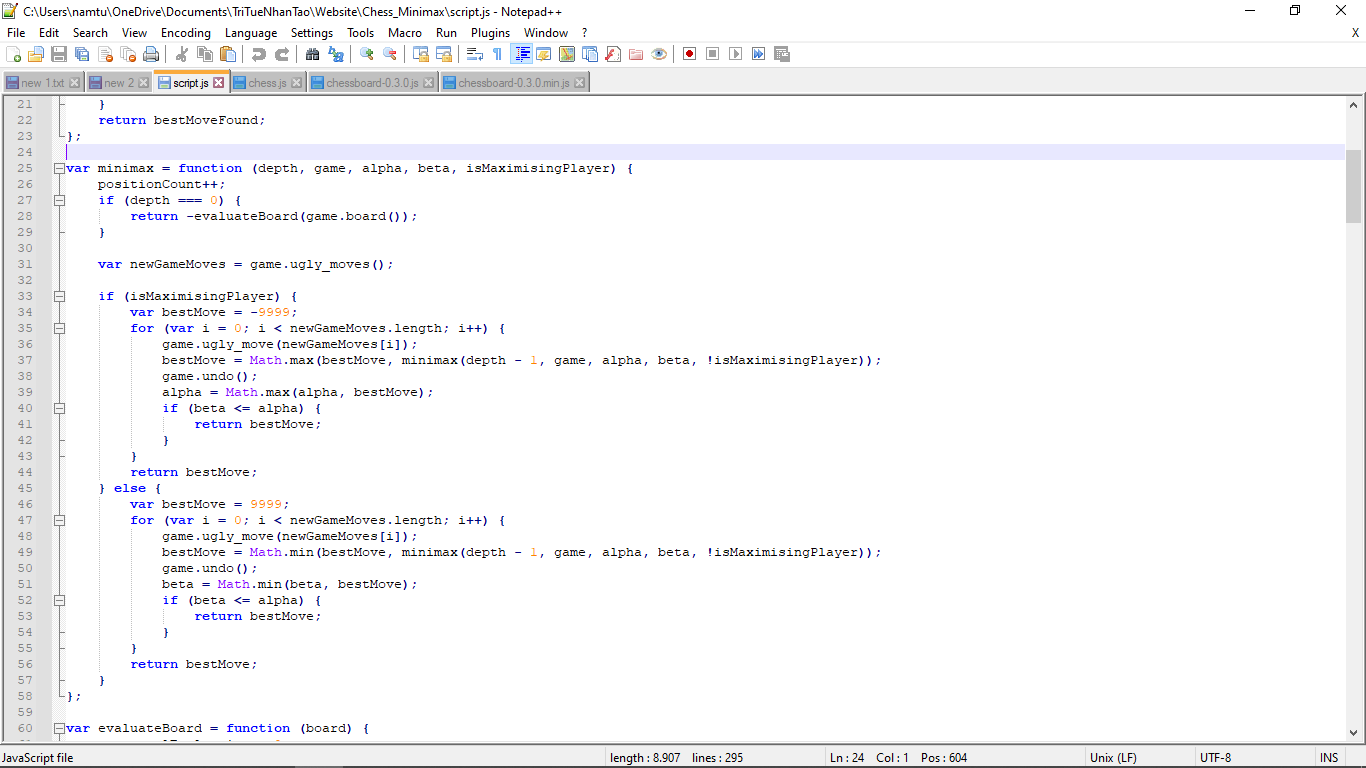
Ngôn ngữ sử dụng: Javascript

Giao diện: Website (HTML+CSS)

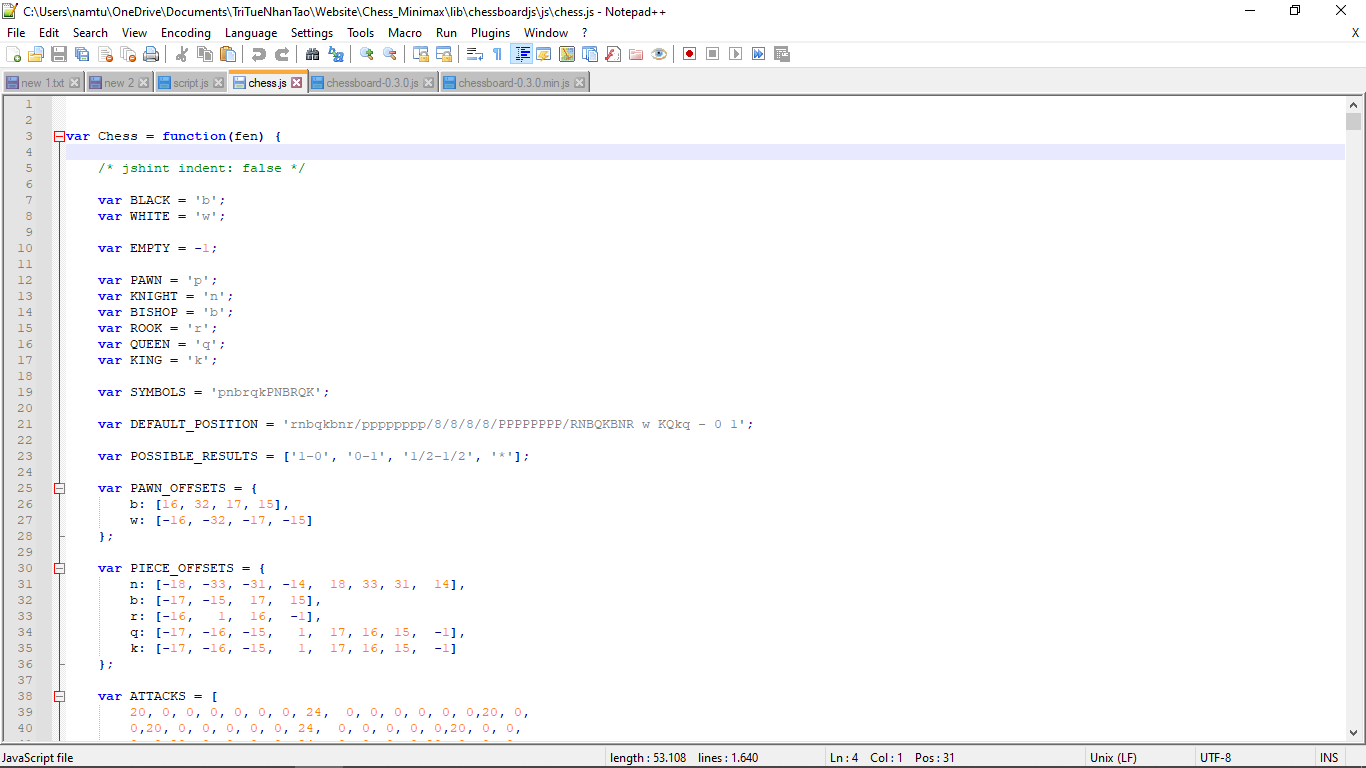
Một số hình ảnh code:

Triển khai thuật toán Minimax: kèm theo các giá trị liên quan như alpha, beta (hỗ trợ cắt nhánh dư thừa) - file nguồn script.js.

