**Đại học Khoa Học Tự Nhiên – Đại học Quốc gia Thành phố Hồ Chí Minh**

**Khoa Công Nghệ Thông Tin**


# BÁO CÁO ĐỒ ÁN TRÒ CHƠI CARO

## Môn: Kỹ thuật lập trình

## Lớp: 22CTT4

## Giảng viên hướng dẫn: TS. Trương Toàn Thịnh

### Các thành viên nhóm 7:

Võ Đình Long – 22120195

Trần Đức Minh – 22120212

Mai Nhật Nam – 22120219

Phạm Tấn Nghĩa– 22120230

Hoàng Thanh Thảo Nguyên – 22120235

# LỜI CẢM ƠN

Lời đầu tiên, xin trân trọng cảm ơn thầy Trương Toàn Thịnh, thầy đã tận tình hướng dẫn chúng em trong quá trình học tập cũng như trong việc hoàn thành báo cáo này để hỗ trợ cho đồ án.

Xin chân thành cảm ơn các Thầy, Cô thuộc khoa Công nghệ Thông tin nói riêng và toàn bộ Thầy, Cô ở các khoa nói chung trường Đại Học Khoa học Tự nhiên, Đại học Quốc gia Thành phố Hồ Chí Minh đã tận tình giảng dạy cho chúng em trong suốt thời gian học tập.

Do giới hạn kiến thức và khả năng lý luận của bản thân còn nhiều thiếu sót và hạn chế, kính mong sự chỉ dẫn và đóng góp của các Thầy, Cô để bản báo cáo và sản phẩm đồ án của chúng em được hoàn thiện hơn. Xin chân thành cảm ơn!

Thành phố Hồ Chí Minh, ngày 30 tháng 05 năm 2023

Đại diện nhóm

Phạm Tấn Nghĩa

# MỤC LỤC

# A. GIỚI THIỆU

## I. Thông tin nhóm

| MSSV | Họ và Tên | Email |
|---|---|---|
| **22120195** | Võ Đình Long | 22120195@student.hcmus.edu.vn |
| **22120212** | Trần Đức Minh | 22120212@student.hcmus.edu.vn |
| **22120219** | Mai Nhật Nam | 22120219@student.hcmus.edu.vn |
| **22120230** | Phạm Tấn Nghĩa | 22120230@student.hcmus.edu.vn |
| **22120235** | Hoàng Thanh Thảo Nguyên | 22120235@student.hcmus.edu.vn |

## II. Kế hoạch

### 1. Các cột mốc sản phẩm

| Cột mốc | Công việc |
|---|---|
| 28/2/2023 – 20/3/2023 | - Tìm hiểu chung nội dung đồ án, liệt kê các vấn đề mà đồ án yêu cầu.<br>- Xây dựng kế hoạch trao đổi thông tin và làm việc online.<br>- Làm bài toán kiểm tra caro cơ bản và kiểm tra dữ liệu nhập vào đã có bên X hay O thắng hay không. |
| 21/3/2023-10/4/2023 | - Xử lý yêu cầu hiện thị người chơi đã đạt 5 ô liên tiếp<br>- Xử lý về yêu cầu tạo cổng, lưu và tải trò chơi.<br>- Xây dựng menu chính với các chức năng: New game, Load game, Setting, High score, Exit và giới thiệu nhóm. |
| 11/4/2023-20/4/2023 | - Thêm các hoạt ảnh, hiệu ứng và âm thanh trò chơi.<br>- Tạo thêm chức năng chơi game với BOT. |
| 20/4/2023-28/4/2023 | - Làm báo cáo đồ án -> Nộp báo cáo lần 02 trên Moodle<br>- Làm slide báo cáo về đồ án (giới thiệu chung, các chủ đề chính của game). |

| | |
|---|---|
| 29/4/2023 – 09/05/2023 | - Chỉnh sửa lần cuối cùng các nội dung trong báo cáo đồ án<br>- Đăng tải video demo game lên youtube, mô tả chi tiết về đồ án. |
| 10/05/2023 – 16/05/2023 | - Tập duyệt báo cáo nội dung đồ án |

## 2. Kế hoạch trao đổi thông tin (dự kiến đến hết ngày 16/05/2023)

| Thời gian | Nội dung dự kiến | Phương pháp |
|---|---|---|
| 19h00 28/02/2023 | - Triển khai các bước chuẩn bị đồ án.<br>- Phân công công việc. | Online qua Zoom |
| 22h00 18/03/2023 | - Đánh giá về khâu chuẩn bị ưu – nhược điểm và cách khắc phục.<br>- Triển khai kế hoạch từ ngày 19/3/2023 đến 31/03/2023. | Online qua zoom |
| 22h00 31/03/2023 | - Đánh giá về khâu chuẩn bị ưu – nhược điểm và cách khắc phục.<br>- Thêm các ý tưởng cần bổ sung cho đồ án<br>- Triển khai kế hoạch từ ngày 31/3/2023 đến 10/04/2023. | Online qua zoom |
| 12h30 05/04/2023 | - Họp bàn và sửa bug các nhóm hàm còn đang bị lỗi chưa chạy được chính xác | Offline tại Hội quán Khoa học Trường ĐHKHTN- CS Linh Trung |
| 22h00 24/04/2023 | - Họp kiểm tra tiến độ đồ án giai đoạn cuối<br>- Đánh giá về khâu chuẩn bị ưu – nhược điểm và cách khắc phục. | Online qua Google Meet |

| | | |
|---|---|---|
| 22h00 09/05/2023 | - Hoàn thành source code của đồ án<br>- Tiến hành làm powerpoint thuyết trình và hoàn thiện báo cáo lần 3(nộp hoàn thiện) | Online qua Google Meet |
| 22h00 14/05/2022 | - Tập dượt | |

## 3. Đánh giá hoạt động nhóm xuyên suốt đồ án

| MSSV | Họ và tên | Mức độ đóng góp |
|---|---|---|
| 22120195 | Võ Đình Long | 100 % |
| 22120212 | Trần Đức Minh | 100 % |
| 22120219 | Mai Nhật Nam | 100 % |
| 22120230 | Phạm Tấn Nghĩa | 100 % |
| 22120235 | Hoàng Thanh Thảo Nguyên | 100 % |

# B. NỘI DUNG

## I. Lời dẫn

Caro là tựa game quen thuộc được nhiều người ưa thích. Đồ án game Caro giúp nhóm học tập và áp dụng các kiến thức cơ bản của môn Kỹ Thuật Lập Trình để tạo nên tựa game đơn giản này. Đây là cơ hội để mỗi thành viên trong nhóm phát triển, nâng cao khả năng của bản thân qua các hoạt động nhóm, học hỏi các kiến thức để hoàn thiện sản phẩm.

## II. Giao diện

### 1. Màn hình Load game

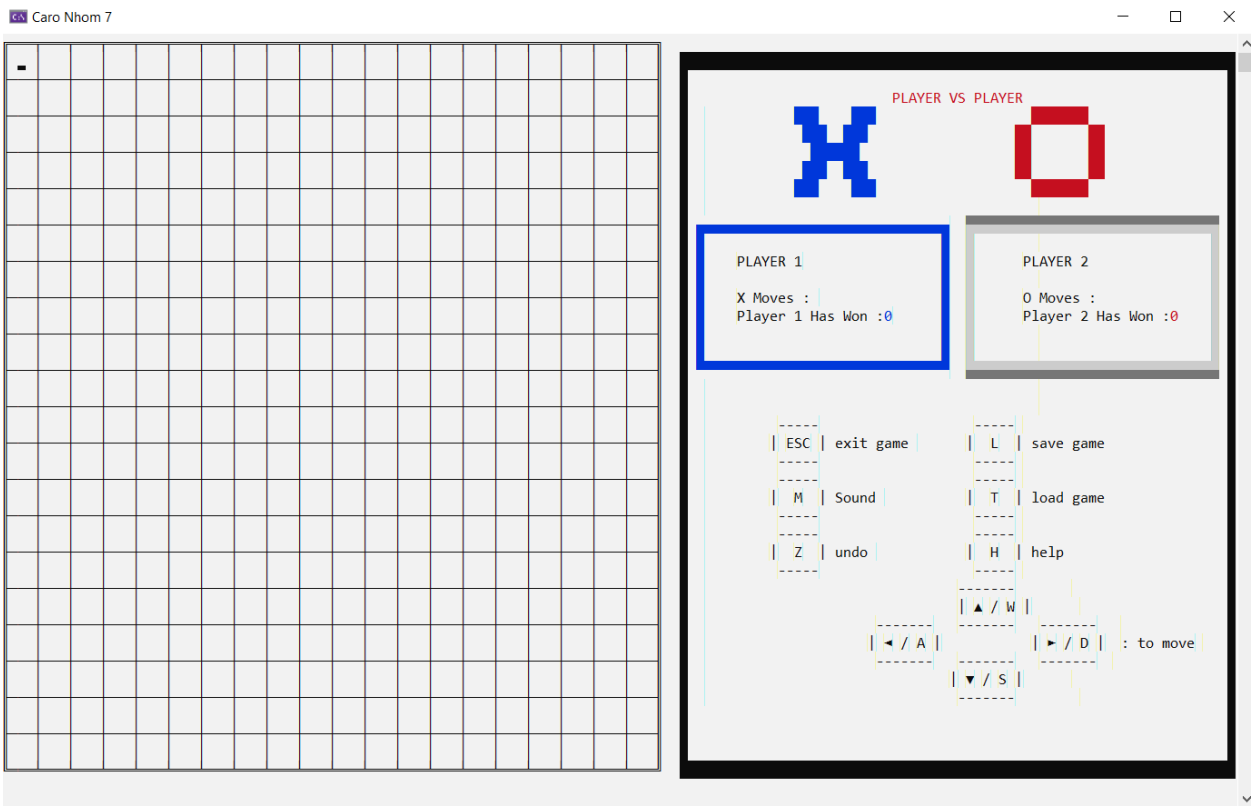

- Đây là màn hình chữ Caro chạy nhấp nháy di chuyển liên tục và các ô đỏ chạy qua lại để load game trước khi vào giao diện menu game.

### 2. Màn hình mở đầu game (menu game)

Dựa vào tựa game Caro, dưới đây là hình ảnh mở màn giao diện game của nhóm:


- Ở đây chúng ta có thể chọn các chế độ chơi như Người vs Người, Người vs Máy.
- Chức năng Load game cho phép ta nhập lại tên file đã lưu trữ dữ liệu game cũ để tiếp tục ván cờ caro chơi chưa hoàn thành
- History là màn hình cho phép người dùng xem lại lịch sử đã chơi
- Instruction là màn hình cho phép người dùng xem hướng dẫn cách chơi
- About là màn hình hiển thị thông tin giới thiệu nhóm
- Exit cho phép người dùng thoát khỏi màn hình console

