Trường Đại Học Khoa Học Tự Nhiên

Khoa Công Nghệ Thông Tin

Báo cáo đồ án

Phương pháp lập trình hướng đối tượng

Giáo viên hướng dẫn: Trương Toàn Thịnh

Nhóm 7

Lớp 17CTT3

List

1. Thông tin nhóm

MSSV	Họ và tên	Phân công
1712379	Đặng Thành Duy	Cấu trúc class và thiết kế đồ họa
1712369	Phạm Quốc Dũng	Cài đặt thuật toán
1712358	Nguyễn Minh Đức	Cấu trúc class và thiết kế đồ họa

2. Về thư viện đồ họa SFML

Nhận thấy lập trình trên console có rất nhiều giới hạn và rất không đẹp mắt nên nhóm em đã quyết định chọn xây dựng chương trình trên window32 application, thực hiện đồ họa bằng thư viện SFML.

SFML là thư viện đồ họa mã nguồn mở miễn phí được phát triển bởi cộng đồng. Chi tiết về download và cài đặt mong thầy hãy xem tại

https://www.sfml-dev.org/download/sfml/2.5.1/

SFML cung cấp bộ hàm về đồ họa, âm thanh, các thao tác xử bàn phím và chuột,... rất thích hợp cho việc phát triển game.

3. Tổng quan chương trình và giao diện

- Chương trình mang tên Caro được phát triển bởi nhóm 7.

- Các thao tác được thực hiện hoàn toàn bằng chuột (chỉ sử dụng bàn phím khi ghi tên file) bao gồm cả thao tác đánh trên bàn cờ.

- Luật trong Caro sử dụng luật chặn 2 đầu truyền thống của Việt Nam.

- Giao diện được kết hợp bởi hình nền, font text với nhiều font chữ khác nhau, màu sắc có thể nhấp nháy và các hàm đồ họa,..

- Có nhạc nền khi chơi game và có âm thanh khi click chuột.

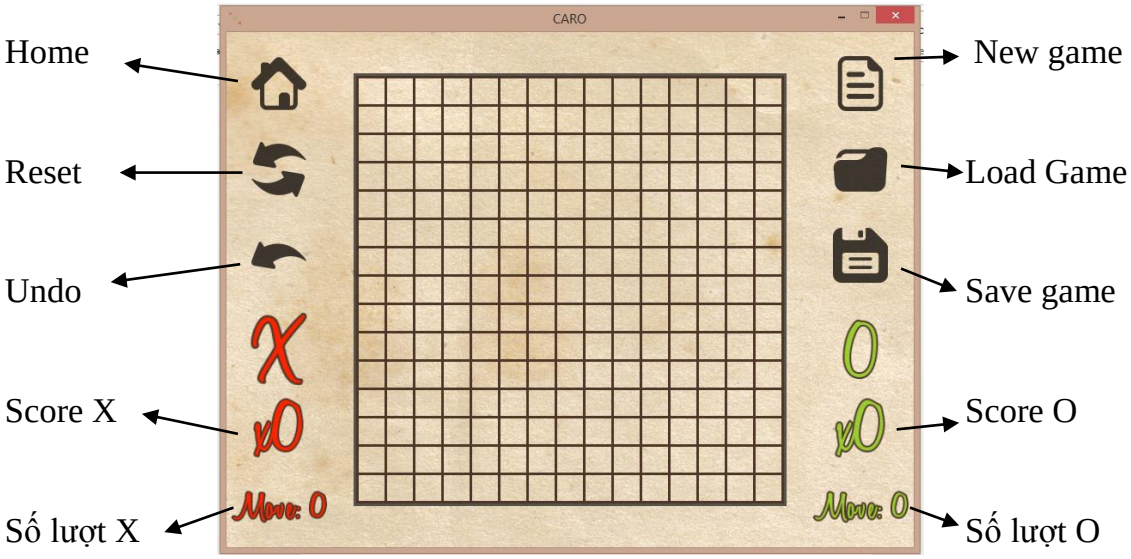
Giao diện màn hình chính

Thực hiện tương tác với con trỏ chuột: khi di chuột đến dòng text có thể click thì kích cỡ dòng text sẽ thay đổi, khi di chuột ra ngoài vùng có thể click thì kích cỡ text trở về dạng ban đầu.