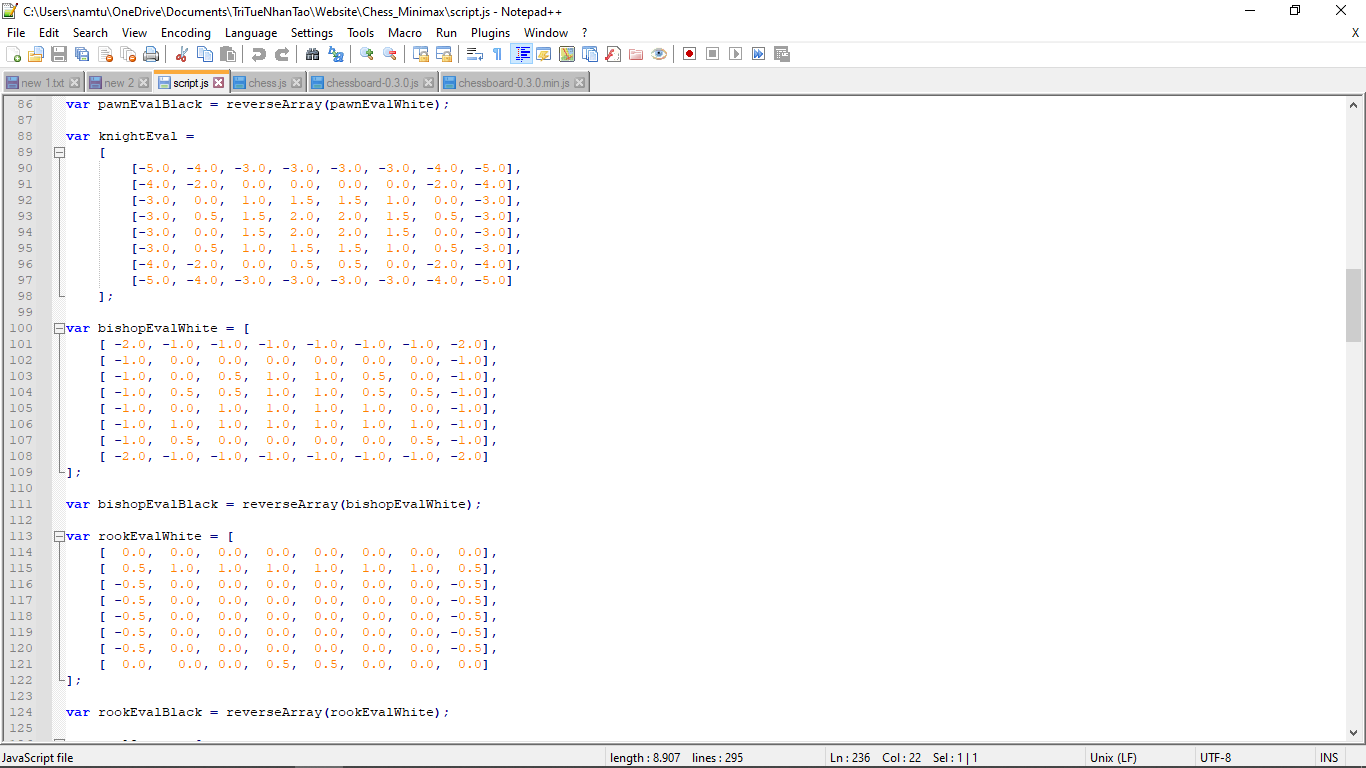




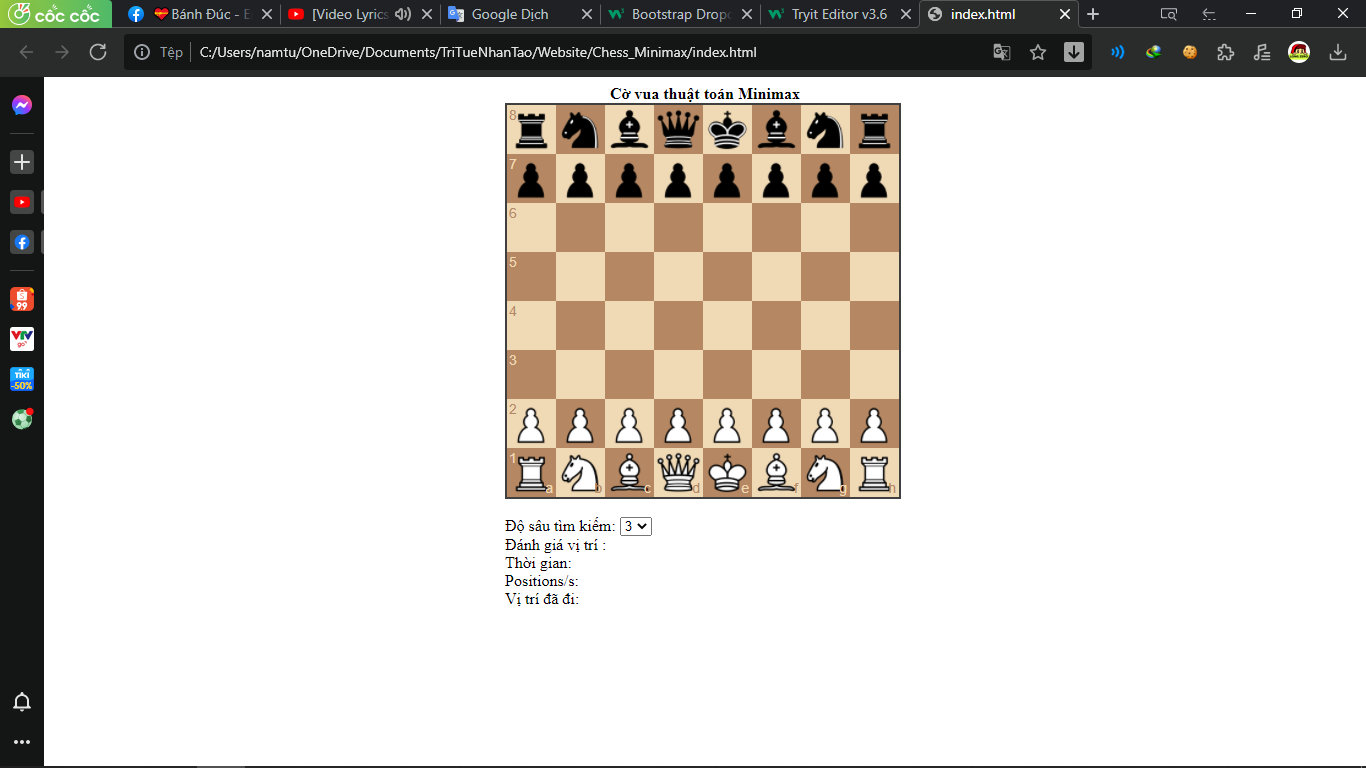
Khai báo các đối tượng trên bàn cờ: màu cờ (trắng-đen), loại cờ (tốt, xe, mã, tượng(tịnh), hậu, vua) – file nguồn chess.js.



Lưu những vị trí tối ưu nhất (theo [Wikipedia](https://en.wikipedia.org/wiki/Chess)) dưới dạng mảng nhằm đạt được hiệu quả cao nhất (những vị trí nằm giữa bàn cờ sẽ cho nhiều lựa chọn – đường đi tối ưu hơn) – file nguồn script.js.



Hình ảnh Demo:



Video Demo được lưu trong thư mục Video.

# 3. Logic and Semantic (Logic và ngữ nghĩa)

## 3.1. Giải thuật Vương Hạo

Thuật giải Vương Hạo là một thuật giải khác đề chứng minh việc suy dẫn mệnh đề (có cùng mục đích với Robinson):

**Bước 1:** Đưa bài toán cần chứng minh về dạng chuẩn:

**GT1 ,GT2 ,...,GTn ⇒ KL1 , KL2 ,..., KLm**

Trong đó các GTi và j KL là các câu chỉ gồm các phép ∧ , ∨ , ¬, không chứa phép ⇒

hay ⇔ . Lưu ý: dấu phẩy (,) ở vế trái tương đương với ∧ , ở vế phải tương đương với

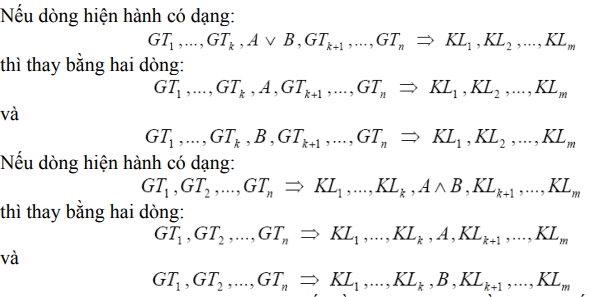
∨.