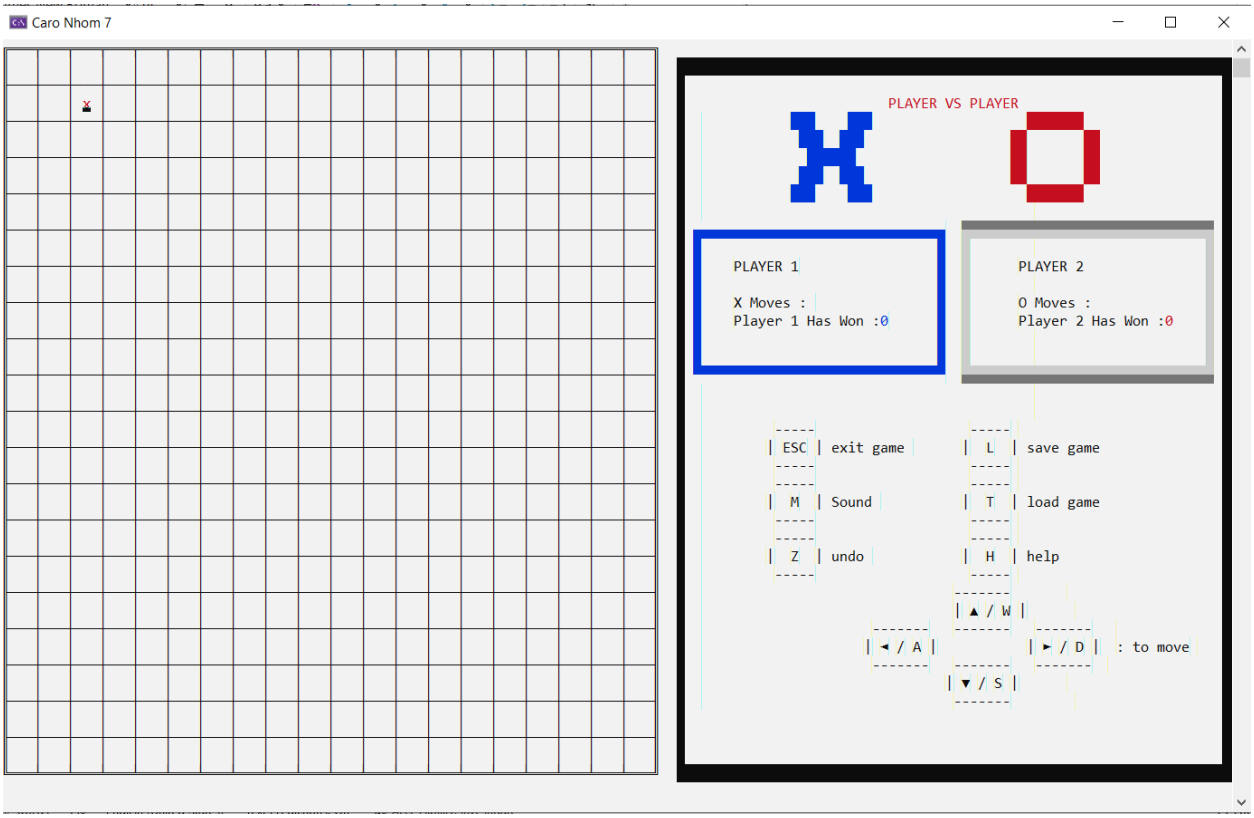
**3. Giao diện chính của game**



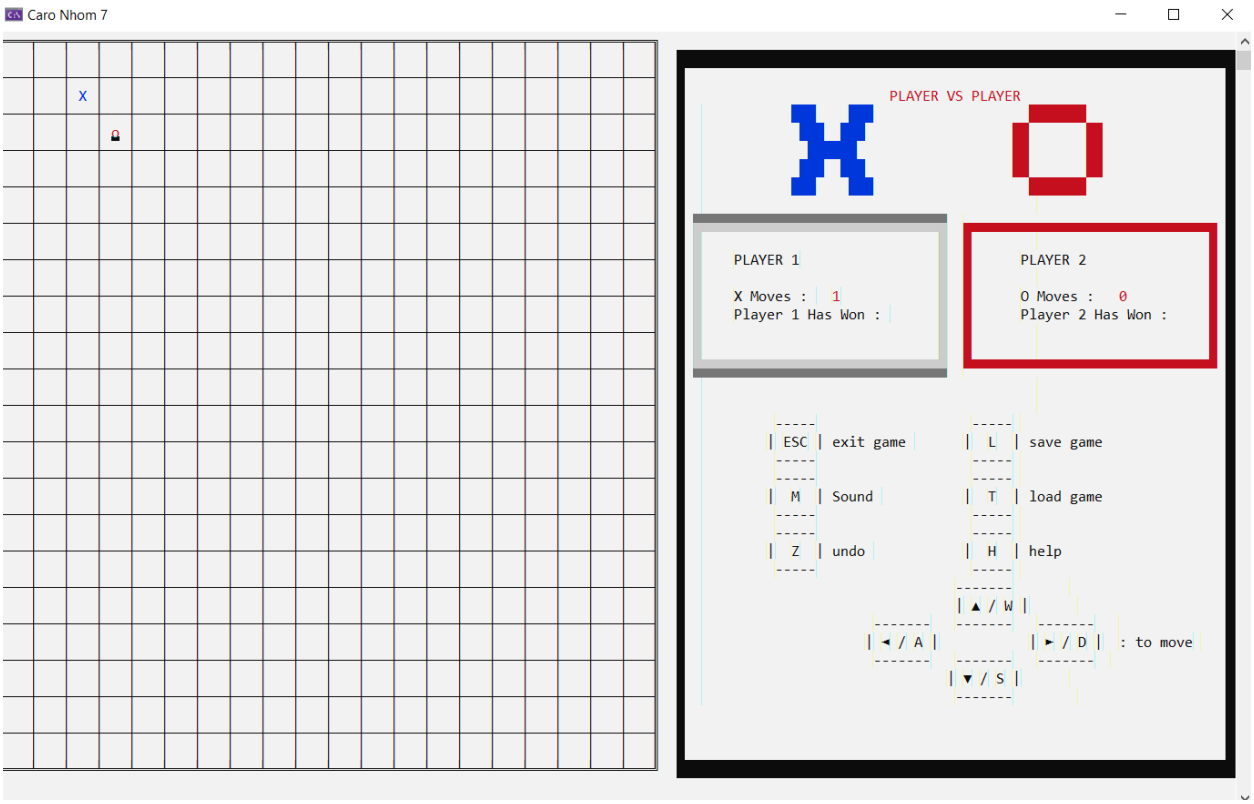

Đây chính là màn hình chính của game bao gồm 1 bàn cờ 20x20 ô.

Phía góc phải hiện thị các thông tin gồm:

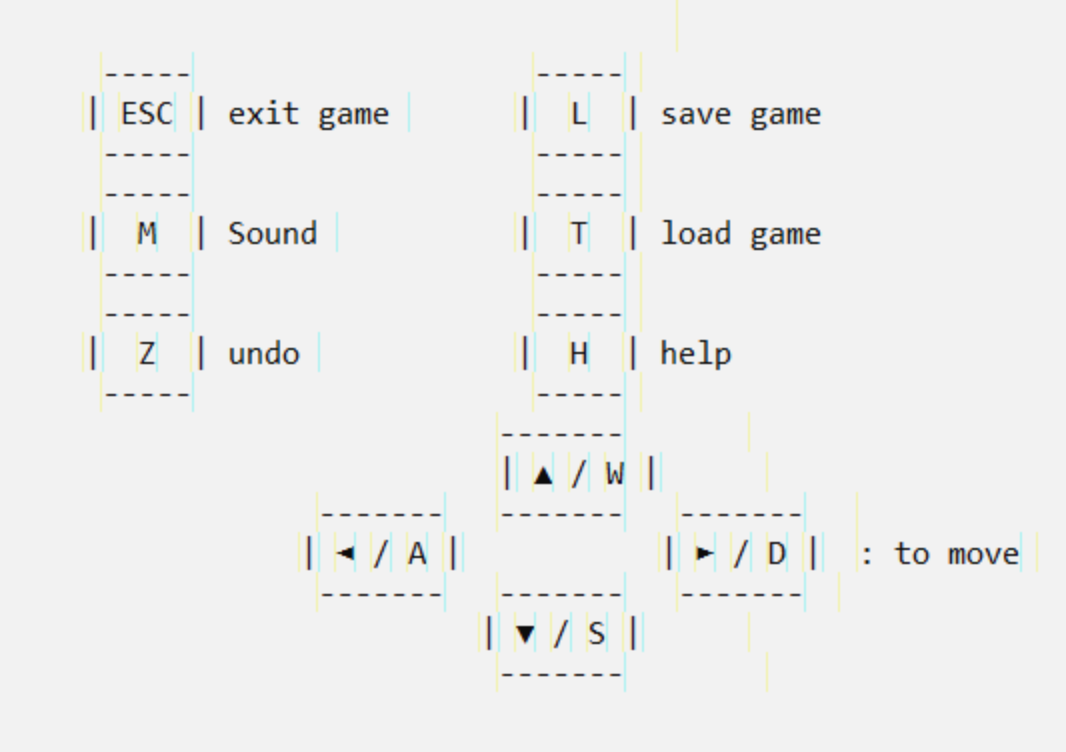
PLAYER 1 / 2: Số quân X/Y đã đánh và số ván đã thắng của từng bên



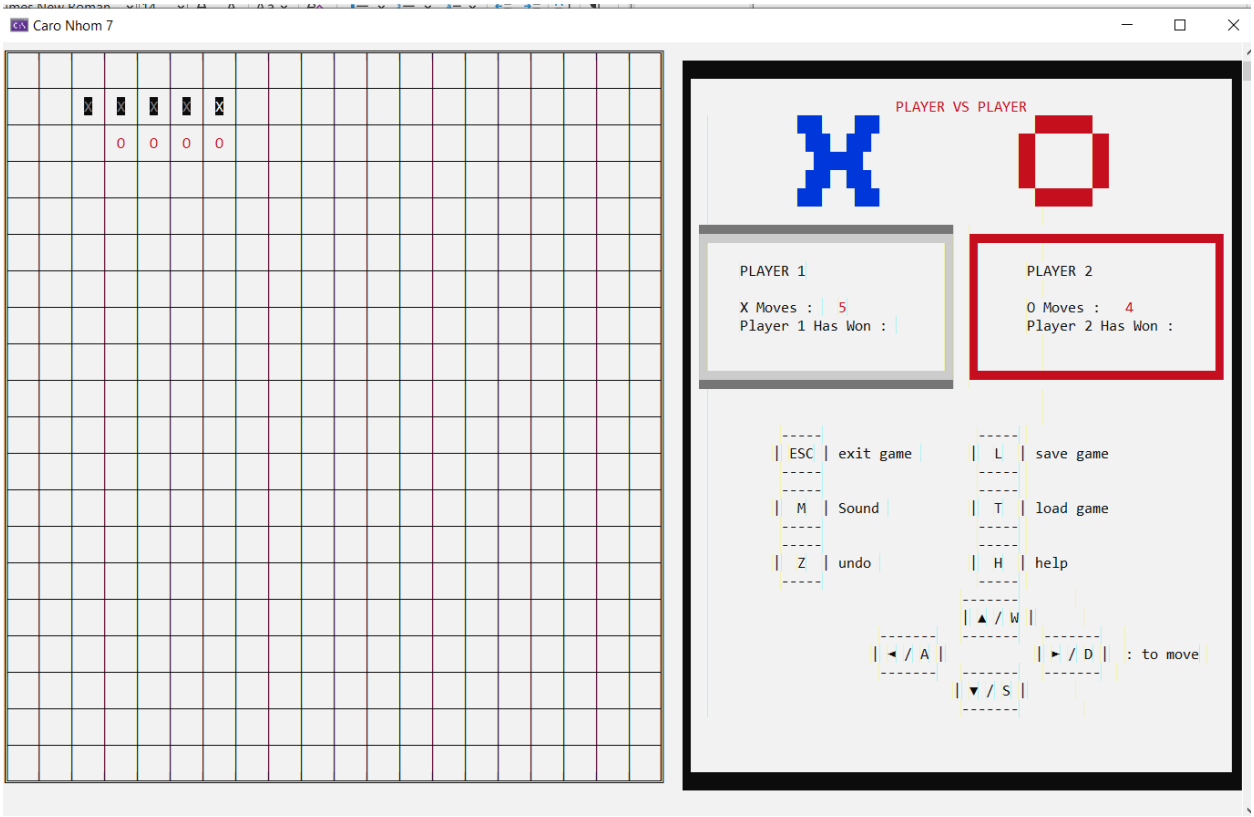

Khi quân x di chuyển thì 1 biểu tượng “x” sẽ di chuyển theo vị trí của người chơi sử dụng các phím “W”, “S”, “A”, “D” (tương tự vậy với quân “O”). Và khi quân X đánh thì phần giao diện thông tin của quân X được chuyển sang màu xanh. Đối với quân O thì phần giao diện thông tin của quân O sẽ chuyển sang màu đỏ.



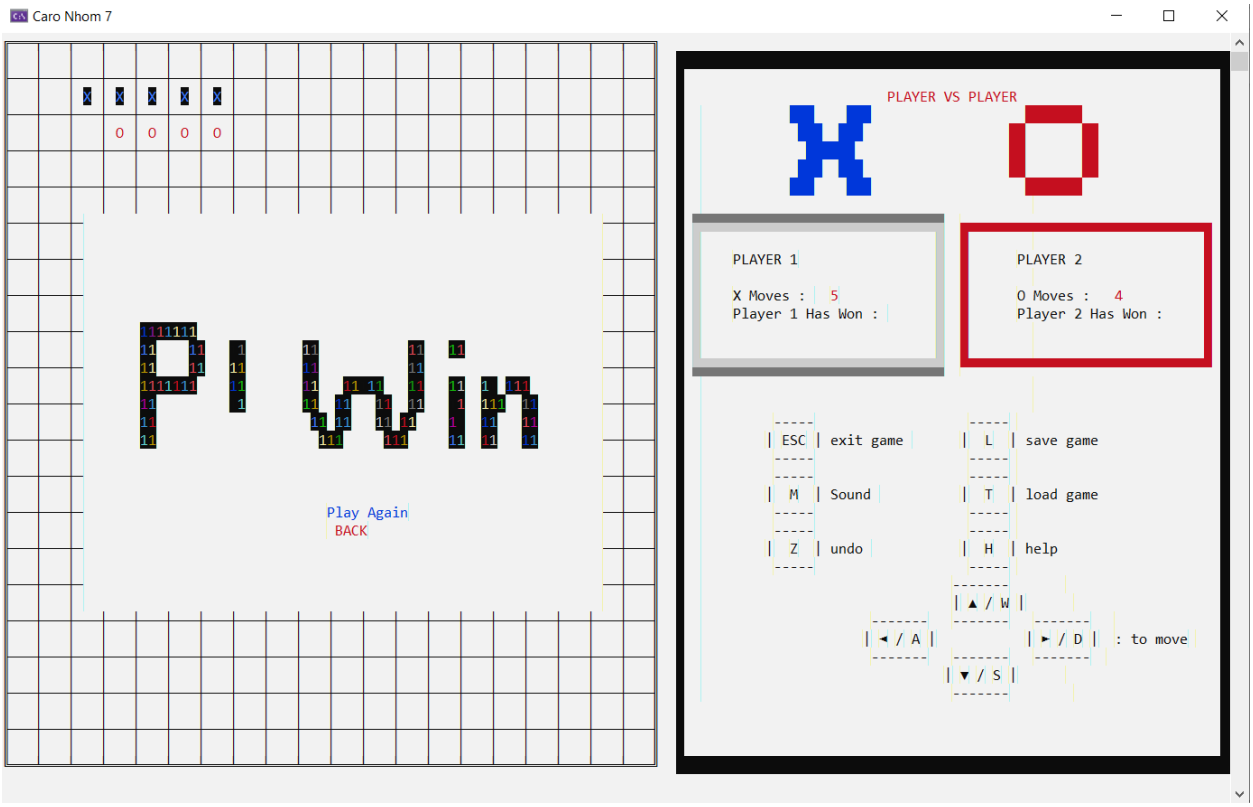

Ở phía dưới ta có thể thấy các biểu tượng “ESC”, “L”, “Y”, “T” , “M”, “H” , “W”, “S”, “A”, “D” tương ứng với các chức năng exit game, save game, sound, load game, undo, help, và các phím di chuyển lên xuống trái phải.



