ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA: CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN GAME BĂNG QUA ĐƯỜNG TRÊN CONSOLE WINDOWS

Giảng viên hướng dẫn: Thầy Trương Toàn Thịnh

Nhóm thực hiện: 13

- NGUYỄN HẢI ĐĂNG – 20120049
- NGUYỄN NHẬT ĐĂNG – 20120050
- PHẠM DƯƠNG TRƯỜNG ĐỨC – 20120061
- QUÁCH BẢO QUÂN – 20120168

Lớp: 20CTT1

Niên khoá: 2020-2021

MỤC LỤC

A. GIỚI THIỆU

I. Lời dẫn:

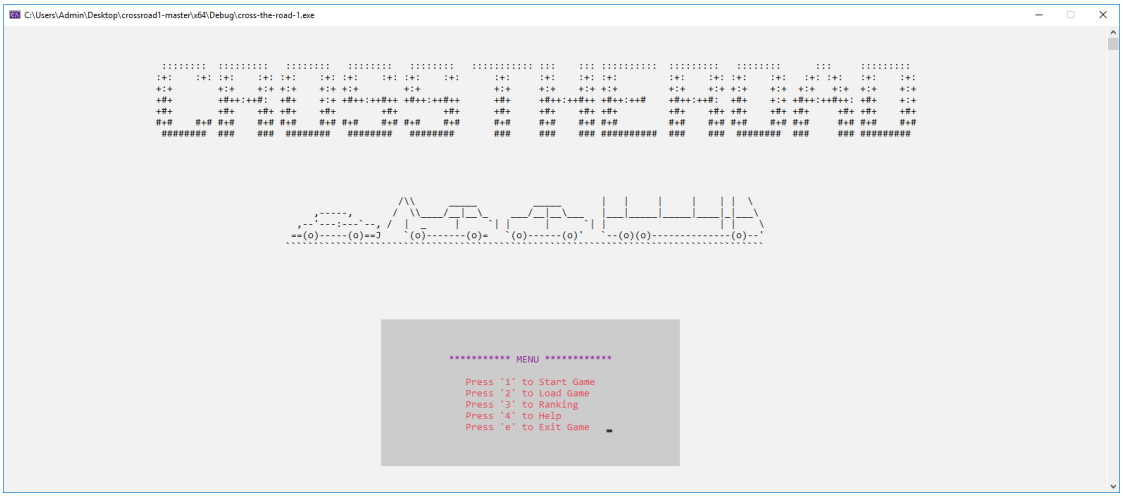
Đồ án game Cross The Road đối với mỗi cá nhân thành viên nhóm mình là một cơ hội rất tuyệt vời để mọi người làm quen cũng như cải thiện khả năng làm việc nhóm, mỗi thành viên đã cố gắng hết sức mình để hoàn thiện sản phẩm chung của nhóm.

II. Giới thiệu trò chơi:

Các chức năng chính:

- ✓ Bắt đầu New Game.
- ✓ Save/ Load (Lưu và Tải lại).
- ✓ Bảng xếp hạng điểm cao.
- ✓ Bảng hướng dẫn.
- ✓ Thoát game.