**Bước 2:** Nếu tồn tại một câu có phép ¬ ở đầu thì chuyển vế câu và loại bỏ phép ¬

**Bước 3:** Thay các dấu ∧ ở vế trái và các dấu ∨ ở vế phải bằng dấu phẩy (,). Khi đó vế

trái chỉ còn dấu ∨ và ¬ , về phải chỉ còn dấu ∧ và ¬ .

**Bước 4:**



**Bước 5:** Một dòng được chứng minh nếu tồn tại một mệnh đề ở cả hai vế.

**Bước 6:** Một dòng không thể tách cũng không thể chuyển vế dấu ¬ mà không có biến

mệnh đề chung ở cả hai vế thì không được chứng minh.

Lặp lại bước 2 đến 6 cho tới khi mọi dòng được chứng minh hay tồn tại một dòng không được chứng minh. Bài toán ban đầu được chứng minh nếu mọi dòng tách ra từ nó được chứng minh.

## 3.2. Chương trình demo: Mô phỏng thuật toán Vương Hạo trong Prolog

Ngôn ngữ sử dụng: Prolog

Giao diện: SWI-Prolog.

Cách sử dụng:

Một số kí tự trong mệnh đề khá khó khăn để sử dụng nên phần mềm chỉ nhận những kí tự tương đương trong bảng:

|  |  |  |
| --- | --- | --- |
| **Phép tính mệnh đề** | **Kí hiệu toán học** | **Kí hiệu chương trình** |
| Phép phủ định (negation) | ¬ | ! |
| Phép hội (conjunction) | ∧ | & |
| Phép tuyển (disjunction) | ˅ | v |
| Phép kéo theo (implication) | → |  |
| Phép tương đương (equivalence) | ↔ |  |
| Phép tuần tự (sequent) | ⊢ | **|**= |

Một số hình ảnh code:

+ Xác định các toán tử

:-op(700,xfy,<->)

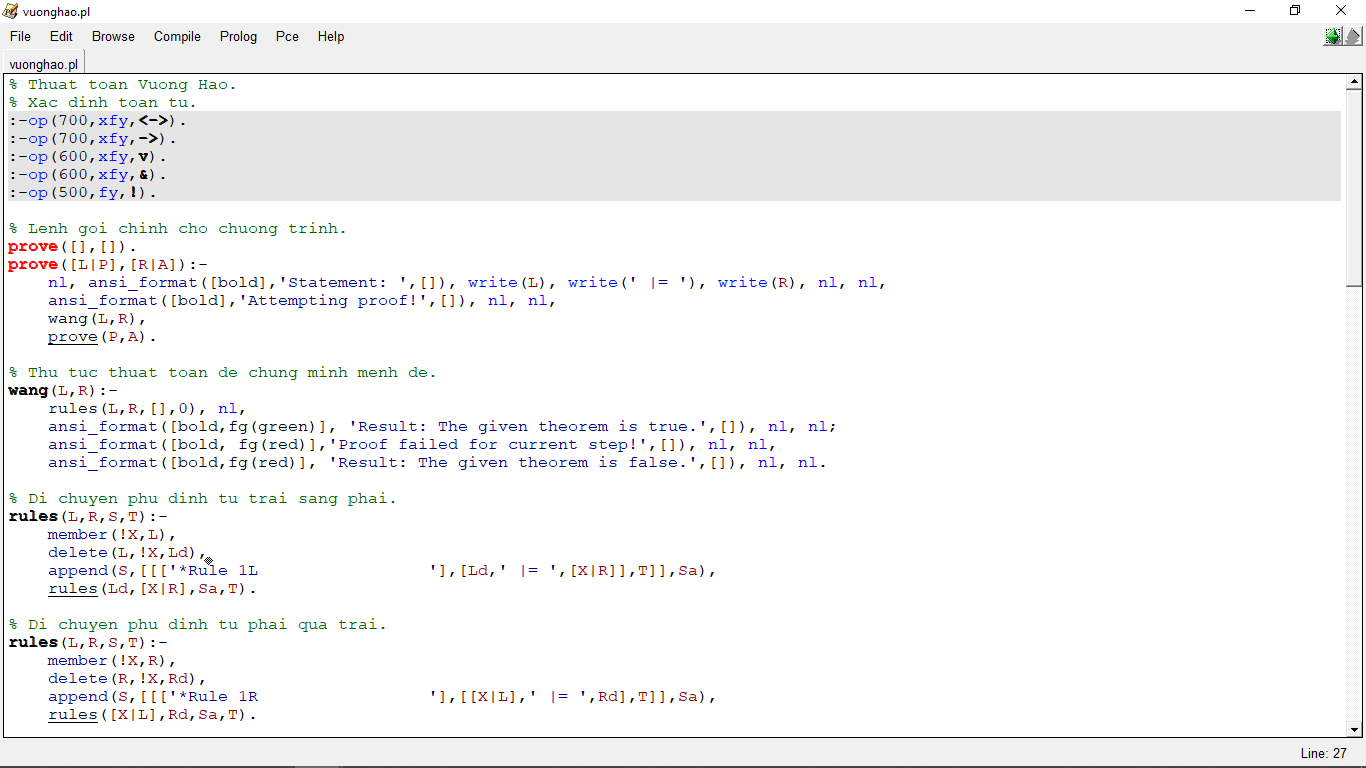
:-op(700,xfy,->)