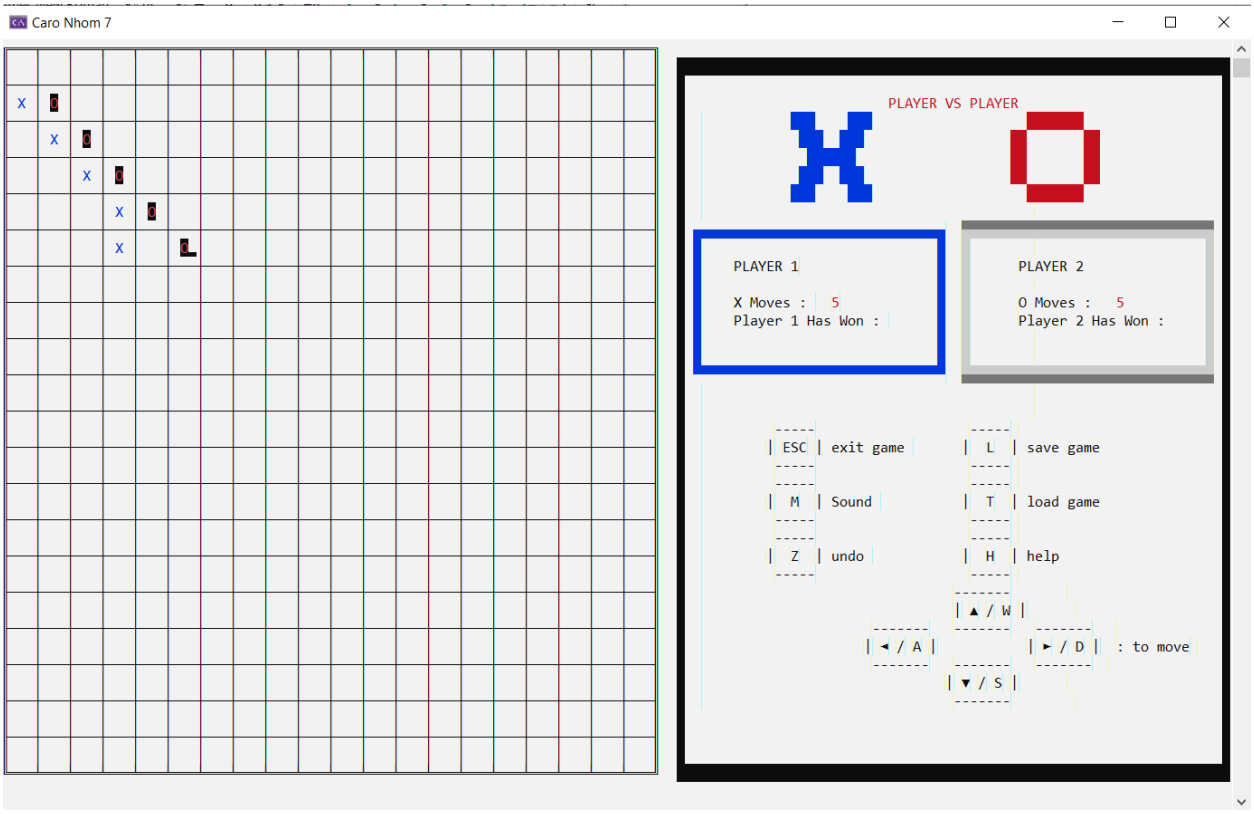

## 4. Màn hình chiến thắng/ hòa



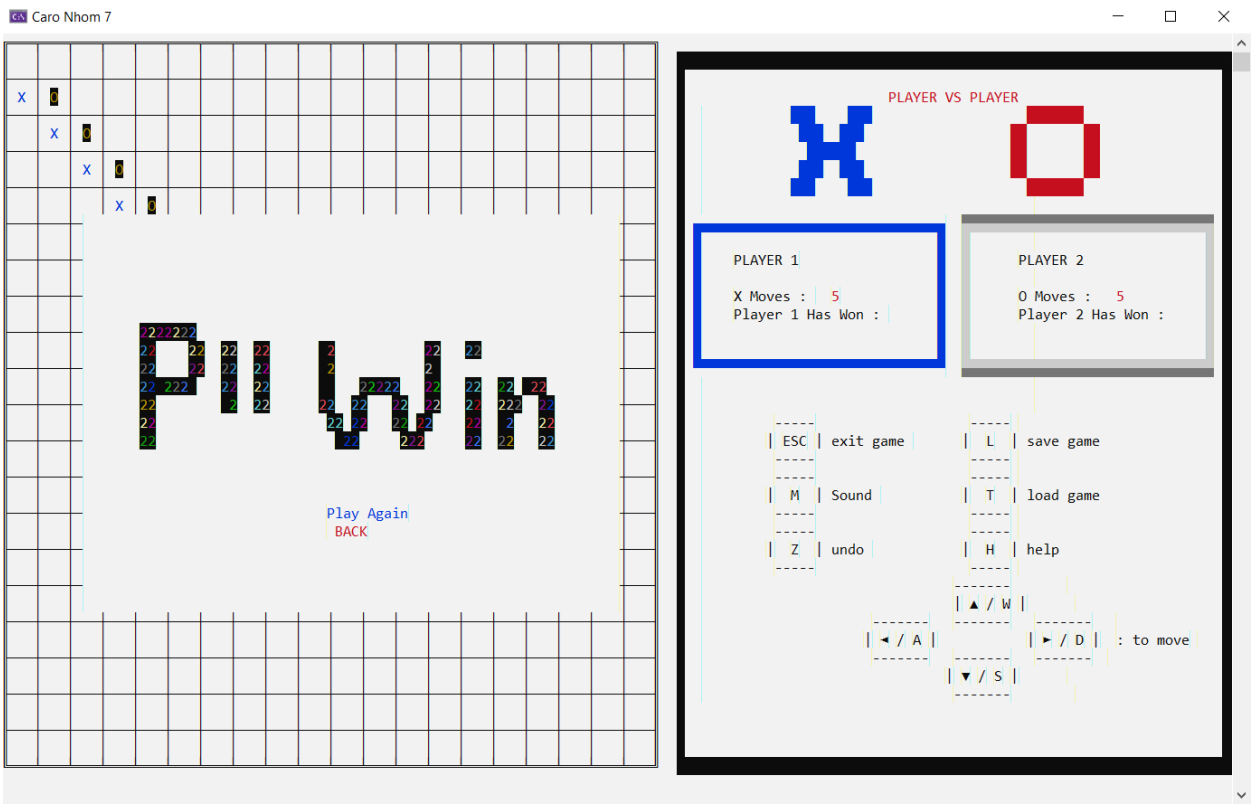
Đây là màn hình khi Player 1 chiến thắng và hiện hiệu ứng nhấp nháy ở 5 quân cờ của Player1 khi chiến thắng. Sau đó màn hình tiếp tục hiện dòng chữ ‘P1 WIN” như hình dưới đây:



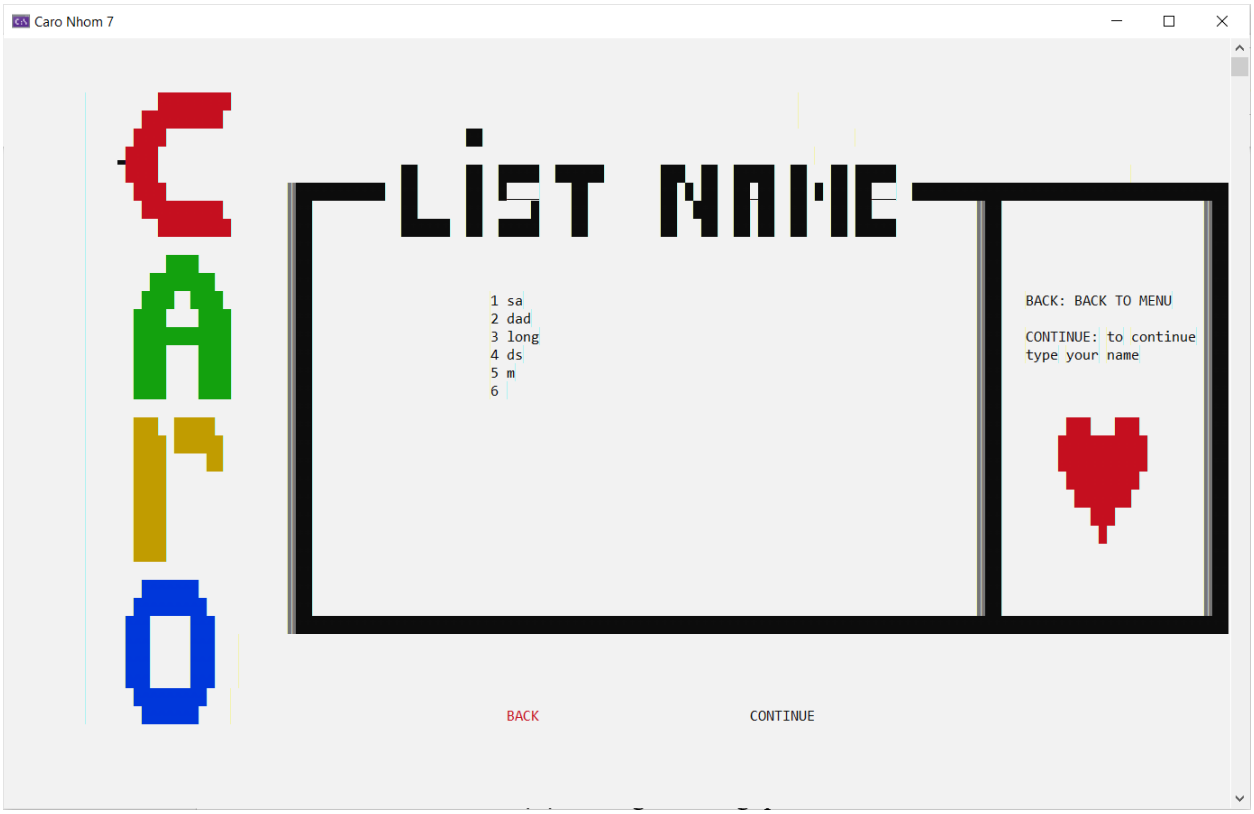

Đây là màn hình khi Player 2 thắng với hiệu ứng tương tự như Player 1:




Màn hình hiển thị Player 2 chiến thắng




## 5. Màn hình Load game




Ở màn hình Load game, LIST FILE NAME để hiển thị các tập tin lưu trữ danh sách các ván game trước được lưu. Để truy cập vào lại các ván game đó, cần di chuyển sang CONTINUE sử dụng các phím W, S,A,D để người dùng có thể nhập đúng File ván chơi cần truy cập để tiếp tục chơi tiếp sau khi save game. Lưu ý ta chỉ có thể lưu tối đa 10 ván chơi.

### 6. Màn hình History


Màn hình hiển thị lịch sử người thắng qua các ván

## 7. Màn hình Instructions/ Hướng dẫn


Đây là màn hình hướng dẫn người chơi gồm các nút di chuyển và luật chơi.

## 8. Màn hình About/ Giới thiệu


Đây là màn hình hiển thị thông tin giới thiệu các thành viên trong nhóm

## III. Mô tả sơ bộ các hàm và chức năng của từng hàm trong mã nguồn đồ án

### 1. Tổ chức file

- File Caro.cpp dùng để viết hàm main tổng thể để chạy chương trình

- File Board.h dùng để khai báo các hàm ở file Board.cpp để vẽ bàn cờ và các thông tin khác trên màn hình chơi

- File Menu.h dùng để khai báo các hàm ở file Menu.cpp để hiển thị các thông tin liên quan đến trò chơi.

- File Game.h dùng để khai báo các hàm ở file Game.cpp để xử lí dữ liệu của trò chơi.

### 2. Nhóm hàm về đồ họa ở file Board.h (Nhóm hàm để vẽ bảng)

Sau đây là trích đoạn code sử dụng để khai báo hàm và chức năng của từng hàm:

```
#include "Menu.h"
void Draw(int i, int x, int y); // Hàm in bảng điểm
void LoadingGame();// Hàm vẽ giao diện LOADING đầu Game
void winDraw(int i, char variableWin, int x, int y); //Vẽ đồ họa Win
void drawBoard(); // Vẽ bảng caro
```

```
void PrintScoreBoard(); //Vẽ bảng điểm
void box(); //Hiển thị nút esc sau khi người dùng thắng
```

* Tất cả các hàm vẽ chữ, vẽ hiệu ứng đều sử dụng phương pháp đọc các kí tự đã được định sẵn bằng file txt bằng kỹ thuật đọc/ghi File.

***2.1 Miêu tả sơ bộ về các nhóm hàm bằng mã giả (pseudo code)***

**2.1.1 Hàm Draw. (Hàm in bảng điểm)**

```
procedure Draw(i, x, y):
    file = open file_name[i] with mode "read"
    while not end_of_file(file):
        line = read_line(file)
        gotoXY(x, y)
        for j = 0 to length(line) - 1:
            if line[j] equals '.':
                Textcolor(15)
                output char(219)
            else if line[j] equals '+':
                Textcolor(FullRed)
                output line[j]
            else if line[j] equals '~':
                Textcolor(FullGreen)
                output line[j]
            else if line[j] equals '@':
                Textcolor(FullYellow)
                output line[j]
            else if line[j] equals '^':
                Textcolor(FullBlue)
                output line[j]
            else if line[j] equals '*':
                Textcolor(0)
                output line[j]
            else if line[j] equals '8':
                Textcolor(FullAzure)
                output line[j]
            else if line[j] equals '/':
                Textcolor(FullAzure)
                output char(254)
            else if line[j] equals '"':
                Textcolor(Red)
                output char(003)
            else if line[j] equals 'z':
                Textcolor(Black)
                output char(179)
            else if line[j] equals '#':
                Textcolor(Grey)
                output char(179)
```

```
            else if line[j] equals '|' or line[j] equals '-':
                Textcolor(0)
                output char(179)
            else:
                Textcolor(Black)
                output line[j]
        y = y + 1
    close file
end procedure
```

### 2.1.2 Hàm Loading Game

```
procedure LoadingGame() :
    playSound(6)
    hideCur()
    for l = 0 to 7 :
        x = 30
        y = 10
        file = open "Loading.txt" with mode "read"
        while not end_of_file(file) :
            line = read_line(file)
            gotoXY(x, y)
            if line equals "load" :
                y = 9
                Sleep(150)
            else:
for j = 0 to length(line) - 1 :
    if line[j] equals '.' :
        Textcolor(15)
        output char(219)
    else if line[j] equals 'x' :
        Textcolor(Blue)
        output char(254)
    else if line[j] equals '=':
Textcolor(FullRed)
output char(219)
    else if line[j] equals '0':
Textcolor(Grey)
output char(219)
    else if line[j] equals '+':
Textcolor(rand() modulo 100)
output line[j]
    else if line[j] equals '~':
Textcolor(rand() modulo 100)
output line[j]
    else if line[j] equals '@':
Textcolor(rand() modulo 100)
output line[j]
    else if line[j] equals '^':
Textcolor(rand() modulo 100)
output line[j]
y = y + 1
close file
clearConsole()
end procedure
```

Hàm LoadingGame có chức năng tạo hiệu ứng với chữ "Caro" nhấp nháy trong lúc mở đầu game bằng cách đọc tập tin "Loading.txt" và cho chạy hiệu ứng màu sắc với các dòng chữ và kí tự tương ứng.

### 2.1.3 Hàm WinDraw

```
procedure winDraw(i, variableWin, x, y) :
    hideCur()
    if i equals 1 or i equals 2 :
        playSound(4)
    else :
        playSound(5)
        for k = 0 to 4 :
            file = open file_name[i] with mode "read"
            while not end_of_file(file) :
                line = read_line(file)
                gotoXY(x, y)
                if line equals "load" :
                    y = 10
                    Sleep(250)
                else:
for j = 0 to length(line) - 1 :
    if line[j] equals '.' :
        Textcolor(15)
        output char(219)
    else if line[j] equals variableWin :
Textcolor(rand() modulo 15)
output line[j]
    else if line[j] equals '/':
Textcolor(Red)
output char(003)
    else if line[j] equals ',':
Textcolor(Blue)
output char(004)
    else if line[j] equals '-':
Textcolor(Green)
output char(005)
y = y + 1
close file
end procedure
```

Hàm này được dùng để vẽ hiệu ứng khi thắng, thua hoặc hòa của ván chơi.