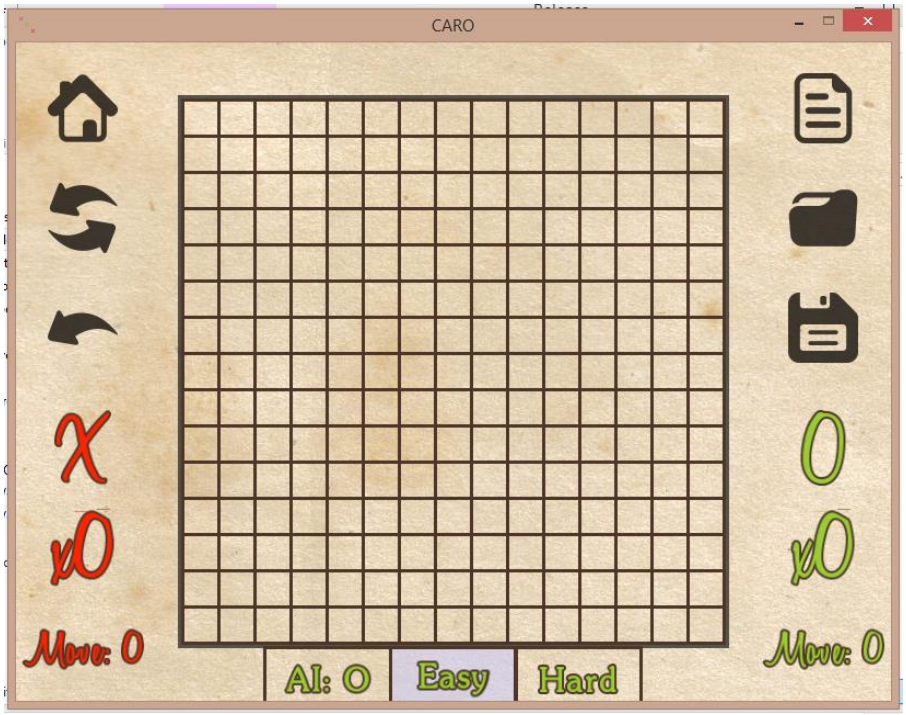
Sau khi click chuột chọn PLAY

Trò chơi được build trong 2 chế độ: 2 người chơi (2 PLAYER) và đánh với máy (1 PLAYER).