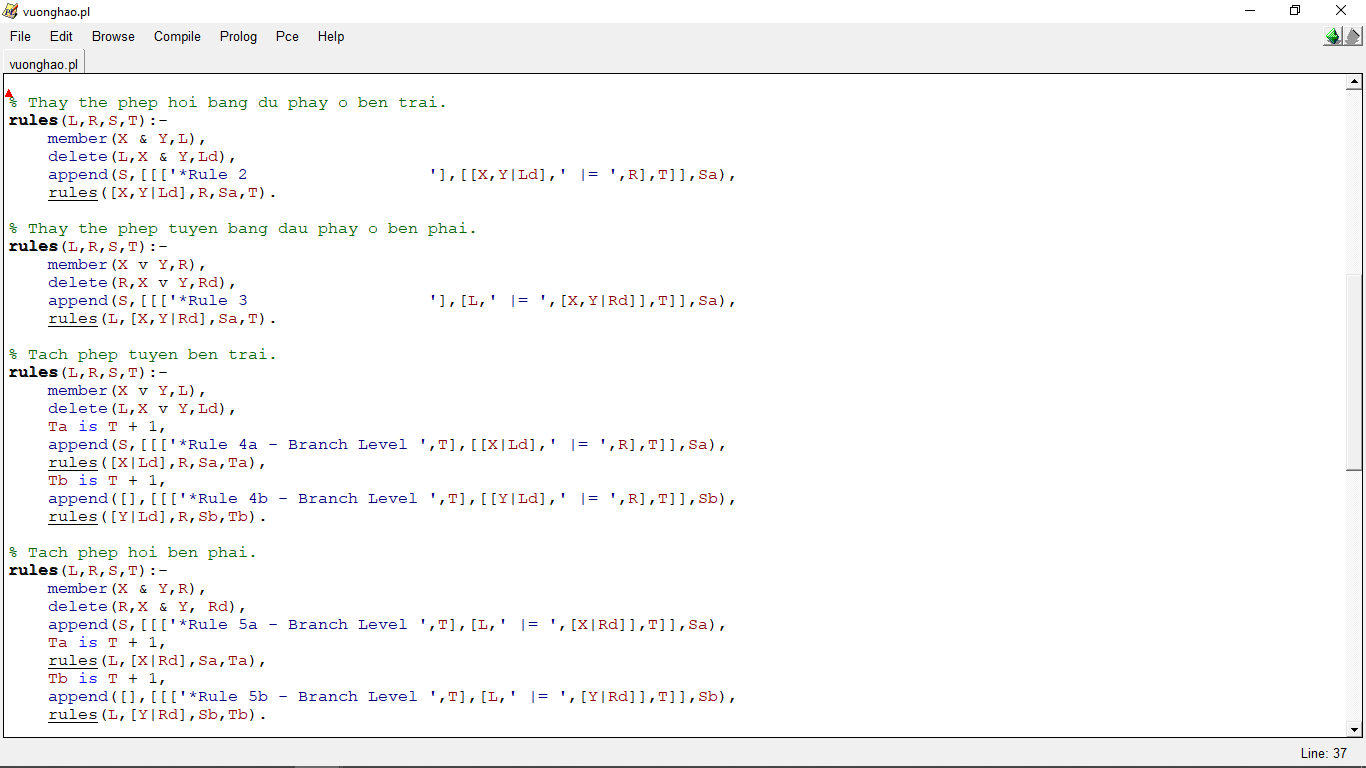
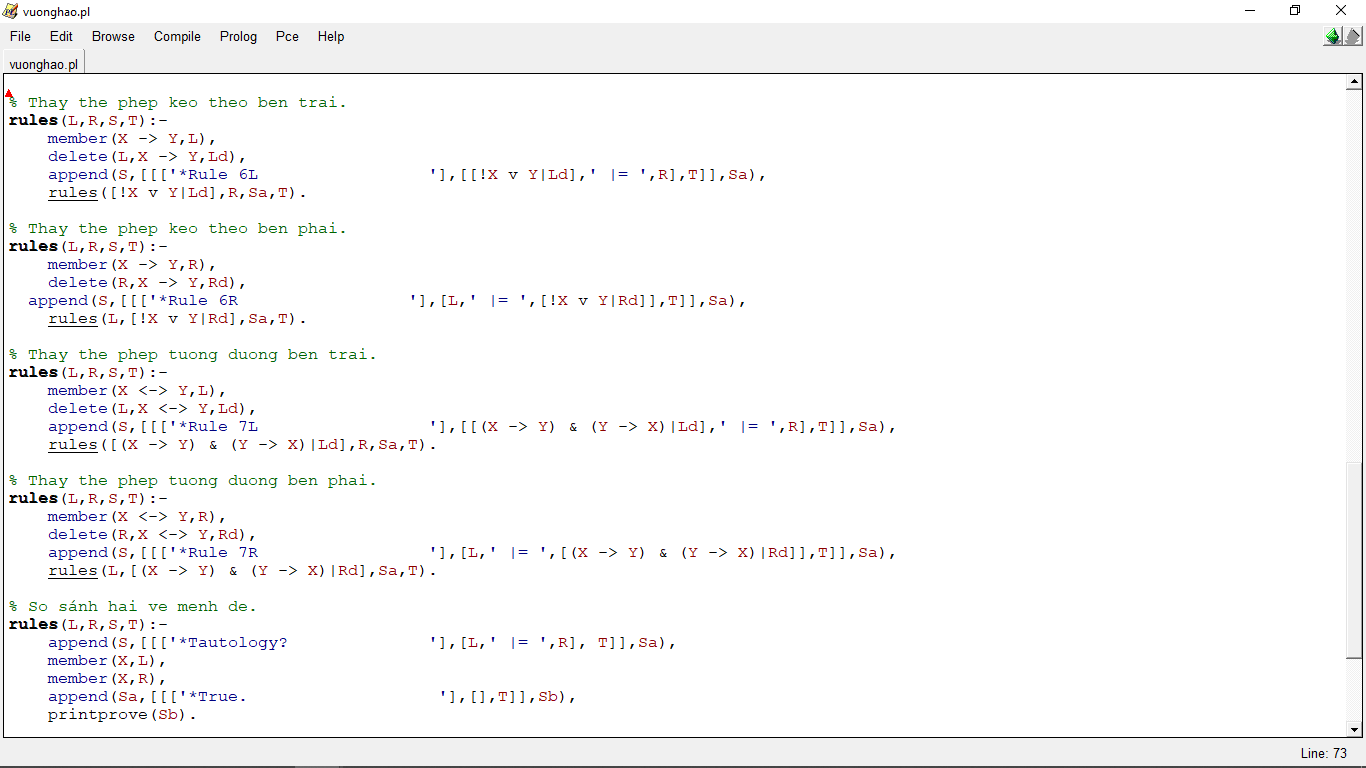
:-op(600,xfy,v)

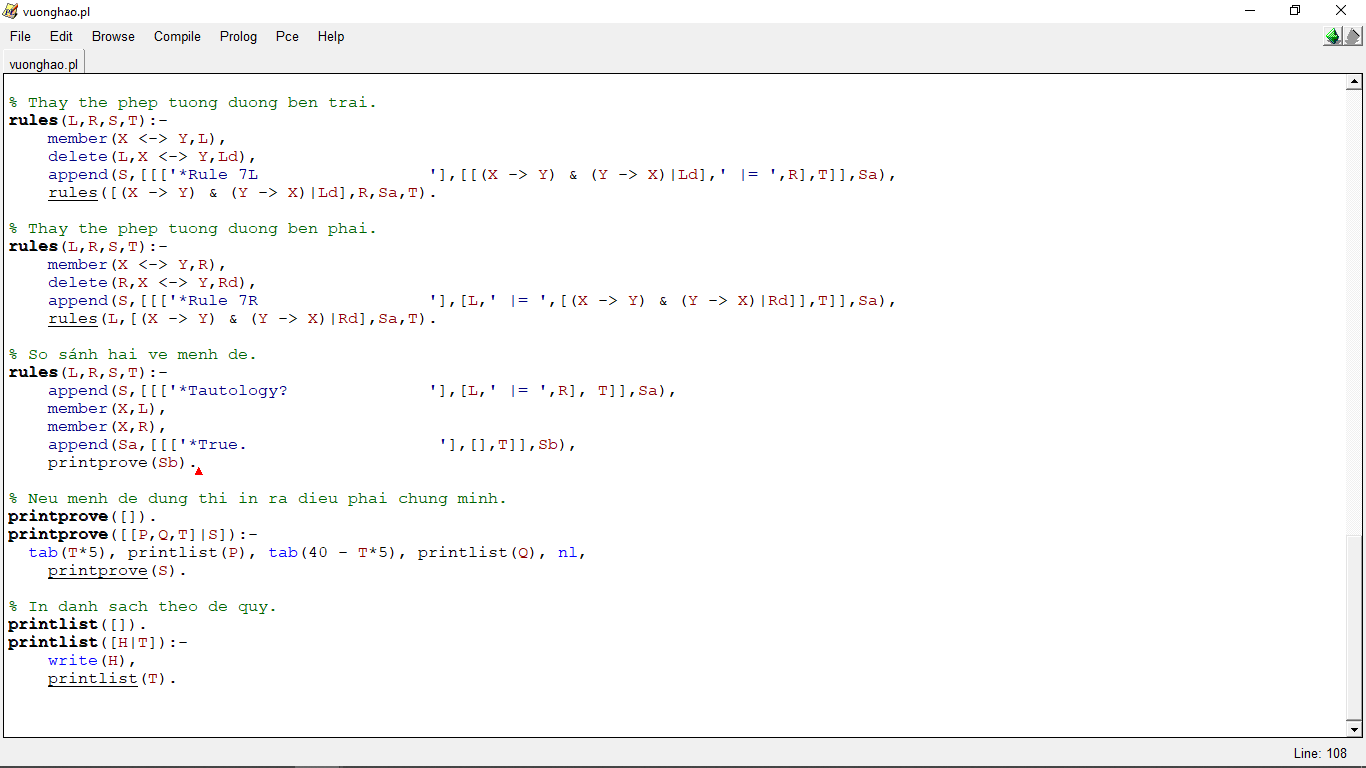
:-op(600,xfy,&)

:-op(500,fy,!)