### 2.1.4 Hàm DrawBoard

```
procedure drawBoard() :
    system("color FA")
    Textcolor(Black)

    // Vẽ cột phải
    for i = 0 to SIZE * 2 - 1 :
        gotoXY(SIZE * 4, i + 1)
        if (i + 1) modulo 2 equals 1 :
            output char(186)
        else :
            output char(182)

            // Vẽ dòng ngang
            for i = 0 to SIZE * 2 - 1 step 2 :
```

```
                gotoXY(0, i)
                for j = 0 to SIZE * 4 - 1 :
                    if j modulo 4 equals 0 :
                        output char(197)
                    else :
                        output char(196)

                        // Vẽ cột trái
                        for i = 0 to SIZE * 2 - 1 :
                            gotoXY(0, i + 1)
                            if (i + 1) modulo 2 equals 1 :
                                output char(186)
                            else :
                                output char(199)

                                // Vẽ cột dọc
                                for i = 1 to SIZE * 2 - 1 step 2 :
                                    for j = 0 to SIZE * 4 - 1 step 4 :
                                        gotoXY(j, i)
                                        if j equals 0 :
                                            continue
                                            output char(179)

                                            // Vẽ dòng đầu
                                            gotoXY(0, 0)
                                            for i = 0 to SIZE * 4 - 1:
if i modulo 4 equals 0 :
    output char(209)
else :
    output char(205)
    gotoXY(0, 0)
    output char(201)
    gotoXY(SIZE * 4, 0)
    output char(187)

    // Vẽ dòng cuối
    gotoXY(0, SIZE * 2)
    for i = 0 to SIZE * 4 - 1:
if i modulo 4 equals 0 :
    output char(207)
else :
    output char(205)
    gotoXY(0, SIZE * 2)
    output char(200)
    gotoXY(SIZE * 4, SIZE * 2)
    output char(188)

    // Về vị trí (2, 1)
    gotoXY(2, 1)
    end procedure
```

Hàm này có chức năng là vẽ bàn cờ caro với kích thước 20x20 sử dụng hàm tọa độ gotoXy(int x, int y) và các kí tự trong bảng mã ASCII.

### 2.1.5 Hàm PrintScoreBoard

```
PrintScoreBoard(int k)
file = mở tệp "1.txt" để đọc
```

```
nếu k == 1 thì
trong khi chưa kết thúc tệp
line = đọc một dòng từ tệp
y = 4

cho mỗi ký tự j trong line
nếu line[j] == '.'
in hình vuông đã được điền với màu chữ là trắng
nếu line[j] == '1'
in hình vuông đã được điền với màu chữ FullBlue
nếu line[j] == '2'
in hình vuông đã được điền với màu chữ đen
nếu line[j] == '4'
in ký tự với màu chữ đen là ký hiệu hộp
nếu line[j] == '5'
in ký tự với màu chữ đen là ký hiệu
nếu line[j] == '6'
in ký tự với màu chữ đen là ký hiệu
nếu line[j] == '7'
in ký tự với màu chữ đen là ký hiệu
ngược lại
in ký tự với màu chữ đen

tăng y lên 1

đóng tệp

in các yếu tố đồ họa cho bảng điểm 1

nếu k == 2 thì
trong khi chưa kết thúc tệp
line = đọc một dòng từ tệp
y = 4

cho mỗi ký tự j trong line
nếu line[j] == '.'
in hình vuông đã được điền với màu chữ là trắng
nếu line[j] == '1'
in hình vuông đã được điền với màu chữ đen
nếu line[j] == '2'
in hình vuông đã được điền với màu chữ FullRed
ngược lại
in ký tự với màu chữ đen

tăng y lên 1

đóng tệp

in các yếu tố đồ họa cho bảng điểm 2

in văn bản và yếu tố đồ họa cho người chơi 1 và người chơi 2

in các yếu tố đồ họa cho bảng điểm 3
```

Hàm này có chức năng hiển thị các thông tin ở phía nửa phải màn hình bao gồm các thông tin đồ họa của X,O và các thông tin và hướng dẫn các phím chức năng của trò chơi.

### 2.1.6 Hàm Box

```
void box()
{
    int i = 5, j = 5;
    Textcolor(Red);
    gotoXY(35, j + 19);
    cout << "ESC : BACK";
}
```

## 3. Nhóm hàm về đồ họa ở file Menu.h

Sau đây là trích đoạn code sử dụng để khai báo hàm và chức năng của từng hàm:

```
#include <iostream>
#include <Windows.h>
#include <conio.h>
#include <mmsystem.h>
#include <time.h>
#include <vector>
#include <fstream>
#include <string>
#include <iomanip>
#include <time.h>
#define SIZE  20  // SIZE BÀN CỜ
//Định nghĩa thông số màu hiển thị
#define Black                   240
#define Blue            241
#define Green                   242
#define Red               244
#define Violet           245
#define FullBlue           17
#define BlueYellow         30
#define FullGreen          34
#define FullAzure        51
#define FullRed          68
#define YellowRed        100
#define FullYellow         102
#define Grey            135
#define GreyBlue        121
#define MintPink           189
#define defaultColor    7
using namespace std;
using std::cout;
#pragma comment(lib, "winmm.lib")
// Dinh nghia ten ham
static string menuItems[] = { "Player Vs Player", "Player Vs Computer", "Load Game",
"History", "Instruction", "About", "Exit" };
static int numItems = sizeof(menuItems) / sizeof(menuItems[0]);
static int currentSelection = 0;

//Dinh nghia file ve
static string file_name[] = { "Caro.txt", "p1win.txt", "p2win.txt", "Draw.txt",
"About.txt","Loading.txt" };
extern bool isSoundOn;
void showCur(); // Hàm hiển thị con trỏ
```

```
void hideCur(); // Hàm ẩn con trỏ
void SetConsole(int width, int height); // Hàm thiết lập chiều dài và rộng của màn
hình Console
void Textcolor(int color); // Hàm set màu chữ
void gotoXY(int x, int y); // Hàm di chuyển tọa độ
void playSound(int); // Hàm xử lí âm thanh
int getConsoleInput(); // Hàm xử lí chức năng khi nhập từ các phím W,S,A,D, L,T
void menu(); // Hàm hiển thị Menu
void clearConsole(); // Hàm xóa/chuyển màn hình console
void setConsoleTitle(); // Hàm đặt tên cho màn hình console

int readMode(char a[30]); //Tìm kiếm file người dùng (hỗ trợ hàm LoadLoad)
void readNameFile(); //Hiển thị các người dùng đã lưu
void Load(); // Mở file người chơi đã lưu
void instruction(); //Mục hướng dẫn
void history(); //Mục history
void About();  // thông tin
```

## *3.1 Miêu tả sơ bộ về các nhóm hàm*

### 3.1.1 Hàm showCur

```
void showCur()
{
    CONSOLE_CURSOR_INFO Info;
    Info.bVisible = TRUE;
    Info.dwSize = 20;
    SetConsoleCursorInfo(GetStdHandle(STD_OUTPUT_HANDLE), &Info);
}
```

### 3.1.2 Hàm hideCur

```
void hideCur()
{
    CONSOLE_CURSOR_INFO Info;
    Info.bVisible = FALSE;
    Info.dwSize = 20;
    SetConsoleCursorInfo(GetStdHandle(STD_OUTPUT_HANDLE), &Info);
}
```

### 3.1.3 Hàm SetConsoleTitle

```
void setConsoleTitle() {
    //system("color FA");
    string narrow_str = "Caro Nhom 7";
    int length = MultiByteToWideChar(CP_UTF8, 0, narrow_str.c_str(), -1, NULL, 0);
    std::wstring wide_str(length, L'\0');
    MultiByteToWideChar(CP_UTF8, 0, narrow_str.c_str(), -1, &wide_str[0], length);
    SetConsoleTitle(wide_str.c_str());
}
```

### 3.1.4 Hàm SetConsole

```
void SetConsole(int width, int height) {
    HWND console = GetConsoleWindow();
    RECT r;
    GetWindowRect(console, &r);
```

```
    MoveWindow(console, r.left, r.top, width, height, TRUE);
}
```

### 3.1.5 Hàm TextColor

```
void Textcolor(int color)
{
    HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE);
    SetConsoleTextAttribute(hConsole, color);
}
```

### 3.1.6 Hàm gotoXY

```
void gotoXY(int x, int y)
{
    COORD coord;
    coord.X = x;
    coord.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coord);
}
```

### 3.1.7 Hàm playSound

```
void playSound(int i) {
    static vector<const wchar_t*> soundFile{ L"beep.wav", L"move.wav",
       L"tick.wav", L"error.wav", L"win.wav", L"draw.wav", L"nhaccho.wav" };
    if (isSoundOn == true) {
        PlaySound(soundFile[i], NULL, SND_FILENAME | SND_ASYNC);
    }
}
```

### 3.1.8 Hàm getConsoleInput

```
function getConsoleInput() :
    c = readCharacterFromConsole()

    if c is special character :
switch c :
            case 0 or 224:
                secondChar = readCharacterFromConsole()

                    switch secondChar :
            case 72: // Up arrow key
                return 2
            case 75: // Left arrow key
                return 3
            case 77: // Right arrow key
                return 4
            case 80: // Down arrow key
                return 5
            default:
                return 0
            default:
                return 0
    else:
switch c :
            case 27: // Escape key
```

```
            return 1
        case 87 or 119: // W or w
            return 2
        case 65 or 97: // A or a
            return 3
        case 68 or 100: // D or d
            return 4
        case 83 or 115: // S or s
            return 5
        case 13: // Enter key
            return 6
        case 76 or 108: // L or l
            return 7
        case 84 or 116: // T or t
            return 10
        case 89 or 121: // Y or y
            return 11
        case 77 or 109: // M or m
            return 8
        case 78 or 110: // N or n
            return 9
        case 90 or 122: // Z or z
            return 12
        case 69 or 101: // E or e
            return 13
        default:
            return 0
```

### 3.1.9 Hàm instruction

```
function instruction() :
    clearConsole()
    setConsoleColor("FA")
    Draw(6, 20, 1)
    setTextColor(Black)
    gotoXY(55, 12)
    output " This is a 20 x 20 chess board."
    gotoXY(55, 14)
    output " Use the buttons W A S D or arrow buttons to move."
    gotoXY(55, 16)
    output " Use Enter to confirm your move."
    gotoXY(55, 22)
    output " Every 5 X or O on the same horizontal, vertical, diagonal will make
the win."
    gotoXY(55, 24)
    output " The rules of the game follow the rule of blocking 2 heads."
    setTextColor(MintPink)
    gotoXY(65, 18)
    output "   W   "
    setTextColor(Grey)
    gotoXY(65, 20)
    output "   S   "
    setTextColor(BlueYellow)
    gotoXY(55, 20)
    output "   A   "
    setTextColor(100)
```

```
    gotoXY(75, 20)
    output "   D   "
    setTextColor(MintPink)
    gotoXY(94, 18)
    output "  " + char(036) + "  "
    setTextColor(Grey)
    gotoXY(94, 20)
    output "  " + char(037) + "  "
    setTextColor(30)
    gotoXY(87, 20)
    output "  " + char(021) + "  "
    setTextColor(100)
    gotoXY(101, 20)
    output "  " + char(020) + "  "
    setTextColor(GreyBlue)
    gotoXY(118, 18)
    output "         "
    gotoXY(114, 19)
    output "     ENTER   "
    gotoXY(114, 20)
    output "             "
    y = 31
    setTextColor(Red)
    gotoXY(68, 38)
    gotoXY(93, 35)
    output "BACK"
    input = getConsoleInput()

    while true:
if input equals 6 :
    y = y + 1
    if y equals 32 :
        clearConsole()
        menu()
        break
```

### 3.1.10 Hàm About

```
void About()
{
    clearConsole();
    Draw(4, 20, 1);
    int y = 31;
    Textcolor(Red);
    gotoXY(68, 38);
    gotoXY(93, 35);
    cout << "BACK";
    int input = getConsoleInput();
    do if (input == 6)
    {
        y++;
        if (y == 32)
        {
            clearConsole();
            menu();
        }
```

```
        break;

    } while (1);
}
```

### 3.1.11 Hàm history

```
Hàm history():
    ClearConsole()
    system("color FA")
    Draw(9, 5 ,1)
    Textcolor(Green)
    gotoXY(40, 19)
    int i = 15
    Textcolor(Black)
    fstream f
    f.open("Lich Su.txt", ios::in)
    chuỗi line
    int pos = 0
    khi getline(f, line):
        pos++
        nếu pos bằng 10:
            // Xóa nội dung trong tệp tin "Lich Su.txt"
            f.close()
            f.open("Lich Su.txt", ios::out | ios::trunc)
            break;

        chuỗi s, s1, buffer
        size_t pos = tìm vị trí của ' ' trong line
        s = lấy phần tử từ vị trí 0 đến pos trong line
        line = xóa phần tử từ vị trí 0 đến pos trong line
        pos = tìm vị trí của ' ' trong line
        s1 = lấy phần tử từ vị trí 0 đến pos trong line
        line = xóa phần tử từ vị trí 0 đến pos trong line
        buffer = line
        gotoXY(60, i)
        cout << s << " " << s1 << setw(50) << left << buffer
        i++

    int y = 31
    Textcolor(Red)
    gotoXY(68, i+1)
    cout << "Enter: TRỞ LẠI"

    do:
        int input = getConsoleInput()
        nếu input bằng 6:
```

```
            y++
            nếu y bằng 32:
                ClearConsole()
                menu()
            break
    while(true)

kết thúc hàm
```

### 3.1.12 Hàm readMode

```
function readMode(a) :
    declare integer variables d1, d2, and chedo
    open file a in read mode and assign it to variable f
    if the file f does not exist :
        play sound 3
        setTextColor(Black)
        gotoXY(58, 37)
        output " Sorry. The name has not been saved yet !"
        Sleep(1000)
        call the Load() function
        y = 31
        setTextColor(Red)
        gotoXY(68, 38)
        gotoXY(93, 35)
        output "BACK"
        input = getConsoleInput()

        while true:
if input equals 6 :
    increment y by 1
    if y equals 32 :
        clearConsole()
        call the menu() function
        break
        Sleep(1500)
    else:
read d1, d2, and chedo from file f
close file f
return chedo
```

### 3.1.13 Hàm readNameFile

```
function readNameFile() :
    open file "Name.txt" for reading
    initialize i = 15, j = 0

    while not end_of_file(file) :
        increment j
        read a name from the file
        display j and the name at position(60, i)
        increment i

        close the file

        end function
```