Màn hình game chế độ 2 người chơi



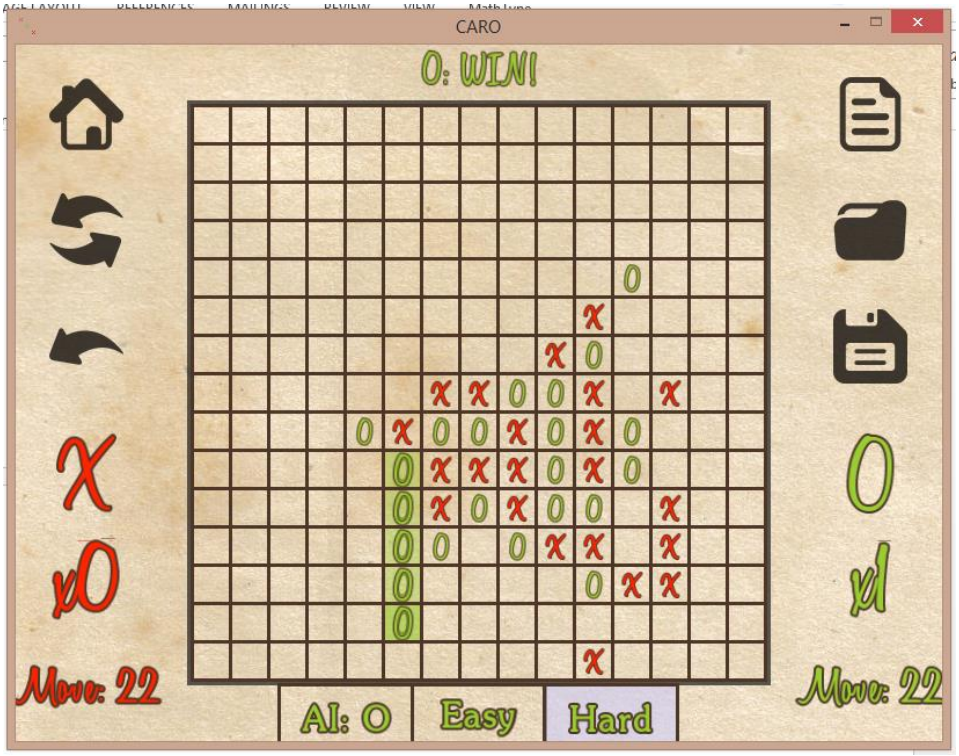
Chế độ đánh với máy



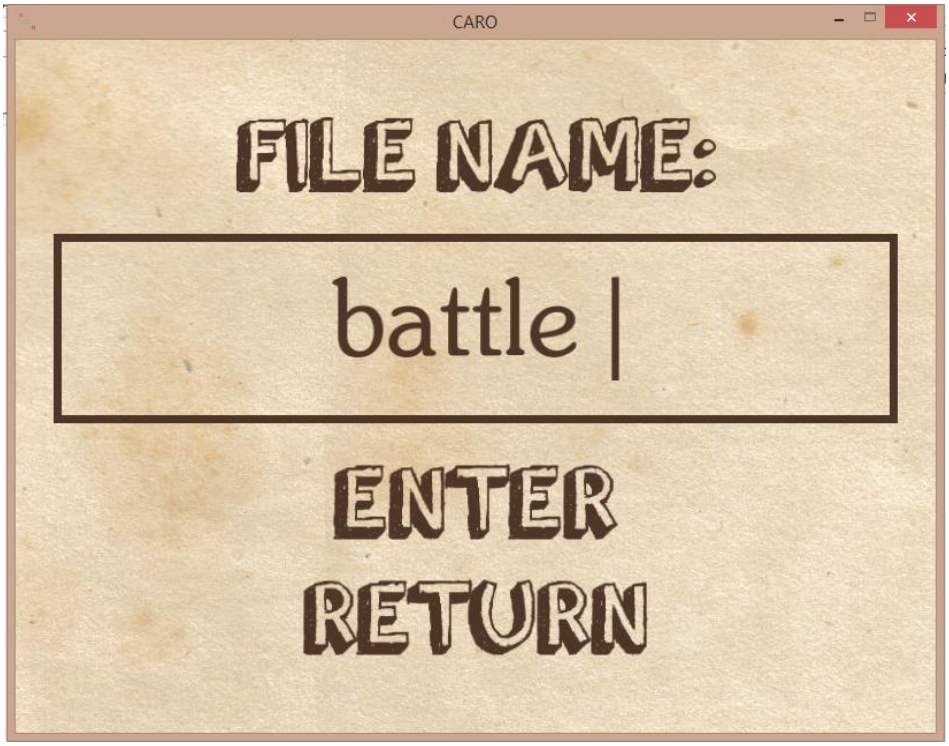
Người chơi sẽ được lựa chọn giữa 2 chế độ máy: Easy và Hard

Đồng thời có thể thay đổi lượt đánh để máy đánh trước hoặc mình đánh trước.