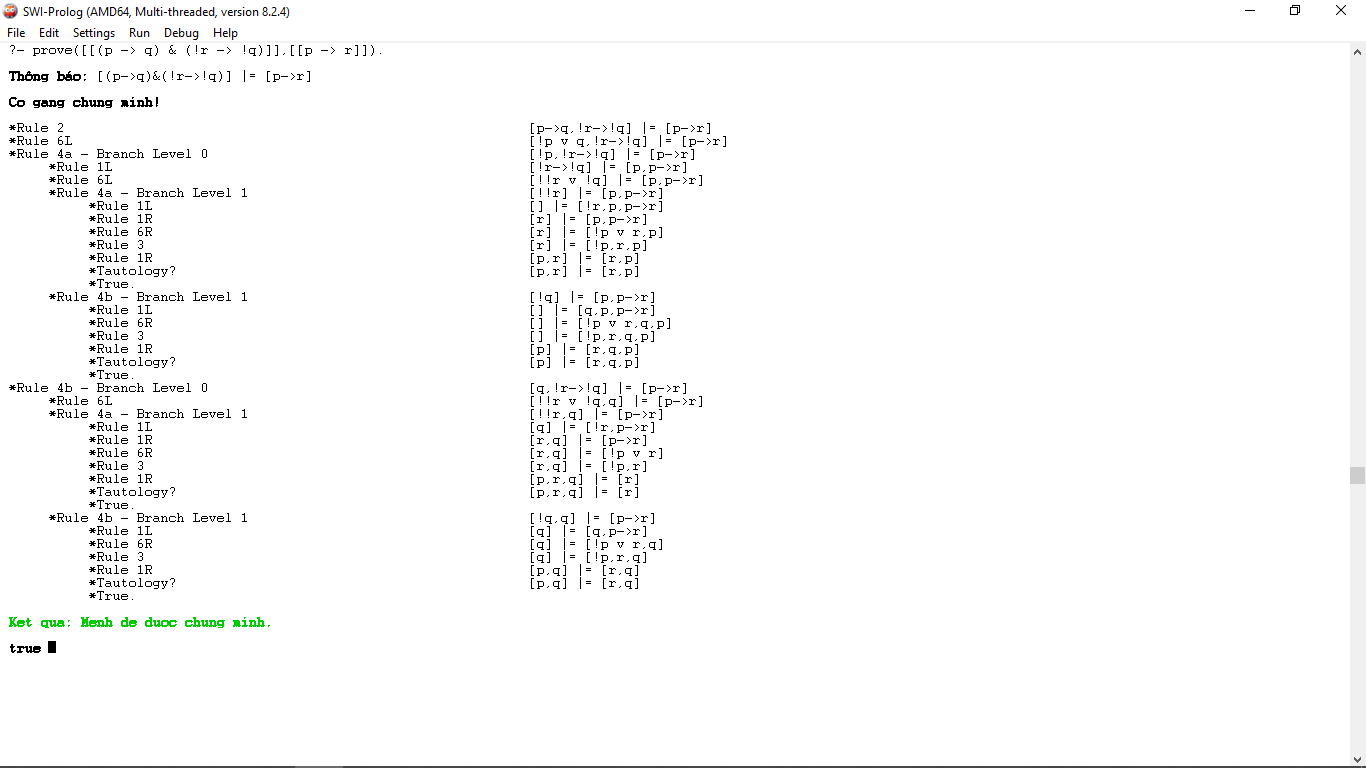
+ Thực thi 7 luật suy diễn trong thuật toán Vương Hạo.







Hình ảnh Demo:



Video thực hiện thử một số mệnh đề được lưu trong thư mục Video.

# 4. Simulation (Mô phỏng)

## 4.1. Genetic Algorithm applied to optimization ( Thuật toán di truyền áp dụng để tối ưu hoá)

Thuật toán di truyền (GA) là thuật toán tìm kiếm dựa trên các khái niệm về chọn lọc tự nhiên và di truyền. GAs là một tập con của một nhánh tính toán lớn hơn nhiều được gọi là Tính toán tiến hóa.

Trong GAs, ta có một nhóm các giải pháp khả thi cho vấn đề đã cho. Những dung dịch này sau đó trải qua quá trình tái tổ hợp và đột biến (giống như trong di truyền tự nhiên), tạo ra những đứa trẻ mới và quá trình này được lặp lại trong nhiều thế hệ khác nhau. Mỗi cá thể (hoặc giải pháp ứng cử viên) được chỉ định một giá trị phù hợp (dựa trên giá trị hàm mục tiêu của nó) và các cá thể lai tạo có cơ hội cao hơn để giao phối và sinh ra các cá thể phù hợp. Điều này phù hợp với Học thuyết của Darwin về sự sống còn của những cá thể khoẻ mạnh nhất.

Do đó, nó tiếp tục phát triển các cá nhân hoặc giải pháp tốt hơn qua nhiều thế hệ, cho đến khi đạt được tiêu chí dừng. Thuật toán di truyền về bản chất là đủ ngẫu nhiên, nhưng chúng hoạt động tốt hơn nhiều so với tìm kiếm cục bộ ngẫu nhiên (nơi ta chỉ thử các giải pháp ngẫu nhiên, theo dõi các giải pháp tốt nhất cho đến nay), vì chúng cũng khai thác thông tin lịch sử.