### 3.1.14 Hàm Load

```
function Load() :
    clear the console
    call readNameFile() to display names on the console
    initialize data as an array of characters
    initialize ask as an array of strings containing menu options : ["BACK",
"CONTINUE"]
    initialize cur = 60
    initialize input = -1

    while input is not equal to 6 :
        display the menu options on the console at their respective positions
        highlight the current selection based on the value of cur

        input = getConsoleInput() // Get keyboard input

        clear the current selection from the console

        if input is equal to 3 or input is equal to 4:
play a sound
if cur is equal to 60 :
    set cur to 90
else :
    set cur to 60

        else if input is equal to 6:
play a sound
if cur is equal to 60 :
    clear the console
    call the menu() function
else :
    clear the console line at position(38, *)
    display "ENTER FILE NAME: " at position(58, 35)
    show the cursor
    read a line of input and store it in the data array

    chedo = readMode(data)

    if chedo is equal to - 30 or chedo is equal to - 31:
create a Diem object a with score1 = 0 and score2 = 0
t = PlayerVsPlayer(a, chedo, data)
if t is equal to 27 :
    clear the console
    call the menu() function

    if chedo is equal to - 4 :
        create a Diem object a with score1 = 0 and score2 = 0
        t = PlayerVsCom(a, -4, data)
        if t is equal to 27 :
            clear the console
            call the menu() function

            end function
```

### 3.1.15 Hàm settingPlaySound

```
void settingPlaySound() {
    int input = getConsoleInput();

    if (input == 9) {
        isSoundOn = true;
        playSound(6);
    }
    if (input == 8)
    {
        isSoundOn = false;
        PlaySound(0, 0, 0);
    }
}
```

### 3.1.16 Hàm Menu

```
Hàm menu():
    system("color FA") // Thiết lập màu sắc của console
    setConsoleTitle() // Đặt tiêu đề của cửa sổ console
    Khởi tạo biến s là một mảng các ký tự
    Khởi tạo biến x = 70
    Khởi tạo biến y = 19

    // Thiết lập ban đầu cho menu
    Draw(0, 10, 1)

    for i từ 0 đến numItems - 1:
        Đặt màu văn bản thành Đen
        Hiển thị menuItems[i] ở vị trí (x, y + i)

    // Thiết lập bổ sung cho menu
    Đặt màu văn bản thành Đen
    Hiển thị "W - S : Di chuyển (Tắt Unikey)" ở vị trí (x - 5, y + numItems +
6)
    Hiển thị "Enter : Chọn" ở vị trí (x - 2, y + numItems + 7)
    Hiển thị "M: Tắt tiếng" ở vị trí (x, y + numItems + 9)
    Hiển thị "N: Bật tiếng" ở vị trí (x, y + numItems + 10)

    ch = ' ' // Khởi tạo biến đầu vào ký tự

    // Vòng lặp chính để xử lý đầu vào người dùng và lựa chọn menu
    khi ch không bằng 'x':
        // Đánh dấu lựa chọn mới
        Đặt màu văn bản thành Đỏ
        Hiển thị menuItems[currentSelection] ở vị trí (x, y +
currentSelection)
        Đặt màu văn bản thành 15

        input = getConsoleInput() // Nhận đầu vào từ bàn phím

        // Xóa lựa chọn hiện tại
        Đặt màu văn bản thành Đen
        Hiển thị menuItems[currentSelection] ở vị trí (x, y +
currentSelection)
```

```
        nếu input bằng 9:
            isSoundOn = true
            chơi âm thanh(6)

        nếu input bằng 8:
            isSoundOn = false
            chơi âm thanh(0, 0, 0)

        nếu input bằng 5 hoặc ch bằng 's':
            chơi âm thanh(1)
            // Di chuyển xuống một hàng
            currentSelection = (currentSelection + 1) % numItems

        nếu input bằng 2 hoặc ch bằng 'w':
            chơi âm thanh(1)
            // Di chuyển lên một hàng
            currentSelection = (currentSelection - 1 + numItems) % numItems

        nếu input bằng 6 hoặc ch bằng '\r': // Kiểm tra phím Enter được nhấn
            chơi âm thanh(2)
            // Gọi hàm tương ứng với mục menu được chọn
            switch currentSelection:
                trường hợp 0:
                    xóaConsole()
                    tạo một đối tượng Diem a với score1 = 0 và score2 = 0
                    t = PlayerVsPlayer(a, 0, s)
                    nếu t bằng 27:
                        xóaConsole()
                        menu()
                   break

                trường hợp 1:
                    xóaConsole()
                    tạo một đối tượng Diem a với score1 = 0 và score2 = 0
                    t = PlayerVsCom(a, 0, s)
                    nếu t bằng 27:
                        xóaConsole()
                        menu()
                    break

                trường hợp 2:
                    xóaConsole()
                    gọi Load()
                    break

                trường hợp 3:
                    xóaConsole()
                    gọi history()
                    break

                trường hợp 4:
                    xóaConsole()
                    gọi instruction()
                    break
```

```
                trường hợp 5:
                    xóaConsole()
                    gọi About()
                    break

                trường hợp 6:
                    xóaConsole()
                    thoát(0)
                    break

    kết thúc hàm
```

## 4. Nhóm hàm xử lí dữ liệu game của file Game.h

### 4.1*Tóm tắt sơ bộ về nội dung và ý tưởng thuật toán*

Game Cờ Caro được xây dựng bằng ngôn ngữ C++. Game có 2 chế độ: người đánh với người, người đánh với máy.
Thuật giải áp dụng:
- Vét cạn thông minh Heuristic (tìm nước đi cho máy).
- Các thuật giải cơ bản và kỹ thuật khác.

Trò chơi sử dụng 2 giải thuật chính
♣ Kiểm tra thắng thua. (Chặn 2 đầu)
- Duyệt theo chiều dọc
- Duyệt theo chiều ngang
- Duyệt theo chiều chéo xuôi
- Duyệt theo chiều chéo ngược

♣ Tìm nước đi cho máy:
- Phòng thủ
- Tấn công Vẹt cạn các ô trống và đưa ra việc tính điểm cho từng ô để tìm ô trống phù hợp cho nước đi kế tiếp là việc tấn công hay phòng thủ và lưu lại tọa độ x y của ô trống để cho máy đánh.

Các kỹ thuật phụ:
♣ Kỹ thuật di chuyển con trỏ, ẩn con trỏ, hiện con trỏ.
♣ Kỹ thuật chèn âm thanh, hiệu ứng cho game sinh động.
♣ Kỹ thuật tô màu kí tự và căn chỉnh màn hình console. (Dùng để làm giao diện và hiệu ứng thắng thua)
♣ Kỹ thuật đọc, ghi file (dùng để Load Game và Save Game).
♣ Kỹ thuật Get Set trong các lớp để lấy thuộc tính sử dụng trong bàn cờ.
♣ Kỹ thuật xử lí tạo Menu Game.
♣ Kỹ thuật đệ quy (để quay trở về Menu và thực hiện tiếp)

♣ Sử dụng vòng lặp và lệnh Sleep () để tạo hiệu ứng chữ chớp.
Sau đây là trích đoạn code sử dụng để khai báo hàm và chức năng của từng hàm:

***4.1 Miêu tả sơ bộ về các nhóm hàm chính về thuật toán trò chơi trong file Game.h / Game.cpp***

```
#include <iostream>
#include <Windows.h>
#include <conio.h>
#include <mmsystem.h>
#include <vector>
#include <fstream>
#include <stack>
using namespace std;


struct Diem
{
	int score1;
	int score2;
};

struct _Point
{
	// Tọa độ x y trên bàn cờ
	int _x;
	int _y;

	// Biến nhận biết X và O ( -1 = X , 1 = O, 0 = Ô trống )
	int _check;

};

static _Point** _pArr;

// Cài đặt biến _check
bool setCheck(int pCheck, int i, int j);
// Lấy giá trị của _x
int getX(int i, int j);
// Lấy giá trị của _y
int getY(int i, int j);
// Lấy giá trị của _check
int getCheck(int i, int j);
// Cài đặt biến _x
void setX(int pX, int i, int j);
// Cài đặt biến _y
void setY(int pY, int i, int j);

_Point Tim_Kiem_NuocDi_1(); // Tìm nước đi cho máy

struct _Board
{
	// Kích thước bàn cờ (size x size)
	int _size;
	// Tọa độ phía bên trái và trên bàn cờ.
	int _left;
	int _top;
```

```
        // Mảng 2 chiều để chuyển đổi tọa độ (x,y) thành các ô trong mảng 2 chiều.
        int CountX; // Đếm nước cờ X
        int CountY; // Đếm nước cờ O
};
static _Board* _b;        // Khởi tạo 1 bàn cờ

// Lấy giá trị _size
template <class T>
int getSize(T* _b);
// Lấy giá trị _left
template <class T>
int getLeft(T* _b);
// Lấy giá trị _top
template <class T>
int getTop(T* _b);
// Lấy tọa độ x,y tại vị trí i,j trên bàn cờ mảng 2 chiều _pArr
int getXAt(int, int);
int getYAt(int, int);
// Lấy giá trị _check trên mảng 2 chiều . nhận biết X O và ô trống.
static int get_Check(int i, int j) { return getCheck(i, j); }
// Load dữ liệu
void loadData(int, int, int);
// Reset bàn cờ cho tất cả các ô trống _check = 0.
void resetData();
// Kiểm tra X hay O
int checkBoard(int, int, bool);
// Kiểm tra thắng thua trên bàn cờ
int testBoard(int x, int y);


long SoDiemTanCong_DuyetDoc(long, long, const long Defend_Score[], const long
Attack_Score[]);
long SoDiemTanCong_DuyetNgang(long, long, const long Defend_Score[], const long
Attack_Score[]);
long SoDiemTanCong_DuyetCheo1(long, long, const long Defend_Score[], const long
Attack_Score[]);
long SoDiemTanCong_DuyetCheo2(long, long, const long Defend_Score[], const long
Attack_Score[]);
long SoDiemPhongThu_DuyetDoc(long, long, const long Defend_Score[], const long
Attack_Score[]);
long SoDiemPhongThu_DuyetNgang(long, long, const long Defend_Score[], const long
Attack_Score[]);
long SoDiemPhongThu_DuyetCheo1(long, long, const long Defend_Score[], const long
Attack_Score[]);
long SoDiemPhongThu_DuyetCheo2(long, long, const long Defend_Score[], const long
Attack_Score[]);
// Duyệt Các Ô Trống tính điểm cho từng ô theo dọc , ngang , chéo ngược , chéo xuôi.

// Kiểm tra thắng theo dòng
int checkWinRow(int x, int y, int value); // value (-1 hoac 1) hay X hoac O
// Kiểm tra thắng theo cột
int checkWinCol(int x, int y, int value);
// Kiểm tra thắng theo đường chéo thứ 1
int checkfirstDiagonal(int x, int y, int value);
// Kiểm tra thắng theo đường chéo thứ 2
int checksecondDiagonal(int x, int y, int value);
```

```
struct _Game
{

	bool _turn;				// True là lượt người chơi 1 , false là người chơi
2.
	int _x, _y;				// Tọa độ
	bool _loop;				// True chơi tiếp, False out.
	int scorep1;			// Số trận thắng P1
	int scorep2;			// Số trận thắng P2
	int chedo;				// Đọc file để nhận biết chế độ chơi
	// -31 : Chế độ P vs P đang đến lượt X
	// -30 : Chế độ P vs P đang đến lượt O
	// -4  : Chế độ P vs Bot (Dễ) đang đến lượt X
	// -5  : Chế độ P vs Bot (Khó) đang đến lượt X

	// Đêm số lương x y trong bàn cờ
	int CountX;
	int CountY;

	// biến để hiện trước x o đi trước
	bool _changeTurn;
	bool _showCursor;

	vector<pair<int, int>> moves; // danh sách các nước cờ đã đi
};
static _Game* g;


// Lấy tỉ số thắng
static int getScore1() { return g->scorep1; }
static int getScore2() { return g->scorep2; }
// Cài đặt tỉ số thắng = 0 .
static void setScore1() { g->scorep1 = 0; }
static void setScore2() { g->scorep2 = 0; }
// Trò chơi tiếp tục
static bool isContinue();
// Chơi lại hay không ?
static char askContinue();


void startGame(); // Khởi tạo game . Bắt đầu game
void exitGame(); // Thoát Game
void SaveGame(int n);  // Lưu Game đang chơi với biến n là chế độ và lượt .
void LoadGame(char data[30]); // Khởi tạo game . Bắt đầu game ( trường hợp Load Game
)
void LichSuGame(int n);

// Kiểm tra thắng thua - tiếp tục
int processFinish(int x, int y);
// Đánh dấu X và O trên bàn cờ
bool processCheckBoard();

// Di chuyển lên - xuống - trái - phải
void moveRight();
void moveLeft();
void moveUp();
void moveDown();
```

```
// Cài đặt biến _x _y
static void gsetX(int x) { g->_x = x; }
static void gsetY(int y) { g->_y = y; }

// Lấy giá trị _x _y
int getXatEnter();
int getYatEnter();
// Lấy giá trị của lượt chơi .
static bool getTurn() { return g->_turn; }

// Đảo lượt chơi
static void setTurn() { g->_turn = !g->_turn; }
// Tìm kiếm nước đi cho máy
void TimKiemNuocDi();
// Đếm nước cờ đã đi được
int DemNuocCoDaDi();
// set up các chỉ cho biến ở struct game
static void setGame(int pSize, int pLeft, int  pTop);
// cài đặt số x y = 0
template <class T>
static void gsetCountXY(T* g);



const long Defend_Score1[7] = { 0, 72, 9216, 589824, 37748736, 2415919104 };
const long Attack_Score1[7] = { 0, 576, 73728, 4718592, 301989888, 19327352832 };


//const  long Defend_Score1[7] = { 0, 8, 512, 32768, 2097152, 134217728 };
//const  long Attack_Score1[7] = { 0, 64, 4096, 262144, 16777216, 1073741824 };
// số nhỏ hơn cho game


int PlayerVsCom(Diem& a, int, char data[30]); // BOT
int PlayerVsPlayer(Diem& a, int, char data[30]); // P VS P

// hàm tắt của PVP và PVC
void PvPaskForRestart(Diem& a, int& load, char data[30]);
void PvCaskForRestart(Diem& a, int& load, char data[30]);

static int input = -1;
static bool flagSymbol = true;
// xóa dòng
void clearConsoleLine(int y);

// các hàm xây dựng show x y trước khi đánh tại vị trí trỏ tới
void printTurnSymbol();
int getCheckAtXY(int pX, int pY);
void showCursor(bool show);

// chức năng undo
void changeTurn();
void deleteXO(int _x, int _y, char c);
void undo(int x, int y);
void undoLastMove();
void addMove(int x, int y);

void goiY();
```

```
void undoSetting(int);
```