III. Xây dựng trò chơi:



B. CÁC NHÓM HÀM

I. Nhóm hàm DATA

//Là nơi chứa những hằng số, cấu trúc cần dùng xuyên suốt chương trình

```
1  #pragma once
2  // Kính thước màn hình console
3  #define WIDTH 1500 //1320
4  #define HEIGHT 700
5  #define X_CENTER WIDTH / 16 + 2
6  #define Y_CENTER HEIGHT / 32
7  //ASCII
8  #define ESC 27
9  #define KEY_UP 72
10 #define KEY_DOWN 80
11 #define KEY_LEFT 75
12 #define KEY_RIGHT 77
13
14 #include <iostream>
15 #include <Windows.h>
16 #include <stdlib.h>
17 #include <conio.h>
18 #include <string>
19 #include <fstream>
20 #include <vector>
21 #include <thread>
22 #include <iomanip>
23 #include <MMsystem.h>
24
25 #define SAVED_LIST "FileDaLuu.txt" //Ten cac file game duoc save
26 #define PLAYER_LIST "DSNguoiChoi.txt" //Thong tin cac player da choi game
27
28 using namespace std;
```

```
29
30  struct MENU
31  {
32      int choices; // Số chức năng của menu
33      int x; // hoành độ điểm bắt đầu
34      int y; // tung độ điểm bắt đầu
35  };
36  struct Cars
37  {
38      int n[5];
39      int x[5][3];
40      bool State[5];
41  };
42
43  struct you
44  {
45      int x, y;
46  };
47  #pragma comment (lib,"winmm.lib") // dùng để gọi âm thanh trong Game.
```

II. Nhóm hàm hiển thị màn hình Console

- **Hàm tạo cửa sổ Console:**

```
1   void CreateConsoleWindow(int pWidth, int pHeight)
2   {
3       HWND consoleWindow = GetConsoleWindow();
4
5       HANDLE hConsole;
6       hConsole = GetStdHandle(STD_OUTPUT_HANDLE);
7       SetConsoleTextAttribute(hConsole, 240);
8       GetWindowRect(consoleWindow, &r);
9       MoveWindow(consoleWindow, 0, 0, pWidth, pHeight, TRUE);
10  }
```

- GetConsoleWindow trả về con trỏ cửa sổ Window để thực hiện những thao tác tới Window. Hàm GetWindowRect để đưa dữ liệu về kích thước Window vào con trỏ r kiểu RECT. Hàm MoveWindow chỉ nơi bắt đầu vẽ cửa sổ, chiều rộng và cao của cửa sổ… Tạo biến HANDLE hConsole để dung biến này chỉnh sửa thuộc tính của các ký tự trên mành hình console qua lệnh SetConsoleTextAttribute(_MENU, int);

- **Hàm cố định cửa sổ Window:**

```
1  void FixConsoleWindow()
2  {
3      HWND consoleWindow = GetConsoleWindow();
4      LONG style = GetWindowLong(consoleWindow, GWL_STYLE);
5      style = style & ~(WS_MAXIMIZEBOX) & ~(WS_THICKFRAME);
6      SetWindowLong(consoleWindow, GWL_STYLE, style);
7  }
```

- GWL_STYLE được xem là dấu hiệu để hàm GetWindowLong lấy các đặc tính của cửa sổ Console. Hàm trả về một số kiểu long, ta có thể hiệu chỉnh lại tại dòng số 5. Hàm này để không cho người dùng tự thay đổi kích thước cửa sổ hiện hành.

III. Nhóm hàm vẽ và đồ hoạ

- **Hàm di chuyển trên Console:**

```
1  void GoTo(int x, int y)
2  {
3      COORD coord;
4      coord.X = x;
5      coord.Y = y;
6       SetConsoleCursorPosition(GetStdHandle
   (STD_OUTPUT_HANDLE), coord);
7  }
```

- **GoTo()**: di chuyển trên màn hình Console, với tọa độ là kiểu Coord và các thuộc tính (x là hoành độ, y là tung độ), sử dụng hàm SetConsoleCursorPosition() để đưa con trỏ trên Console đến tọa độ coord được truyền vào.