## 4.2. Chương trình demo: Tìm đường đi sử dụng thuật toán di truyền

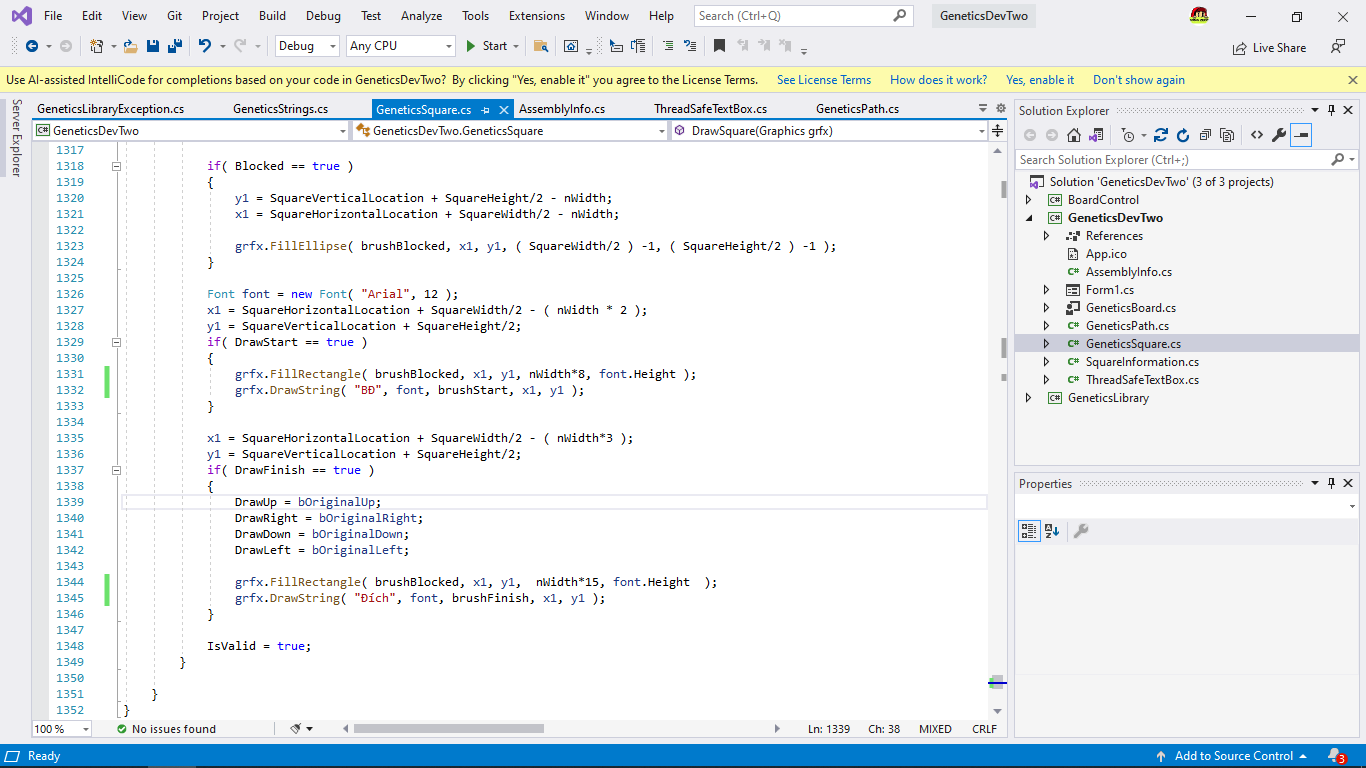
Ngôn ngữ sử dụng: C#

Giao diện: Winform

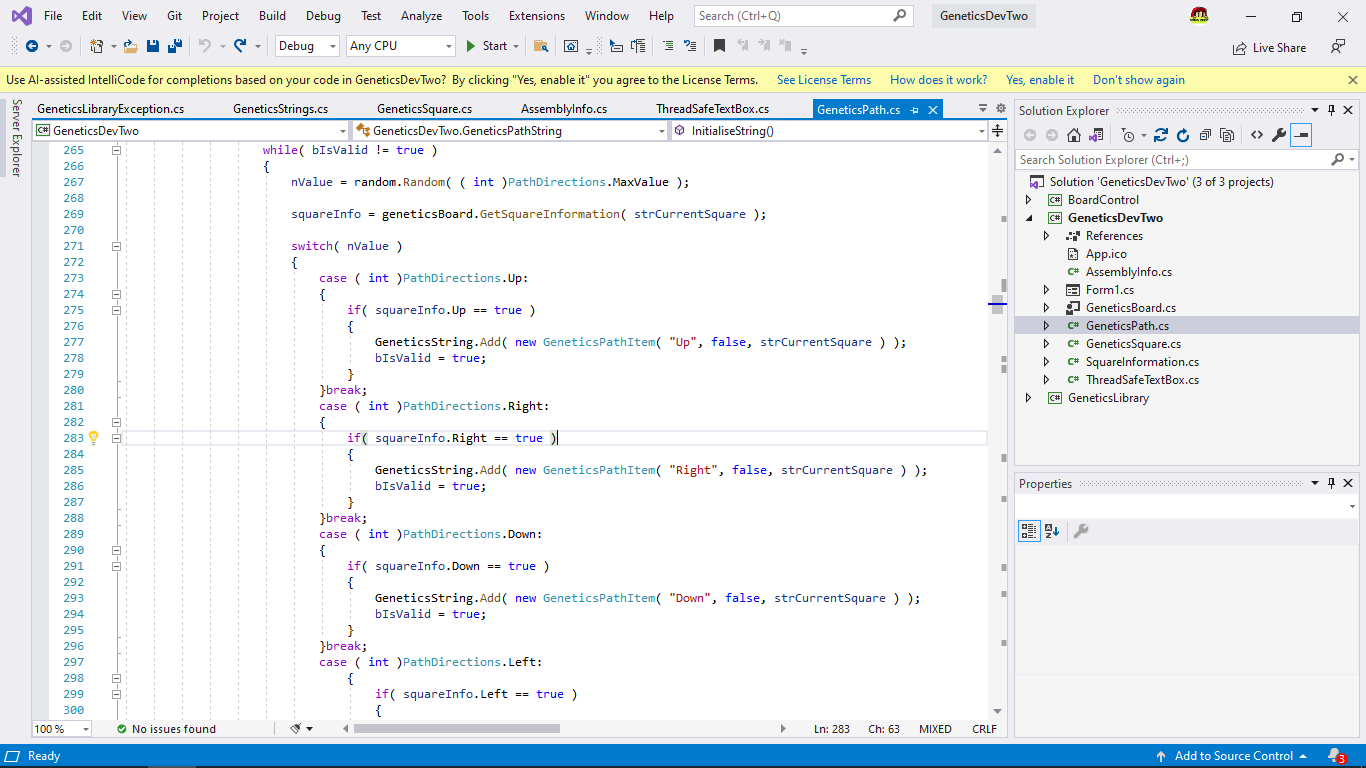
Mô tả: Project gồm nhiều file chức năng khác nhau phân là 3 mục chính: Control ( điều khiển), GeneticsDevTwo (Chương trình chính) và Library (Thư viện chứa các chức năng của thuật toán).

Một số hình ảnh code:

+ Khai báo thông số cho điểm Bắt đầu (BĐ), điểm Đích cùng một vài thông số về vị trí – file GeneticsSquare.cs.



+ Thực thi các đường đi một cách ngẫu nhiên theo 4 hướng (tràn ra 4 hướng theo quy mô đối tượng) – file GeneticsPath.cs.

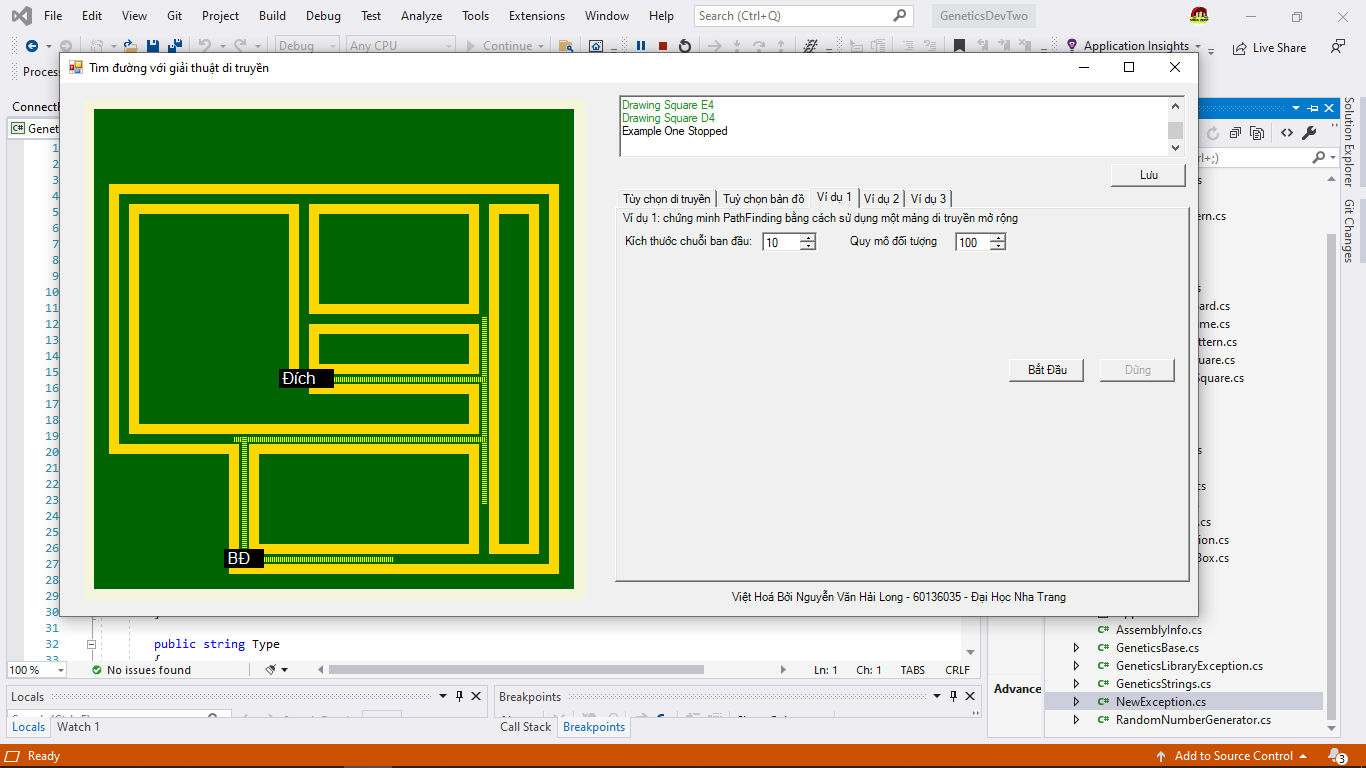


+ Các file thực thi khác:

- ThreadSafeTextBox.cs : TextBox chứ thông tin tìm đường (hướng đi).