### 4.1.1 Hàm LoadData

```
function loadData(i, j, k) :
    nếu _b->_size bằng 0, trả về

    // Gán tọa độ x trên bàn cờ bằng giá trị từ chỉ số i trong mảng 2 chiều
    setX(4 * j + _b->_left + 2, i, j)
    // Gán tọa độ y trên bàn cờ bằng giá trị từ chỉ số j trong mảng 2 chiều
    setY(2 * i + _b->_top + 1, i, j)
    // Gán trạng thái của ô trên bàn cờ (X, O, ô trống) vào các phần tử tương ứng
trong mảng
    setCheck(k, i, j)

    // Nếu trạng thái của ô là X, hiển thị X lên màn hình và tăng biến đếm X lên 1
    nếu k bằng - 1:
Textcolor(Red)
gotoXY(4 * j + _b->_left + 2, 2 * i + _b->_top + 1)
hiển thị "X"
tăng _b->CountX lên 1

// Nếu trạng thái của ô là O, hiển thị O lên màn hình và tăng biến đếm O lên 1
nếu k bằng 1:
Textcolor(Blue)
gotoXY(4 * j + _b->_left + 2, 2 * i + _b->_top + 1)
hiển thị "O"
tăng _b->CountY lên 1
end function
```

### 4.1.2 Hàm resetData

```
function resetData() :
    nếu _b->_size bằng 0, trả về

    // Duyệt qua tất cả các ô trên bàn cờ
    cho i chạy từ 0 đến _b->_size - 1 :
    cho j chạy từ 0 đến _b->_size - 1 :
    // Gán tọa độ x trên bàn cờ bằng giá trị từ chỉ số i trong mảng 2 chiều
    setX(4 * j + _b->_left + 2, i, j)
    // Gán tọa độ y trên bàn cờ bằng giá trị từ chỉ số j trong mảng 2 chiều
    setY(2 * i + _b->_top + 1, i, j)
    // Gán trạng thái của ô thành ô trống -> Reset lại bàn cờ
    setCheck(0, i, j)
    end for
 end function
```

### 4.1.3 Hàm CheckBoard

```
function checkBoard(pX, pY, pTurn) :
    // Duyệt qua tất cả các ô trên bàn cờ
```

```
    cho i chạy từ 0 đến _b->_size - 1 :
    cho j chạy từ 0 đến _b->_size - 1 :
    // Kiểm tra ô có tọa độ (pX, pY) và chưa được đánh
    nếu getX(i, j) bằng pX và getY(i, j) bằng pY và getCheck(i, j) bằng 0 :
    // Nếu là chế độ PVP -> Người chơi X đánh trước
    nếu pTurn :
setCheck(-1, i, j)
tăng _b->CountX lên 1
// Nếu là chế độ PVE (Bot) -> Máy (O) đánh trước
ngược lại :
setCheck(1, i, j)
tăng _b->CountY lên 1
// Trả về kết quả xem ai đánh trước (X hay O)
trả về getCheck(i, j)
end if
end for
end for
trả về 0
end function
```

### 4.1.4 Hàm checkWinRow (Check chiến thắng trên dòng)

```
function checkWinRow(x, y, value) :
    khai báo các biến cục bộ :
int dong, cot
int loop = 1
int test = -1
int check2dau = 0
int countRowLeft = 0, countRowRight = 0
int dongtrai, dongphai // Xét đếm X (hoặc O) về phía trái và về phía phải
đặt dong = (x - _b->_left - 2) / 4
đặt cot = (y - _b->_top - 1) / 2
đặt dongtrai = dongphai = dong

// Xét về phía trái ô đang xét
khi getCheck(cot, dongtrai) bằng value :
nếu dongtrai == 0 :
    break
    đẩy _pArr[cot][dongtrai] vào win
    tăng countRowLeft lên 1
    giảm dongtrai đi 1

    // Xét về phía phải ô đang xét
    khi getCheck(cot, dongphai) bằng value :
nếu dongphai == _b->_size - 1 :
    break
    đẩy _pArr[cot][dongphai] vào win
    tăng countRowRight lên 1
    tăng dongphai lên 1

    // Kiểm tra trường hợp chặn 2 đầu
    nếu getCheck(cot, dongtrai) == -value và getCheck(cot, dongphai) == -value:
đặt check2dau = 1
```

```
// Kiểm tra chiến thắng
nếu(countRowLeft + countRowRight – 1) bằng 5 và check2dau bằng 0 :
    đặt test = 1
    ngược lại :
đặt test = 0

// Xử lý chiến thắng
nếu test bằng 1 :
    phát ra âm thanh chiến thắng
    lặp cho đến khi loop < 30 :
    đặt Textcolor ngẫu nhiên từ 1 đến 15
    cho i chạy từ 0 đến kích thước của win – 1 :
    điều chỉnh vị trí con trỏ đến win[i]._x, win[i]._y
    nếu win[i]._check bằng – 1 :
    in ra màn hình ký tự "X"
    ngược lại :
in ra màn hình ký tự "O"
tạm dừng 100ms
tăng loop lên 1

ngược lại :
xóa hết phần tử trong win

trả về test
end function
```

### 4.1.5 Hàm CheckWinCol

```
function checkWinCol(x, y, value) :
    khai báo các biến cục bộ :
int dong, cot
int test = –1, loop = 1
int check2dau = 0
int countColTop = 0, countColBot = 0
đặt dong = (x – _b->_left – 2) / 4
đặt cot = (y – _b->_top – 1) / 2
int cottren, cotduoi
đặt cottren = cotduoi = cot

// Xét lên trên ô đang xét
khi getCheck(cottren, dong) bằng value :
tăng countColTop lên 1
đẩy _pArr[cottren][dong] vào win
nếu cottren == 0 :
    break
    giảm cottren đi 1

    // Xét xuống dưới ô đang xét
    khi getCheck(cotduoi, dong) bằng value :
tăng countColBot lên 1
đẩy _pArr[cotduoi][dong] vào win
nếu cotduoi == _b->_size – 1 :
    break
    tăng cotduoi lên 1
```

```
    // Kiểm tra trường hợp chặn 2 đầu
    nếu getCheck(cottren, dong) == -value và getCheck(cotduoi, dong) == -value :
    đặt check2dau = 1

    // Kiểm tra chiến thắng
    nếu(countColTop + countColBot - 1) bằng 5 và check2dau bằng 0 :
    đặt test = 1
    ngược lại :
đặt test = 0

// Xử lý chiến thắng
nếu test bằng 1 :
    lặp cho đến khi loop < 30 :
    đặt Textcolor ngẫu nhiên từ 1 đến 15
    cho i chạy từ 0 đến kích thước của win - 1 :
    điều chỉnh vị trí con trỏ đến win[i]._x, win[i]._y
    nếu win[i]._check bằng - 1 :
    in ra màn hình ký tự "X"
    ngược lại :
in ra màn hình ký tự "O"
tạm dừng 100ms
tăng loop lên 1

ngược lại :
xóa hết phần tử trong win

trả về test
end function
```

### 4.1.6 Hàm checksecondDiagonal// Hàm check chiến thắng trên đường chéo phụ

```
function checksecondDiagonal(x, y, value) :
    khai báo các biến cục bộ :
int dong, cot
int test = -1, loop = 1
int check2dau = 0
int countDiaTop = 0, countDiaBot = 0
đặt dong = (x - _b->_left - 2) / 4
đặt cot = (y - _b->_top - 1) / 2
int cottren, cotduoi
int dongphai, dongtrai
đặt cottren = cotduoi = cot
đặt dongphai = dongtrai = dong

// Xét đường chéo phụ phía trên ô đang xét
khi getCheck(cottren, dongtrai) bằng value :
đẩy _pArr[cottren][dongtrai] vào win
tăng countDiaTop lên 1
nếu cottren == 0 hoặc dongtrai == 0 :
    break
    giảm cottren đi 1
    giảm dongtrai đi 1
```

```
    // Xét đường chéo phụ phía dưới ô đang xét
    khi getCheck(cotduoi, dongphai) bằng value :
đẩy _pArr[cotduoi][dongphai] vào win
tăng countDiaBot lên 1
nếu cotduoi == _b->_size - 1 hoặc dongphai == _b->_size - 1 :
    break
    tăng cotduoi lên 1
    tăng dongphai lên 1

    // Kiểm tra trường hợp chặn 2 đầu
    nếu getCheck(cottren, dongtrai) == -value và getCheck(cotduoi, dongphai) == -
value:
đặt check2dau = 1

// Kiểm tra chiến thắng
nếu(countDiaTop + countDiaBot - 1) bằng 5 và check2dau bằng 0 :
    đặt test = 1
    ngược lại :
đặt test = 0

// Xử lý chiến thắng
nếu test bằng 1 :
    lặp cho đến khi loop < 30 :
    đặt Textcolor ngẫu nhiên từ 1 đến 15
    cho i chạy từ 0 đến kích thước của win - 1 :
    điều chỉnh vị trí con trỏ đến win[i]._x, win[i]._y
    nếu win[i]._check bằng - 1 :
    in ra màn hình ký tự "X"
    ngược lại :
in ra màn hình ký tự "O"
tạm dừng 100ms
tăng loop lên 1

ngược lại :
xóa hết phần tử trong win

trả về test
end function
```

### 4.1.7 Hàm checkfirstDiagonal // Hàm check chiến thắng trên đường chéo chính

```
function checkfirstDiagonal(x, y, value) :
    khai báo các biến cục bộ :
int dong, cot
int test = -1, loop = 1
int check2dau = 0
int countDiaTop = 0, countDiaBot = 0
đặt dong = (x - _b->_left - 2) / 4
đặt cot = (y - _b->_top - 1) / 2
int cottren, cotduoi
int dongphai, dongtrai
đặt cottren = cotduoi = cot
đặt dongphai = dongtrai = dong

// Xét đường chéo chính phía trên ô đang xét
khi getCheck(cottren, dongphai) bằng value :
tăng countDiaTop lên 1
```

```
đẩy _pArr[cottren][dongphai] vào win
nếu cottren == 0 hoặc dongphai == _b->_size - 1 :
    break
    giảm cottren đi 1
    tăng dongphai lên 1

    // Xét đường chéo chính phía dưới ô đang xét
    khi getCheck(cotduoi, dongtrai) bằng value :
tăng countDiaBot lên 1
đẩy _pArr[cotduoi][dongtrai] vào win
nếu cotduoi == _b->_size - 1 hoặc dongtrai == 0 :
    break
    tăng cotduoi lên 1
    giảm dongtrai đi 1

    // Kiểm tra trường hợp chặn 2 đầu
    nếu getCheck(cottren, dongphai) == -value và getCheck(cotduoi, dongtrai) == -
value:
đặt check2dau = 1

// Kiểm tra chiến thắng
nếu(countDiaTop + countDiaBot - 1) bằng 5 và check2dau bằng 0 :
    đặt test = 1
    ngược lại :
đặt test = 0

// Xử lý chiến thắng
nếu test bằng 1 :
    lặp cho đến khi loop < 30 :
    đặt Textcolor ngẫu nhiên từ 1 đến 15
    cho i chạy từ 0 đến kích thước của win - 1 :
    điều chỉnh vị trí con trỏ đến win[i]._x, win[i]._y
    nếu win[i]._check bằng - 1 :
    in ra màn hình ký tự "X"
    ngược lại :
in ra màn hình ký tự "O"
tạm dừng 100ms
tăng loop lên 1

ngược lại :
xóa hết phần tử trong win

trả về test
end function
```

Từ các hàm ở mục 4.1.8 trở đi, ta sẽ tập trung vào thuật toán tìm nước đi tốt nhất đối với các chế độ chơi Người với Máy và cũng như chức năng Tìm gợi ý nước đi.

### 4.1.8 Hàm SoDiemTanCong_DuyetNgang

```
function SoDiemTanCong_DuyetNgang(Dong, Cot, Defend_Score, Attack_Score) :
    khai báo các biến cục bộ :
long iScoreTempNgang = 0
long iScoreAttack = 0
int iSoQuanTa = 0
```

```
int iSoQuanDich = 0
int iSoQuanTa2 = 0
int iSoQuanDich2 = 0

// Duyệt sang phải
cho iDem chạy từ 1 đến 5 và Cot + iDem < _b->_size:
nếu getCheck(Dong, Cot + iDem) bằng 1 :
    tăng iSoQuanTa lên 1
    nếu getCheck(Dong, Cot + iDem) bằng - 1 :
    tăng iSoQuanDich lên 1
    thoát vòng lặp
    nếu getCheck(Dong, Cot + iDem) bằng 0 :
    cho iDem2 chạy từ 2 đến 6 và Cot + iDem2 < _b->_size :
    nếu getCheck(Dong, Cot + iDem2) bằng 1 :
    tăng iSoQuanTa2 lên 1
    nếu getCheck(Dong, Cot + iDem2) bằng - 1 :
    tăng iSoQuanDich2 lên 1
    thoát vòng lặp
    nếu getCheck(Dong, Cot + iDem2) bằng 0 :
    thoát vòng lặp
    thoát vòng lặp

    // Duyệt sang trái
    cho iDem chạy từ 1 đến 5 và Cot - iDem >= 0 :
    nếu getCheck(Dong, Cot - iDem) bằng 1 :
    tăng iSoQuanTa lên 1
    nếu getCheck(Dong, Cot - iDem) bằng - 1 :
    tăng iSoQuanDich lên 1
    thoát vòng lặp
    nếu getCheck(Dong, Cot - iDem) bằng 0 :
    cho iDem2 chạy từ 2 đến 6 và Cot - iDem2 >= 0 :
    nếu getCheck(Dong, Cot - iDem2) bằng 1 :
    tăng iSoQuanTa2 lên 1
    nếu getCheck(Dong, Cot - iDem2) bằng - 1 :
    tăng iSoQuanDich2 lên 1
    thoát vòng lặp
    nếu getCheck(Dong, Cot - iDem2) bằng 0 :
    thoát vòng lặp
    thoát vòng lặp

    nếu iSoQuanDich bằng 2 :
    trả về 0

    nếu iSoQuanDich bằng 0 :
    iScoreTempNgang += Attack_Score[iSoQuanTa] * 2
    ngược lại :
iScoreTempNgang += Attack_Score[iSoQuanTa]

nếu iSoQuanDich2 bằng 0 :
    iScoreTempNgang += Attack_Score[iSoQuanTa2] * 2
    ngược lại :
iScoreTempNgang += Attack_Score[iSoQuanTa2]

nếu iSoQuanTa >= iSoQuanTa2 :
    iScoreTempNgang -= 1
    ngược lại :
iScoreTempNgang -= 2
```

```
nếu iSoQuanTa bằng 4 :
    iScoreTempNgang *= 2

    nếu iSoQuanTa bằng 0 :
    iScoreTempNgang += Defend_Score[iSoQuanDich] * 2
    ngược lại :
iScoreTempNgang += Defend_Score[iSoQuanDich]

nếu iSoQuanTa2 bằng 0 :
    iScoreTempNgang += Defend_Score[iSoQuanDich2] * 2
    ngược lại :
iScoreTempNgang += Defend_Score[iSoQuanDich2]

trả về iScoreTempNgang
end function
```