- **Các hàm liên quan tới đồ họa:**

```
1  void SetColor(int color) //Hàm setup màu nền và màu chữ
2  {
3      HANDLE hConsole;
4      hConsole = GetStdHandle(STD_OUTPUT_HANDLE);
5      SetConsoleTextAttribute(hConsole, color);
6  }
7
8  void Text(string name, int color, int x, int y)//In text
9  {
10     GoTo(x, y);
11     SetColor(color);
12     cout << name;
13 }
14
15 void Box(int color, int width, int height, int x, int y)//In
   hộp màu
16 {
17     SetColor(color);
18     for (int i = 0; i < height; i++)
19         {
20             GoTo(x, y + i);
21                 for (int j = 0; j < width; j++)
22             {
23                 cout << " ";
24             }
25          }
26     SetColor(240);
27 }
28
29 void BoxLoading(int color, int width, int height, int x, int
   y) //Hiệu ứng loading
30 {
31     int n = 0;
32     BigText("Logo.txt", 240, 27, 3);
33     BigText("LogoCar.txt", 240, 50, 15);
```

```
34      for (int i = 0; i < height; i++)
35      {
36          GoTo(x, y + i);
37          for (int j = 0; j < width; j++)
38          {
39              SetColor(color);
40              Sleep(30);
41              cout << " ";
42              n++;
43              Text(to_string(n) + "%", 240, x, y + 1);
44              GoTo(x + j, y);
45          }
46      }
47      SetColor(240);
48  }
49
50  void BigText(string filename, int color, int x, int y) //In dữ liệu tệp txt
51  {
52      fstream f;
53      f.open(filename, ios::in);
54
55      string line;
56      vector<string> subline;
57      while (!f.eof())
58      {
59          getline(f, line);
60          subline.push_back(line);
61      }
62      for (int i = 0; i < subline.size(); i++)
63          Text(subline[i], color, x, y++);
64      f.close();
65  }
66
```

```
67  void Clear(string& s, string t, int x, int y) //Xóa vùng
68  {
69      for (int i = 0; i < s.size() + t.size() + 2; i++)
70      {
71          Text(" ", 124, x + i, y);
72      }
73  }
74  void ClearScreen(int width, int height, int x, int y)
75  {
76       for (int i = 0; i < height; i++)
77       {
78            for (int j = 0; j < width; j++)
79                 Text(" ", 255, x + j, y + i);
80       }
81  }
```

- **SetColor():** Chỉnh màu cho các ký tự tiếp theo được output trên Console, HANDLE là con trỏ liên quan tới xử lý sự kiện trên Console Window, **GetStdHandle()** sẽ trả về con trỏ đó. Và hàm **SetConsoleTextAttribute()** sẽ thay đổi thuộc tính màu đó trên cửa sổ Console.
- **Text():** Dùng để in một xâu ký tự với màu và toạ độ in xâu ký tự, giúp cho việc in xâu ký tự trên màn hình Console với màu và những vị trí khác nhau trên màn hình Console được dễ dàng hơn.
- **BigText():** Dùng để in các hình cỡ lớn được vẽ bằng các ký tự ASCII lên màn hình Console. Bằng việc đọc file chứa các hình cỡ lớn đó, rồi đọc từng dòng của hình đó vào lưu từng dòng vào mỗi phần tử của một vector. Rồi in từng phần tử đó ra màn hình Console. Hàm này cũng hỗ trợ in màu và toạ độ in trên màn hình Console.
- **Box():** Dùng để in các hộp màu lên màn hình Console. Tham số truyền vào là màu hộp, kích thước hộp (chiều dài, chiều rộng), toạ độ in hộp (x, y). Bằng cách in các ký tự khoảng trắng " " có màu đã được định sẵn rồi dùng vòng lặp vẽ liên tục theo kích thước hộp,
- **BoxLoading():** Là hàm dùng để in hiệu ứng Loading cho game. Bằng cách sử dụng hàm Box để in thanh tiến trình (màu xanh), và liên tục thay đổi số phần trăm đã Loading.

- **Clear():** Hàm dùng để xoá các xâu ký tự với tham số truyền vào là 2 xâu ký tự và toạ độ nơi xoá. Hàm này sẽ biến các ký tự trong xâu thành ký tự khoảng trắng " " có màu tương ứng với màu của hộp màu **Box()** dưới các ký tự.
- **ClearScreen()**: Hàm dùng để xoá các "hộp màu" trong Game (ví dụ như hộp màu xám như hình bên dưới – mũi tên đang chỉ). Bản chất là biến các ký tự màu dùng để vẽ hộp thành ký tự khoảng trắng " " (màu trắng – trùng màu với màn hình Console).