- SquareInformation.cs: Nơi khai báo các biến mang giá trị Boolean.

Hình ảnh Demo:



Video demo được lưu trữ trong thư mục Video.

# 5. IoT and Robotics applied in with AI (IoT và Robotics được ứng dụng với AI)

## 5.1. Sử dụng Arduino để chế tạo cánh tay robot

Do tình hình dịch phức tạp nên em không có sử dụng được demo bằng phần cứng nên em gửi các file liên quan đến về chủ đề.

Các thiết bị cần thiết:

* Arduino UNO
* Module Bluetooth HC05
* Động cơ RC Servo MG996
* Động cơ RC Servo 9G
* Điện trở 1K Ohm
* Điện trở 2K Ohm
* Adapter 5V - 2A
* Jack cắm DC cái
* Breadboard 400 chân
* Dây bus đực - đực

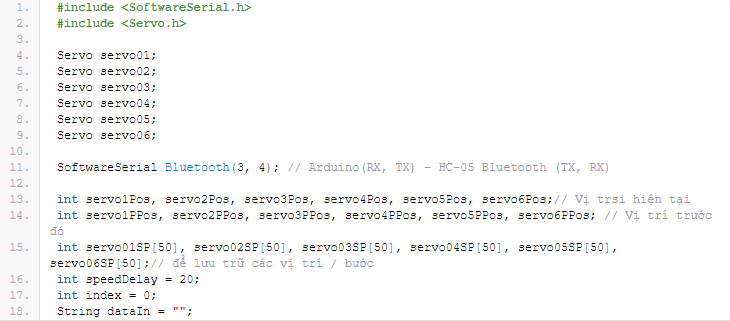
Phần mềm yêu cầu:

* **Arduino IDE:** Arduino IDE là một phần mềm lập trình dựa trên ngôn ngữ lập trình C. Bằng cách viết những câu lệnh có cấu trúc tương tự cấu trúc câu lệnh trong C, chúng ta có thể điều khiển hoặc thu nhận các dữ liệu từ các module, cảm biến thông qua vi điều khiển arduino.
* **MIT App Inventor:** là một ứng dụng web nguồn mở ban đầu được cung cấp bởi Google và hiện tại được duy trì bởi Viện Công nghệ Massachusetts (MIT). Nền tảng cho phép nhà lập trình tạo ra các ứng dụng phần mềm cho hệ điều hành Android (OS). Bằng cách sử dụng giao diện đồ họa, nền tảng cho phép người dùng kéo và thả các khối mã (blocks) để tạo ra các ứng dụng có thể chạy trên thiết bị Android. Nguồn trang: http://ai2.appinventor.mit.edu.

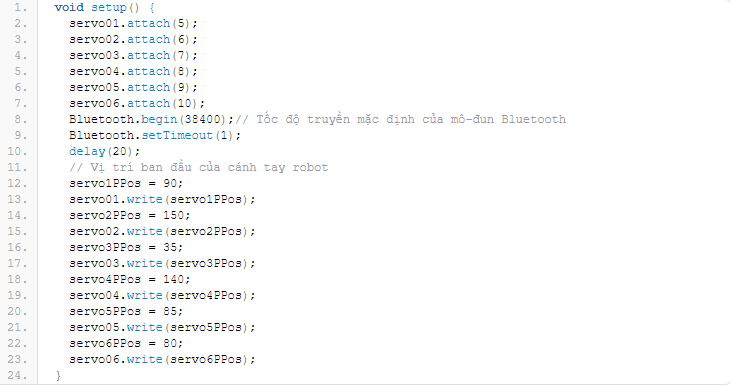
## 5.2. Chương trình demo: Cánh tay robot

**Lập trình với Arduino IDE:**

Đầu tiên ta khai báo thư viện SoftwareSerial cho giao tiếp nối tiếp của mô-đun Bluetooth cũng như thư viện servo. Cả hai thư viện này đều được tích hợp sẵn với Arduino IDE nên ta không cần phải cài đặt chúng bên ngoài. Sau đó, ta cần xác định sáu servos, mô-đun Bluetooth HC-05 và một số biến để lưu trữ vị trí hiện tại và trước đó của servos, cũng như các mảng để lưu trữ các vị trí hoặc các bước cho chế độ tự động.



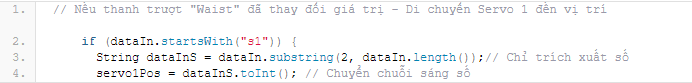
Trong phần thiết lập, ta cần khởi tạo servos và mô-đun Bluetooth và di chuyển cánh tay robot về vị trí ban đầu của nó. Sử dụng hàm write () di chuyển servo đến bất kỳ vị trí nào từ 0 đến 180 độ.



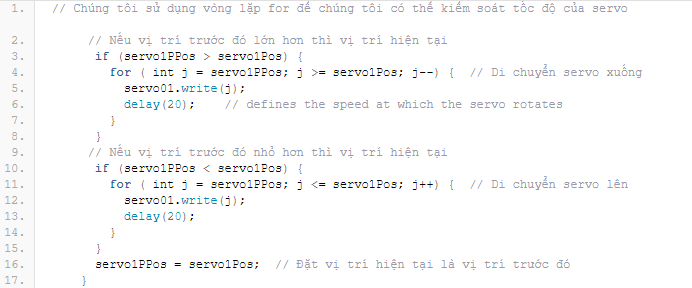
Tiếp đó khởi tạo vòng lặp, bằng cách sử dụng chức năng Bluetooth.available (),chương trình sẽ liên tục kiểm tra xem có bất kỳ dữ liệu nào đến từ Điện thoại thông minh hay không. Nếu đúng, sử dụng hàm readString (), chương trình đọc dữ liệu dưới dạng chuỗi và lưu trữ nó vào biến dataIn. Tùy thuộc vào dữ liệu đã đến, chương trình sẽ điều khiển cho cánh tay robot biết phải làm gì.



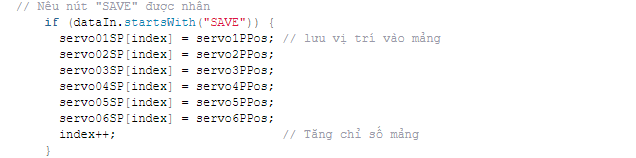
Sử dụng hàm startedWith (), chương trình kiểm tra tiền tố của mỗi dữ liệu đến và vì vậy nó biết phải làm gì tiếp theo.



Gọi hàm write () và servo sẽ đi đến vị trí đó, nhưng theo cách đó, servo sẽ chạy ở tốc độ tối đa, quá sức đối với cánh tay robot. Thay vào đó, ta cần kiểm soát tốc độ của servo, vì vậy nên sử dụng một số vòng lặp FOR để dần dần di chuyển servo từ vị trí trước đó đến vị trí hiện tại bằng cách triển khai thời gian trễ giữa mỗi lần lặp. Bằng cách thay đổi thời gian trễ, ta có thể thay đổi tốc độ của servo.



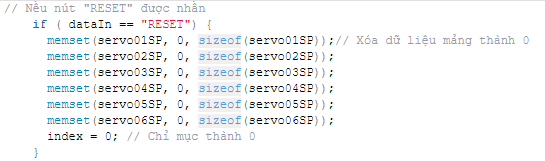
Thiết đặt lưu thông tin. Nếu chúng ta nhấn nút SAVE, vị trí của mỗi động cơ servo được lưu trong một mảng. Với mỗi lần nhấn, chỉ số tăng lên vì vậy mảng được lấp đầy từng bước.