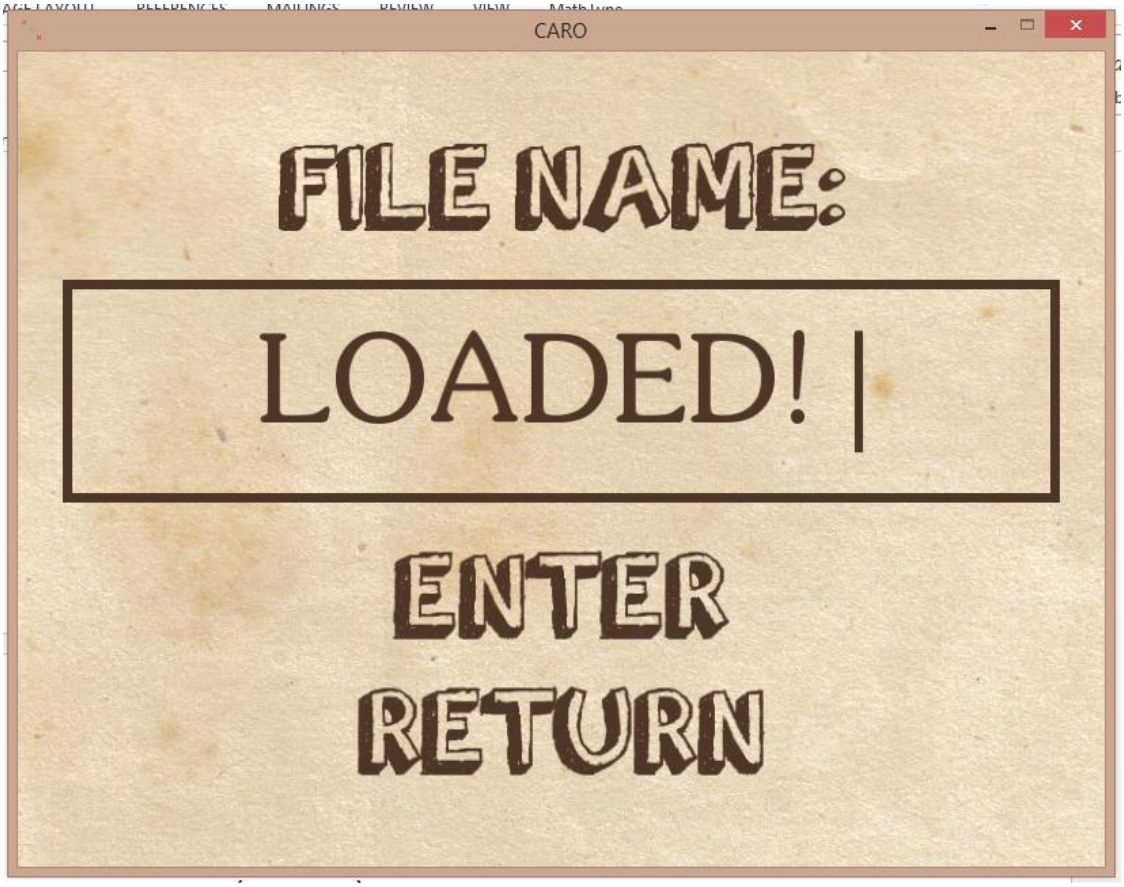
Khi O thắng



Khi nhấn vào nút save game



Khi load game thành công



File sẽ được lưu tại data\save\1 player (khi đánh với máy) hoặc trong data\save\2 player (khi 2 người chơi).

Trận đấu khi được save lại và load lên chứa hoàn toàn đầy đủ thông tin như 1 trận đấu bình thường như: số lượt đánh 2 bên, thứ tự các quân đã đánh (tức người chơi có thể sử dụng chức năng undo kể cả với bàn cờ được load),...

4. Tổ chức mã nguồn

4.1 Class

Class	Chức năng
Common	Lớp chung chứa các hàm hay dùng
Game	Quản lý, vận hành trò chơi
State	Lớp cha để thực hiện đa hình trong state machine
File	Kế thừa State, điều hướng khi save, load
Menu	Kế thừa State, điều hướng ở menu chính
Play	Kế thừa State, điều hướng ở trong 1 trận game
Audio	Quản lý âm thanh
Board	Quản lý bàn cờ
Bot	Gồm các hàm, thuật toán đánh với máy
Button	Quản lý nút bấm
Display	Thực hiện các hàm đồ họa
TextBox	Thực hiện các chức năng in chữ ra màn hình

4.2 Về chế độ build state machine

Trong game, khi tương tác với người dùng, chương trình phải thực hiện các chức năng khác nhau liên tục, trong khi các hàm đều hầu như là riêng lẻ nhau thì chúng nhiều lúc lại dính líu một cách phức tạp thông qua các biến truyền vào nên việc build 1 chương trình chạy xuyên suốt từ hàm này qua hàm khác là một việc khá khó khăn (đặc biệt với các chương trình lớn). Để đảm bảo việc chỉnh sửa mã nguồn trở nên dễ dàng hơn, chương trình dễ hiểu hơn thì nhóm em đã sử dụng một phương pháp bố trí chương trình tên là State Machine.

Cụ thể ở Caro, chương trình sẽ chia làm 3 State (trạng thái) là: File, Menu và Play. Mỗi State đều có 2 việc phải làm là “chạy xử lý” (Run()) và hiển thị ra

màn hình (Display()). Trong mỗi State lại có những State khác nữa. Như vậy việc chạy chương trình chính là việc các State thay đổi liên tục, với State tương ứng ta có xử lý tương ứng. Khi mã nguồn được chia rõ ràng thì việc chỉnh sửa tại 1 nơi sẽ ít ảnh hưởng tới nơi khác hơn nhờ đó ít gây ra lỗi.

4.3 Các hàm chính

Class Audio:

Quản lí âm thanh

Hàm:

void click() phát tiếng click

void music() phát nhạc

Class Board:

Quản lí caro

Hàm:

void reset() đưa các giá trị của board về mặc định ngoại trừ score

bool load(const string&) load dữ liệu của board

bool save(const string&) save dữ liệu của board

bool getCur(int, int) lấy vị trí vừa đánh

void undo() undo nước vừa đánh

bool check() check cờ

void set() đánh cờ

bool inRange(int, int) kiểm tra vị trí hợp lệ không

bool win() kiểm tra thắng thua

bool test(int, int) kiểm tra thắng thua ở hàng hay cột gì đó

bool count(int, int) đếm số quân cùng loại và kiểm tra có bị chặn

Class Bot:

Quản lí bot

Hàm:

void move() đánh cờ

bool load() load bot

bool save() save bot

void easy() đánh random

bool TestAround(int, int) kiểm tra xung quanh có địch

void hard() đánh có chặn và tấn công

int DiemTC…(int, int) tính điểm tấn công

int DiemPN…(int, int) tính điểm phòng ngự

Class Button: kế thừa sf::Drawable

Quản lí mấy cái nút bấm

Hàm:

void setTexture(const sf::Texture&) nạp texture cho cái nút

void setColor(const sf::Color&) bôi màu lên texture

void setPos(int, int) đặt vị trí của nút

void scale(float) phóng to/nhỏ

void unScale() trả về mặc định

virtual void draw(sf::RenderTarget&, sf::RenderStates) const vẽ kế thừa

Class Display:

Quản lý vẽ

Hàm:

void Intro(int) vẽ màn hình menu mới vô

void Menu(int) vẽ màn hình menu sau đó

void Info(int) vẽ màn hình info

void Play(int, const board&, const Bot&) vẽ màn hình chơi game

void File(int, const string&) vẽ màn hình save/load

void Caro(const Board&) vẽ bàn caro

void Title() vẽ cái title

Class TextBox:

Quản lí textbox để in ra màn hình

Hàm:

void setFont(const sf::Font&) nạp font cho text

void setSize(int) đặt size text

void setColor(const sf::Color&) đặt màu cho text

void setSizeOL(int) đặt size cho viền text

void setColorOL(const sf::Color&) đặt màu cho viền text

void setStyle(sf::Text::Style) đặt kiểu text

void setText(const sf::String&) nạp chữ cho text

void boxColor(const sf::Color&) đặt màu cho box

void boxSize(int, int) đặt size cho box

void boxColorOL(const sf::Color&) đặt màu cho viền box

void boxSizeOL(int, int) đặt size cho viền box

void setRotate(float) xoay text

void setPos(int, int) đặt vị trí cho text

void scale(float) phóng to/nhỏ cho text

void unScale() trả về mặc định

virtual void draw(sf::RenderTarget&, sf::RenderStates) const vẽ kế thừa

class Common

static sf::Vector2i index(sf::RenderWindow&, sf::Rect<int>, int, int) trả về vị trí con chuột

class Game

Quản lí game

Hàm:

void run() chạy game

void change() đổi màn hoặc thoát

class State

Class thuần ảo

Hàm:

virtual void run() lấy input

virtual void display() vẽ màn hình

Class File

Quản lí màn hình save/load kế thừa State

Hàm:

int update() cập nhập vị trí con chuột

void change() xử lí input

Class Menu:

Quản lí màn hình menu kế thừa State

Hàm:

int update() cập nhập vị trí con chuột

void change() xử lí input

Class Play

Quản lí màn hình save/load kế thừa State

Hàm:

sf::Vector2i update() cập nhập vị trí con chuột

void change() xử lí input

5. Data dùng thêm

5.1 Về thuật toán đánh với máy

Nguyên tắc chọn ô: Tính lần lượt điểm tấn công và điểm phòng ngự của từng ô trống trên bàn cờ. Ô được đánh là ô có tổng điểm tấn công và phòng ngự cao nhất.

Class của AI: Computer

Khai báo: bao gồm 15 hàm tĩnh và 2 mảng điểm tĩnh

```
static int MangDiemTanCong[7] = { 0,9,90,1000,6001,30000,100000 };
static int MangDiemPhongNgu[7] = { 0,3,30,400,2050,13000,50000 };
```

Có 4 loại ô trống trên bàn cờ (lưu ý rằng có thể có một hoặc một số ô trống cùng loại và nằm kế với ô trống đang xét)

- Loại 1: Ô trống nằm giữa quân địch (quân của người chơi) và quân ta (quân của máy)
- Loại 2: Ô trống nằm giữa 2 quân ta.
- Loại 3: Ô trống nằm giữa 2 quân địch.
- Loại 4: Ô trống không thuộc 3 loại trên.

Nguyên tắc tính điểm một ô: Xét xem ô đó thuộc loại nào, đồng thời tính số quân địch và số quân ta trong phạm vi 7 ô gần nhất mà không ra khỏi bàn cờ theo hàng ngang, hàng dọc và 2 đường chéo. Sau đó dựa vào 2 mảng điểm để tính điểm tấn công và phòng ngự của ô đó.

Nếu có 2 ô bằng điểm, xét xem ô nào có liên kết với nhiều quân ta hơn thì sẽ ưu tiên đánh vào ô đó.

5.2 Data dùng thêm

Trong chương trình nhóm em có sử dụng các nguồn data như hình nền, font chữ (gồm tiếng việt và tiếng anh), các file âm thanh,... từ internet. Các file này được lưu trong folder data trong project. Vì vậy khi chạy file .exe thì nhất thiết phải để file này cùng vị trí lưu với folder data.