### 4.1.9 Hàm SoDiemTanCong_DuyetDoc

```
function SoDiemTanCong_DuyetDoc(Dong, Cot, Defend_Score, Attack_Score):
    khai báo các biến cục bộ:
        long iScoreTempDoc = 0
        long iScoreAttack = 0
        int iSoQuanTa = 0
        int iSoQuanDich = 0
        int iSoQuanTa2 = 0
        int iSoQuanDich2 = 0

    // Duyệt xuống dưới
    cho iDem chạy từ 1 đến 5 và Dong + iDem < _b->_size:
        nếu getCheck(Dong + iDem, Cot) bằng 1:
            tăng iSoQuanTa lên 1
        nếu getCheck(Dong + iDem, Cot) bằng -1:
            tăng iSoQuanDich lên 1
            thoát vòng lặp
        nếu getCheck(Dong + iDem, Cot) bằng 0:
            cho iDem2 chạy từ 2 đến 6 và Dong + iDem2 < _b->_size:
                nếu getCheck(Dong + iDem2, Cot) bằng 1:
                    tăng iSoQuanTa2 lên 1
                nếu getCheck(Dong + iDem2, Cot) bằng -1:
                    tăng iSoQuanDich2 lên 1
                    thoát vòng lặp
                nếu getCheck(Dong + iDem2, Cot) bằng 0:
                    thoát vòng lặp
            thoát vòng lặp

    // Duyệt lên trên
    cho iDem chạy từ 1 đến 5 và Dong - iDem >= 0:
        nếu getCheck(Dong - iDem, Cot) bằng 1:
            tăng iSoQuanTa lên 1
        nếu getCheck(Dong - iDem, Cot) bằng -1:
            tăng iSoQuanDich lên 1
            thoát vòng lặp
        nếu getCheck(Dong - iDem, Cot) bằng 0:
            cho iDem2 chạy từ 2 đến 6 và Dong - iDem2 >= 0:
                nếu getCheck(Dong - iDem2, Cot) bằng 1:
                    tăng iSoQuanTa2 lên 1
                nếu getCheck(Dong - iDem2, Cot) bằng -1:
```

```
                    tăng iSoQuanDich2 lên 1
                    thoát vòng lặp
                nếu getCheck(Dong - iDem2, Cot) bằng 0:
                    thoát vòng lặp
            thoát vòng lặp

    nếu iSoQuanDich bằng 2:
        trả về 0

    nếu iSoQuanDich bằng 0:
        iScoreTempDoc += Attack_Score[iSoQuanTa] * 2
    ngược lại:
        iScoreTempDoc += Attack_Score[iSoQuanTa]

    nếu iSoQuanDich2 bằng 0:
        iScoreTempDoc += Attack_Score[iSoQuanTa2] * 2
    ngược lại:
        iScoreTempDoc += Attack_Score[iSoQuanTa2]

    nếu iSoQuanTa >= iSoQuanTa2:
        iScoreTempDoc -= 1
    ngược lại:
        iScoreTempDoc -= 2

    nếu iSoQuanTa bằng 4:
        iScoreTempDoc *= 2

    nếu iSoQuanTa bằng 0:
        iScoreTempDoc += Defend_Score[iSoQuanDich] * 2
    ngược lại:
        iScoreTempDoc += Defend_Score[iSoQuanDich]

    nếu iSoQuanTa2 bằng 0:
        iScoreTempDoc += Defend_Score[iSoQuanDich2] * 2
    ngược lại:
        iScoreTempDoc += Defend_Score[iSoQuanDich2]

    trả về iScoreTempDoc
end function
```

### 4.1.10 Hàm SoDiemTanCong_DuyetCheo1

```
function SoDiemTanCong_DuyetCheo1(Dong, Cot, Defend_Score, Attack_Score):
    khai báo các biến cục bộ:
        long iScoreTempCheoNguoc = 0
        long iScoreAttack = 0
        int iSoQuanTa = 0
        int iSoQuanDich = 0
        int iSoQuanTa2 = 0
        int iSoQuanDich2 = 0

    // Duyệt theo đường chéo ngược (\)
    cho iDem chạy từ 1 đến 5 và Cot + iDem < _b->_size và Dong + iDem < _b->_size:
        nếu getCheck(Dong + iDem, Cot + iDem) bằng 1:
            tăng iSoQuanTa lên 1
        nếu getCheck(Dong + iDem, Cot + iDem) bằng -1:
            tăng iSoQuanDich lên 1
```

```
            thoát vòng lặp
        nếu getCheck(Dong + iDem, Cot + iDem) bằng 0:
            cho iDem2 chạy từ 2 đến 6 và Cot + iDem2 < _b->_size và Dong + iDem2 <
_b->_size:
                nếu getCheck(Dong + iDem2, Cot + iDem2) bằng 1:
                    tăng iSoQuanTa2 lên 1
                nếu getCheck(Dong + iDem2, Cot + iDem2) bằng -1:
                    tăng iSoQuanDich2 lên 1
                    thoát vòng lặp
                nếu getCheck(Dong + iDem2, Cot + iDem2) bằng 0:
                    thoát vòng lặp
            thoát vòng lặp

    // Duyệt theo đường chéo ngược (--)
    cho iDem chạy từ 1 đến 5 và Cot - iDem >= 0 và Dong - iDem >= 0:
        nếu getCheck(Dong - iDem, Cot - iDem) bằng 1:
            tăng iSoQuanTa lên 1
        nếu getCheck(Dong - iDem, Cot - iDem) bằng -1:
            tăng iSoQuanDich lên 1
            thoát vòng lặp
        nếu getCheck(Dong - iDem, Cot - iDem) bằng 0:
            cho iDem2 chạy từ 2 đến 6 và Cot - iDem2 >= 0 và Dong - iDem2 >= 0:
                nếu getCheck(Dong - iDem2, Cot - iDem2) bằng 1:
                    tăng iSoQuanTa2 lên 1
                nếu getCheck(Dong - iDem2, Cot - iDem2) bằng -1:
                    tăng iSoQuanDich2 lên 1
                    thoát vòng lặp
                nếu getCheck(Dong - iDem2, Cot - iDem2) bằng 0:
                    thoát vòng lặp
            thoát vòng lặp

    nếu iSoQuanDich bằng 2:
        trả về 0
    nếu iSoQuanDich bằng 0:
        iScoreTempCheoNguoc += Attack_Score[iSoQuanTa] * 2
    ngược lại:
        iScoreTempCheoNguoc += Attack_Score[iSoQuanTa]

    nếu iSoQuanDich2 bằng 0:
        iScoreTempCheoNguoc += Attack_Score[iSoQuanTa2] * 2
    ngược lại:
        iScoreTempCheoNguoc += Attack_Score[iSoQuanTa2]

    nếu iSoQuanTa >= iSoQuanTa2:
        iScoreTempCheoNguoc -= 1
    ngược lại:
        iScoreTempCheoNguoc -= 2

    nếu iSoQuanTa bằng 4:
        iScoreTempCheoNguoc *= 2

    nếu iSoQuanTa bằng 0:
        iScoreTempCheoNguoc += Defend_Score[iSoQuanDich] * 2
    ngược lại:
        iScoreTempCheoNguoc += Defend_Score[iSoQuanDich]

    nếu iSoQuanTa2 bằng 0:
        iScoreTempCheoNguoc += Defend_Score[iSoQuanDich2] * 2
```

```
    ngược lại:
        iScoreTempCheoNguoc += Defend_Score[iSoQuanDich2]

    trả về iScoreTempCheoNguoc
end function
```

### 4.1.11 Hàm SoDiemTanCong_DuyetCheo2

```
function SoDiemTanCong_DuyetCheo2(Dong, Cot, Defend_Score, Attack_Score):
    khai báo các biến cục bộ:
        long iScoreTempCheoXuoi = 0
        long iScoreAttack = 0
        int iSoQuanTa = 0
        int iSoQuanDich = 0
        int iSoQuanTa2 = 0
        int iSoQuanDich2 = 0

    // Duyệt theo đường chéo xuôi (/)
    cho iDem chạy từ 1 đến 5 và Cot - iDem >= 0 và Dong + iDem < _b->_size:
        nếu getCheck(Dong + iDem, Cot - iDem) bằng 1:
            tăng iSoQuanTa lên 1
        nếu getCheck(Dong + iDem, Cot - iDem) bằng -1:
            tăng iSoQuanDich lên 1
            thoát vòng lặp
        nếu getCheck(Dong + iDem, Cot - iDem) bằng 0:
            cho iDem2 chạy từ 2 đến 6 và Cot - iDem2 >= 0 và Dong + iDem2 < _b-
>_size:
                nếu getCheck(Dong + iDem2, Cot - iDem2) bằng 1:
                    tăng iSoQuanTa2 lên 1
                nếu getCheck(Dong + iDem2, Cot - iDem2) bằng -1:
                    tăng iSoQuanDich2 lên 1
                    thoát vòng lặp
                nếu getCheck(Dong + iDem2, Cot - iDem2) bằng 0:
                    thoát vòng lặp
            thoát vòng lặp

    // Duyệt theo đường chéo ngược (--)
    cho iDem chạy từ 1 đến 5 và Cot + iDem < _b->_size và Dong - iDem >= 0:
        nếu getCheck(Dong - iDem, Cot + iDem) bằng 1:
            tăng iSoQuanTa lên 1
        nếu getCheck(Dong - iDem, Cot + iDem) bằng -1:
            tăng iSoQuanDich lên 1
            thoát vòng lặp
        nếu getCheck(Dong - iDem, Cot + iDem) bằng 0:
            cho iDem2 chạy từ 2 đến 6 và Cot + iDem2 < _b->_size và Dong - iDem2
>= 0:
                nếu getCheck(Dong - iDem2, Cot + iDem2) bằng 1:
                    tăng iSoQuanTa2 lên 1
                nếu getCheck(Dong - iDem2, Cot + iDem2) bằng -1:
                    tăng iSoQuanDich2 lên 1
                    thoát vòng lặp
                nếu getCheck(Dong - iDem2, Cot + iDem2) bằng 0:
                    thoát vòng lặp
            thoát vòng lặp

    nếu iSoQuanDich bằng 2:
        trả về 0
```

```
    nếu iSoQuanDich bằng 0:
        iScoreTempCheoXuoi += Attack_Score[iSoQuanTa] * 2
    ngược lại:
        iScoreTempCheoXuoi += Attack_Score[iSoQuanTa]

    nếu iSoQuanDich2 bằng 0:
        iScoreTempCheoXuoi += Attack_Score[iSoQuanTa2] * 2
    ngược lại:
        iScoreTempCheoXuoi += Attack_Score[iSoQuanTa2]

    nếu iSoQuanTa >= iSoQuanTa2:
        iScoreTempCheoXuoi -= 1
    ngược lại:
        iScoreTempCheoXuoi -= 2

    nếu iSoQuanTa bằng 4:
        iScoreTempCheoXuoi *= 2

    nếu iSoQuanTa bằng 0:
        iScoreTempCheoXuoi += Defend_Score[iSoQuanDich] * 2
    ngược lại:
        iScoreTempCheoXuoi += Defend_Score[iSoQuanDich]

    nếu iSoQuanTa2 bằng 0:
        iScoreTempCheoXuoi += Defend_Score[iSoQuanDich2] * 2
    ngược lại:
        iScoreTempCheoXuoi += Defend_Score[iSoQuanDich2]

    trả về iScoreTempCheoXuoi
end function
```

### 4.1.12 Hàm SoDiemPhongThu_DuyetDoc

```
function SoDiemPhongThu_DuyetDoc(Dong, Cot, Defend_Score, Attack_Score):
    khai báo các biến cục bộ:
        long iScoreTempDoc = 0
        long iScoreDefend = 0
        int iSoQuanDich = 0
        int iSoQuanTa = 0
        int iSoQuanDich2 = 0
        int iSoQuanTa2 = 0

    // Duyệt theo chiều dọc (dòng)
    cho iDem chạy từ 1 đến 5 và Dong + iDem < _b->_size:
        nếu getCheck(Dong + iDem, Cot) bằng 1:
            tăng iSoQuanTa lên 1
            thoát vòng lặp
        nếu getCheck(Dong + iDem, Cot) bằng -1:
            tăng iSoQuanDich lên 1
        nếu getCheck(Dong + iDem, Cot) bằng 0:
            cho iDem2 chạy từ 2 đến 6 và Dong + iDem2 < _b->_size:
                nếu getCheck(Dong + iDem2, Cot) bằng 1:
                    tăng iSoQuanTa2 lên 1
                    thoát vòng lặp
                nếu getCheck(Dong + iDem2, Cot) bằng -1:
                    tăng iSoQuanDich2 lên 1
                nếu getCheck(Dong + iDem2, Cot) bằng 0:
```

```
                    thoát vòng lặp
            thoát vòng lặp

    // Duyệt theo chiều dọc (dòng) ngược lại
    cho iDem chạy từ 1 đến 5 và Dong - iDem >= 0:
        nếu getCheck(Dong - iDem, Cot) bằng 1:
            tăng iSoQuanTa lên 1
            thoát vòng lặp
        nếu getCheck(Dong - iDem, Cot) bằng -1:
            tăng iSoQuanDich lên 1
        nếu getCheck(Dong - iDem, Cot) bằng 0:
            cho iDem2 chạy từ 2 đến 6 và Dong - iDem2 >= 0:
                nếu getCheck(Dong - iDem2, Cot) bằng 1:
                    tăng iSoQuanTa2 lên 1
                    thoát vòng lặp
                nếu getCheck(Dong - iDem2, Cot) bằng -1:
                    tăng iSoQuanDich2 lên 1
                nếu getCheck(Dong - iDem2, Cot) bằng 0:
                    thoát vòng lặp
            thoát vòng lặp

    nếu iSoQuanTa bằng 2:
        trả về 0
    nếu iSoQuanTa bằng 0:
        iScoreTempDoc += Defend_Score[iSoQuanDich] * 2
    ngược lại:
        iScoreTempDoc += Defend_Score[iSoQuanDich]

    nếu iSoQuanDich >= iSoQuanDich2:
        iScoreTempDoc -= 1
    ngược lại:
        iScoreTempDoc -= 2

    nếu iSoQuanDich bằng 4:
        iScoreTempDoc *= 2

    trả về iScoreTempDoc
end function
```