Khai báo nút RUN và thao tác thực thi, gọi hàm tùy chỉnh runningervo () chạy các bước được lưu trữ. Ở đây chương trình chạy đi chạy lại các bước đã lưu trữ cho đến khi chúng tôi nhấn nút RESET. Sử dụng vòng lặp FOR, chạy qua tất cả các vị trí được lưu trữ trong các mảng và đồng thời hàm kiểm tra xem có bất kỳ dữ liệu đến từ điện thoại thông minh hay không. Dữ liệu này có thể là nút RUN / PAUSE, nút này sẽ tạm dừng rô bốt và nếu được nhấp lại sẽ tiếp tục với các chuyển động tự động. Ngoài ra nếu điều khiển tốc độ, chương trình sẽ sử dụng giá trị đó để thay đổi thời gian trễ giữa mỗi lần lặp trong vòng lặp FOR bên dưới, điều khiển tốc độ của động cơ servo.

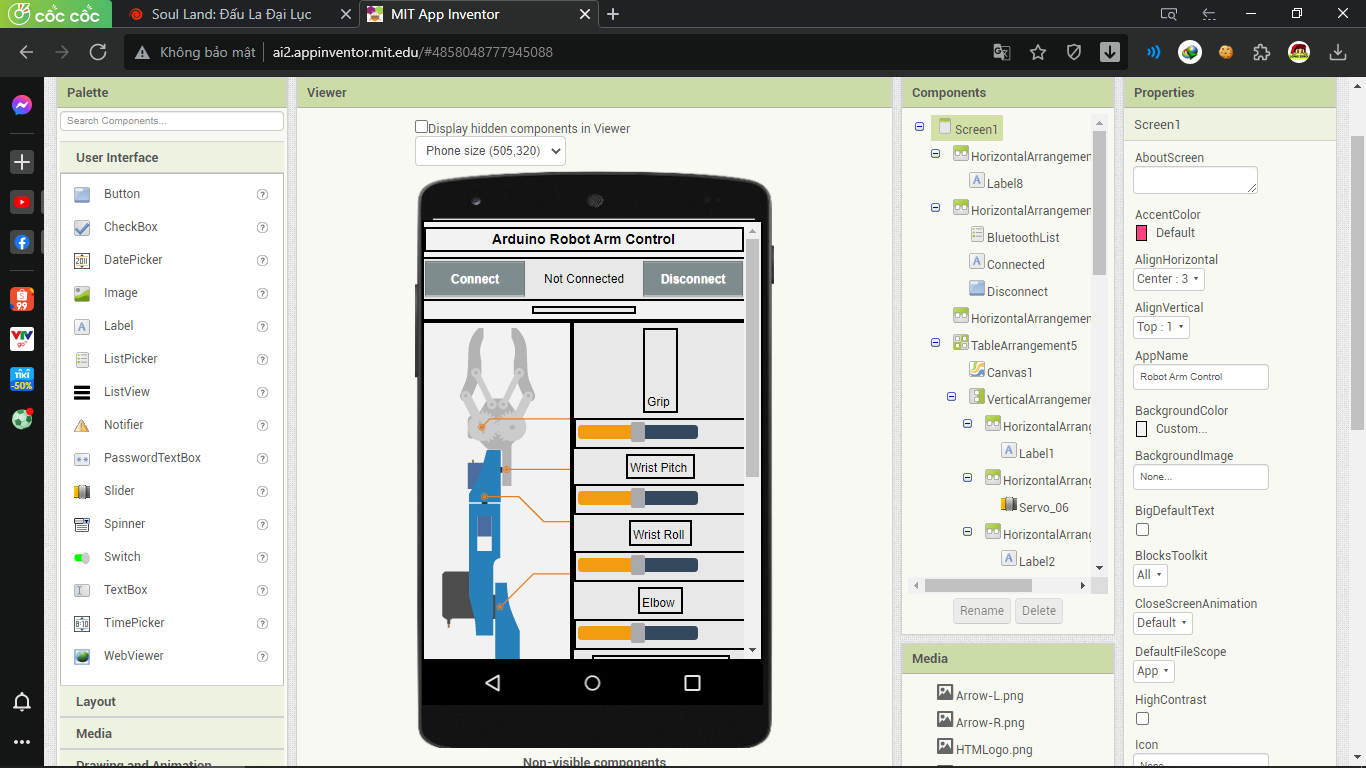


Di chuyển các servos đến vị trí tiếp theo của chúng. Sau cùng nếu nhấn nút RESET, chương trình sẽ xóa tất cả dữ liệu từ các mảng và cũng đặt lại chỉ mục về 0 để ta có thể thực hiện các chuyển động mới.

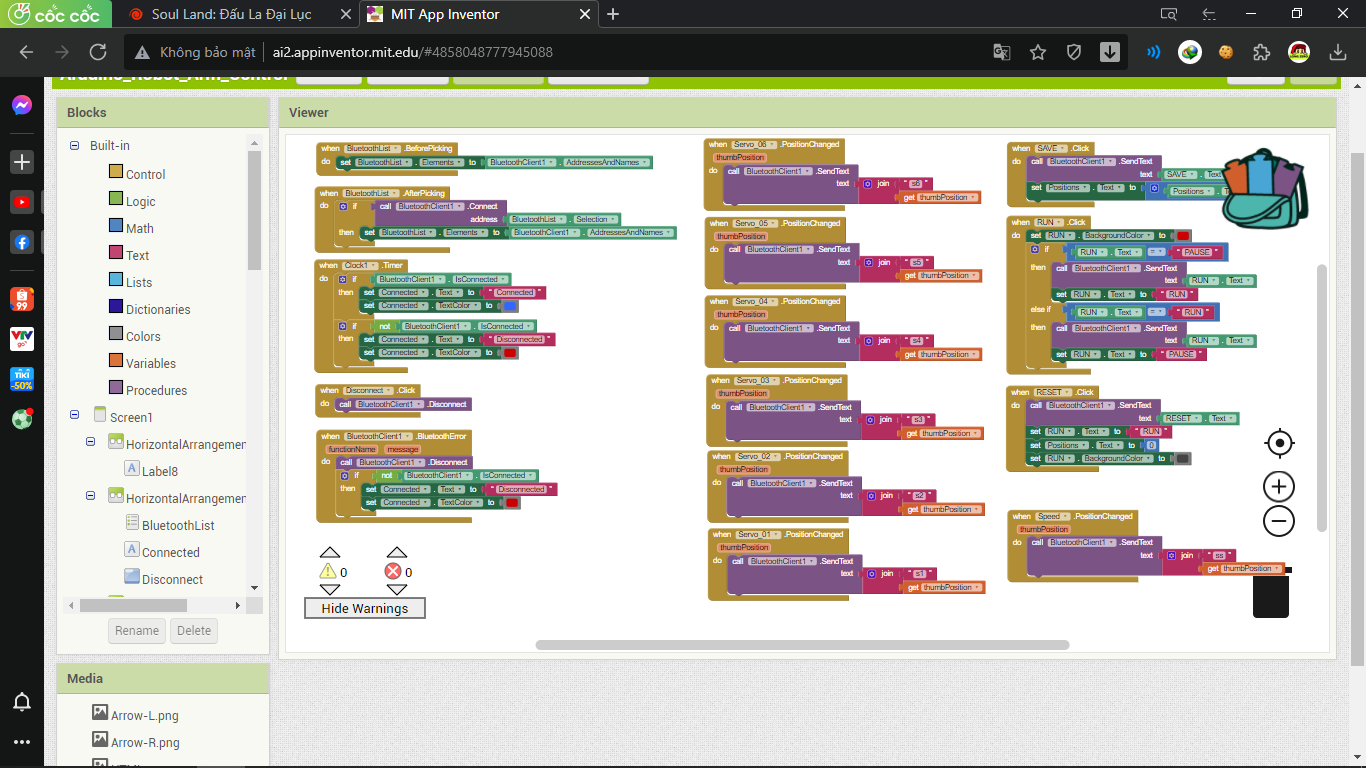


**Lập trình MIT App Inventor:**

Sử dụng biểu tượng bằng cách kéo thả để tạo giao diện



Xây dựng những khối lệnh logic trong Blocks



File thực thi được lưu trong thư mục B5