### 4.1.13 Hàm SoDiemPhongThu_DuyetNgang

```
Hàm SoDiemPhongThu_DuyetNgang(Dong, Cot, Defend_Score[], Attack_Score[])
    Khai báo biến:
        iScoreTempNgang = 0
        iScoreDefend = 0
        iSoQuanDich = 0
        iSoQuanTa = 0
        iSoQuanDich2 = 0
        iSoQuanTa2 = 0

    Vòng lặp từ iDem = 1 đến 5 và Cot + iDem < _b->_size:
        Nếu getCheck(Dong, Cot + iDem) = 1 thì:
            Tăng iSoQuanTa lên 1
            Thoát vòng lặp

        Nếu getCheck(Dong, Cot + iDem) = -1 thì:
```

```
            Tăng iSoQuanDich lên 1

        Nếu getCheck(Dong, Cot + iDem) = 0 thì:
            Vòng lặp từ iDem2 = 2 đến 6 và Cot + iDem2 < _b->_size:
                Nếu getCheck(Dong, Cot + iDem2) = 1 thì:
                    Tăng iSoQuanTa2 lên 1
                    Thoát vòng lặp

                Nếu getCheck(Dong, Cot + iDem2) = -1 thì:
                    Tăng iSoQuanDich2 lên 1

                Nếu getCheck(Dong, Cot + iDem2) = 0 thì:
                    Thoát vòng lặp

            Thoát vòng lặp

    Vòng lặp từ iDem = 1 đến 5 và Cot - iDem >= 0:
        Nếu getCheck(Dong, Cot - iDem) = 1 thì:
            Tăng iSoQuanTa lên 1
            Thoát vòng lặp

        Nếu getCheck(Dong, Cot - iDem) = -1 thì:
            Tăng iSoQuanDich lên 1

        Nếu getCheck(Dong, Cot - iDem) = 0 thì:
            Vòng lặp từ iDem2 = 2 đến 6 và Cot - iDem2 >= 0:
                Nếu getCheck(Dong, Cot - iDem2) = 1 thì:
                    Tăng iSoQuanTa2 lên 1
                    Thoát vòng lặp

                Nếu getCheck(Dong, Cot - iDem2) = 0 thì:
                    Thoát vòng lặp

                Nếu getCheck(Dong, Cot - iDem2) = -1 thì:
                    Tăng iSoQuanDich2 lên 1

            Thoát vòng lặp

    Nếu iSoQuanTa = 2 thì:
        Trả về 0

    Nếu iSoQuanTa = 0 thì:
        iScoreTempNgang += Defend_Score[iSoQuanDich] * 2

    Ngược lại:
        iScoreTempNgang += Defend_Score[iSoQuanDich]

    Nếu iSoQuanDich >= iSoQuanDich2 thì:
        iScoreTempNgang -= 1

    Ngược lại:
        iScoreTempNgang -= 2

    Nếu iSoQuanDich = 4 thì:
        iScoreTempNgang *= 2

    Trả về iScoreTempNgang
```

### 4.1.14 Hàm SoDiemPhongThu_DuyetCheo1

```
Hàm SoDiemPhongThu_DuyetCheo1(Dong, Cot, Defend_Score[], Attack_Score[])
    Khai báo biến:
        iScoreTempCheoNguoc = 0
        iScoreDefend = 0
        iSoQuanDich = 0
        iSoQuanTa = 0
        iSoQuanDich2 = 0
        iSoQuanTa2 = 0

    Vòng lặp từ iDem = 1 đến 5 và Cot + iDem < _b->_size và Dong + iDem < _b-
>_size:
        Nếu getCheck(Dong + iDem, Cot + iDem) = 1 thì:
            Tăng iSoQuanTa lên 1
            Thoát vòng lặp

        Nếu getCheck(Dong + iDem, Cot + iDem) = 0 thì:
            Vòng lặp từ iDem2 = 2 đến 6 và Cot + iDem2 < _b->_size và Dong + iDem2
< _b->_size:
                Nếu getCheck(Dong + iDem2, Cot + iDem2) = 1 thì:
                    Tăng iSoQuanTa2 lên 1
                    Thoát vòng lặp

                Nếu getCheck(Dong + iDem2, Cot + iDem2) = 0 thì:
                    Thoát vòng lặp

                Nếu getCheck(Dong + iDem2, Cot + iDem2) = -1 thì:
                    Tăng iSoQuanDich2 lên 1

            Thoát vòng lặp

        Nếu getCheck(Dong + iDem, Cot + iDem) = -1 thì:
            Tăng iSoQuanDich lên 1

    Vòng lặp từ iDem = 1 đến 5 và Cot - iDem >= 0 và Dong - iDem >= 0:
        Nếu getCheck(Dong - iDem, Cot - iDem) = 1 thì:
            Tăng iSoQuanTa lên 1
            Thoát vòng lặp

        Nếu getCheck(Dong - iDem, Cot - iDem) = 0 thì:
            Vòng lặp từ iDem2 = 2 đến 6 và Cot - iDem2 >= 0 và Dong - iDem2 >= 0:
                Nếu getCheck(Dong - iDem2, Cot - iDem2) = 1 thì:
                    Tăng iSoQuanTa2 lên 1
                    Thoát vòng lặp

                Nếu getCheck(Dong - iDem2, Cot - iDem2) = 0 thì:
                    Thoát vòng lặp

                Nếu getCheck(Dong - iDem2, Cot - iDem2) = -1 thì:
                    Tăng iSoQuanDich2 lên 1

            Thoát vòng lặp

        Nếu getCheck(Dong - iDem, Cot - iDem) = -1 thì:
            Tăng iSoQuanDich lên 1

    Nếu iSoQuanTa = 2 thì:
        Trả về 0
```

```
    Nếu iSoQuanTa = 0 thì:
        iScoreTempCheoNguoc += Defend_Score[iSoQuanDich] * 2
    Ngược lại:
        iScoreTempCheoNguoc += Defend_Score[iSoQuanDich]

    Nếu iSoQuanDich >= iSoQuanDich2 thì:
        iScoreTempCheoNguoc -= 1
    Ngược lại:
        iScoreTempCheoNguoc -= 2

    Nếu iSoQuanDich = 4 thì:
        iScoreTempCheoNguoc *= 2

    Trả về iScoreTempCheoNguoc
```

### 4.1.15 Hàm SoDiemPhongThu_DuyetCheo2

```
Hàm SoDiemPhongThu_DuyetCheo2(Dong, Cot, Defend_Score[], Attack_Score[])
    Khai báo biến:
        iScoreTempCheoXuoi = 0
        iScoreDefend = 0
        iSoQuanDich = 0
        iSoQuanTa = 0
        iSoQuanDich2 = 0
        iSoQuanTa2 = 0

    Vòng lặp từ iDem = 1 đến 5 và Cot - iDem >= 0 và Dong + iDem < _b->_size:
        Nếu getCheck(Dong + iDem, Cot - iDem) = 1 thì:
            Tăng iSoQuanTa lên 1
            Thoát vòng lặp

        Nếu getCheck(Dong + iDem, Cot - iDem) = 0 thì:
            Vòng lặp từ iDem2 = 2 đến 6 và Cot - iDem2 >= 0 và Dong + iDem2 < _b-
>_size:
                Nếu getCheck(Dong + iDem2, Cot - iDem2) = 1 thì:
                    Tăng iSoQuanTa2 lên 1
                    Thoát vòng lặp

                Nếu getCheck(Dong + iDem2, Cot - iDem2) = 0 thì:
                    Thoát vòng lặp

                Nếu getCheck(Dong + iDem2, Cot - iDem2) = -1 thì:
                    Tăng iSoQuanDich2 lên 1

            Thoát vòng lặp

        Nếu getCheck(Dong + iDem, Cot - iDem) = -1 thì:
            Tăng iSoQuanDich lên 1

    Vòng lặp từ iDem = 1 đến 5 và Cot + iDem < _b->_size và Dong - iDem >= 0:
        Nếu getCheck(Dong - iDem, Cot + iDem) = 1 thì:
            Tăng iSoQuanTa lên 1
            Thoát vòng lặp

        Nếu getCheck(Dong - iDem, Cot + iDem) = 0 thì:
```

```
            Vòng lặp từ iDem2 = 2 đến 6 và Cot + iDem2 < _b->_size và Dong - iDem2
>= 0:
                Nếu getCheck(Dong - iDem2, Cot + iDem2) = 1 thì:
                    Tăng iSoQuanTa2 lên 1
                    Thoát vòng lặp

                Nếu getCheck(Dong - iDem2, Cot + iDem2) = 0 thì:
                    Thoát vòng lặp

                Nếu getCheck(Dong - iDem2, Cot + iDem2) = -1 thì:
                    Tăng iSoQuanDich2 lên 1

            Thoát vòng lặp

        Nếu getCheck(Dong - iDem, Cot + iDem) = -1 thì:
            Tăng iSoQuanDich lên 1

    Nếu iSoQuanTa = 2 thì:
        Trả về 0

    Nếu iSoQuanTa = 0 thì:
        iScoreTempCheoXuoi += Defend_Score[iSoQuanDich] * 2
    Ngược lại:
        iScoreTempCheoXuoi += Defend_Score[iSoQuanDich]

    Nếu iSoQuanDich >= iSoQuanDich2 thì:
        iScoreTempCheoXuoi -= 1
    Ngược lại:
        iScoreTempCheoXuoi -= 2

    Nếu iSoQuanDich = 4 thì:
        iScoreTempCheoXuoi *= 2

    Trả về iScoreTempCheoXuoi
```

### 4.1.16 Hàm Tim_Kiem_NuocDi_1()

```
Hàm Tim_Kiem_NuocDi_1()
    Khai báo biến:
        Oco là một cấu trúc _Point (bao gồm hai thuộc tính _x và _y)
        dong = 0, cot = 0
        Diem = 0

    Vòng lặp từ i = 0 đến _b->_size:
        Vòng lặp từ j = 0 đến _b->_size:
            Khởi tạo DiemTanCong = 0 và DiemPhongThu = 0

            Nếu getCheck(i, j) = 0 thì:
                Tính DiemTanCong bằng cách cộng các giá trị trả về từ các hàm
SoDiemTanCong_DuyetDoc, SoDiemTanCong_DuyetNgang, SoDiemTanCong_DuyetCheo1 và
SoDiemTanCong_DuyetCheo2 với tham số tương ứng

                Tính DiemPhongThu bằng cách cộng các giá trị trả về từ các hàm
SoDiemPhongThu_DuyetDoc, SoDiemPhongThu_DuyetNgang, SoDiemPhongThu_DuyetCheo1 và
SoDiemPhongThu_DuyetCheo2 với tham số tương ứng

                Nếu DiemTanCong > DiemPhongThu thì:
                    Nếu Diem < DiemTanCong thì:
```

```
                        Gán Diem = DiemTanCong
                        Gán dong = i
                        Gán cot = j

                Ngược lại:
                    Nếu Diem < DiemPhongThu thì:
                        Gán Diem = DiemPhongThu
                        Gán dong = i
                        Gán cot = j

    Gán Oco._x = cot * 4 + 2
    Gán Oco._y = dong * 2 + 1

    Trả về Oco
```

### 4.1.17 Hàm undoLastMove ()

```
Hàm undoLastMove():
    Nếu danh sách g->moves rỗng:
        Trả về

    Lấy x, y là tọa độ của nước đi cuối cùng trong g->moves

    Nếu giá trị tại ô (x, y) trên bàn cờ là -1:
        Giảm số lượng quân X (_b->CountX) đi 1
    Ngược lại nếu giá trị tại ô (x, y) trên bàn cờ là 1:
        Giảm số lượng quân Y (_b->CountY) đi 1

    Xóa nước đi cuối cùng khỏi danh sách g->moves

    Hoàn tác trạng thái của ô (x, y) trên bàn cờ bằng cách gọi hàm undo(x, y)
Kết thúc hàm
```

### 4.1.18 Hàm undoSetting(int check)// Undo theo chế độ

```
Hàm undoSetting(check):
    Nếu check bằng 1:
        Hoàn tác nước đi cuối cùng bằng cách gọi hàm undoLastMove()
        Di chuyển con trỏ đến vị trí (g->_x, g->_y) trên bàn cờ sử dụng hàm
gotoXY(g->_x, g->_y)
        Thay đổi lượt đi sử dụng hàm changeTurn()
    Ngược lại nếu check bằng 2:
        Hoàn tác 2 nước đi cuối cùng bằng cách gọi hai lần hàm undoLastMove()

Kết thúc hàm
```

### 4.1.19 Hàm goiY()

```
Hàm goiY():
    Lấy x là giá trị nhập từ bàn phím sử dụng hàm getXatEnter()
    Lấy y là giá trị nhập từ bàn phím sử dụng hàm getYatEnter()
    Lấy giá trị check tại ô (x, y) trên bàn cờ sử dụng hàm getCheckAtXY(x, y)

    Nếu check bằng 0:
        Thêm nước đi (x, y) vào danh sách g->moves sử dụng hàm addMove(x, y)
```

```
        Hoàn tác nước đi cuối cùng bằng cách gọi hàm undoLastMove() (xóa biểu
tượng tại vị trí (x, y))

    Gọi hàm TimKiemNuocDi()

    Lấy x là giá trị nhập từ bàn phím sử dụng hàm getXatEnter()
    Lấy y là giá trị nhập từ bàn phím sử dụng hàm getYatEnter()

    Di chuyển con trỏ đến vị trí (x, y) trên bàn cờ sử dụng hàm gotoXY(x, y)

    Đặt màu chữ thành màu xanh sử dụng hàm Textcolor(Green)

    Nếu lượt đi hiện tại là false:
        In ra ký tự "O"
    Ngược lại nếu lượt đi hiện tại là true:
        In ra ký tự "X"

    Lấy giá trị nhập từ bàn phím sử dụng hàm getConsoleInput() và lưu vào biến
input

    Nếu input bằng 6:
        Di chuyển con trỏ đến vị trí (x, y) trên bàn cờ sử dụng hàm gotoXY(x, y)
        In ra ký tự trống ""
        Xử lý trạng thái bàn cờ sử dụng hàm processCheckBoard()
        Thay đổi lượt đi sử dụng hàm changeTurn()

Kết thúc hàm
```

# C. TÀI LIỆU THAM KHẢO

1. Tài liệu hướng dẫn đồ án Caro được đăng tải trên hệ thống Moodle môn học
2. Kênh Youtube Thien Tam Nguyen, Danh sách kết hợp: [Lập trình đồ họa console C/C++] Hướng dẫn viết game dò mìn (7 thg 12, 2019), tại: https://www.youtube.com/watch?v=KAD7vo_n44k&list=RDCMUCSSMahX759Ppn7diWOpo7Yw&start_radio=1&t=1s
3. Hướng dẫn cách chạy file âm thanh trong lập trình C/C++: https://www.iostream.vn/article/chay-file-wav-voi-windows-h-Y4lmL
4. Vẽ bàn cờ caro: https://www.youtube.com/watch?v=AOvXmLpucXk
5. Video đồ án demo của các anh chị khóa trước trên nền tảng Youtube.