IV. Nhóm hàm hiển thị MENU ban đầu.

- **Màn hình MENU ban đầu:**

```
1  MENU DisplayMenu()
2  {
3      MENU menu;
4      menu.choices = 3;
5      menu.x = X_CENTER - 15;
6      menu.y = Y_CENTER + 10;
7
8      system("cls");
9      PlaySound(TEXT("Menu.wav"), NULL, SND_FILENAME | SND_ASYNC | SND_LOOP);
10
11     BigText("Logo.txt", 240, 27, 3);
12     BigText("LogoCar.txt", 240, 50, 15);
13     Box(124, 53, 13, X_CENTER - 27, Y_CENTER + 5);
14     Text("**********  MENU  **********", 117, menu.x, menu.y - 2);
15     Text("    Press '1' to Start Game    ", 124, menu.x, menu.y);
```

```
16      Text("    Press '2' to Load Game     ", 124, menu.x, menu.y + 1);
17      Text("   Press '3' to Ranking    ", 124, menu.x, menu.y + 2);
18      Text("   Press '4' to Help      ", 124, menu.x, menu.y + 3);
19      Text("   Press 'e' to Exit Game   ", 124, menu.x, menu.y + 4);
20      SetColor(240);
21      return menu;
22  }
```

- **DisplayMenu():** Hàm in Logo và hiển thị màn hình chính của Game bao gồm 5 chức năng: New Game, Load Game, Show Rank, Help, Exit Game.

- **Từng chức năng của MENU:**

```
1   vector<string> FileSaved(string filename)
2   {
3       fstream f;
4       vector<string> File;
5       string line;
6       f.open(filename, ios::in);
7       while (!f.eof())
8       {
9           getline(f, line);
10          File.push_back(line);
11      }
12      return File;
13  }
14  void ShowFile()
15  {
16      MENU menu;
17      vector<string> File = FileSaved("FileDaLuu.txt");
18      File.resize(File.size() - 1);
19      string s;
20      menu.x = X_CENTER - 15;
21      menu.y = Y_CENTER + 10;
```

```
22
23      system("cls");
24
25      BigText("Logo.txt", 240, 27, 3);
26      BigText("LogoCar.txt", 240, 50, 15);
27      Box(124, 53, 13, X_CENTER - 27, Y_CENTER + 5);
28      Text("***********  FILES  ************", 117, menu.x,
menu.y - 3);
29      Text("Press ESC to Return Menu", 112, menu.x, menu.y +
6);
30      if (File.size() >= 1)
31          Text(File[File.size() - 1], 124, menu.x, menu.y -
1);
32      if (File.size() >= 2)
33          Text(File[File.size() - 2], 124, menu.x, menu.y);
34      if (File.size() >= 3)
35          Text(File[File.size() - 3], 124, menu.x, menu.y +
1);
36      InputFileName(s, menu.x, menu.y + 3, 0);
37      SetColor(240);
38  }
39  bool FileAvailable(string s)
40  {
41      vector<string> File = FileSaved("FileDaLuu.txt");
42      for (int i = 0; i < File.size(); i++)
43      {
44          if (s == File[i])
45          {
46              return 1;
47          }
48
49      }
50      return 0;
51  }
52  void InputFileName(string& s, int x, int y, int n)
```

```
53  {
54      MENU menu;
55      string s1 = "Nhap ten file: ", s2 = "Sai ten file!", s3 = "Nhap lai ten: ";
56      vector<string> File = FileSaved("FileDaLuu.txt");
57      Text(s1, 124, x, y);
58      s = InputName(n);
59      while (FileAvailable(s) == 0)
60      {
61          Clear(s, s1, x, y);
62          Text(s2, 124, x, y);
63          Clear(s, s3, x, y + 1);
64          Text(s3, 124, x, y + 1);
65          s.clear();
66          s = InputName(n);
67      }
68      Clear(s, s2, x, y);
69      Clear(s, s3, x, y + 1);
70      Text("Dung ten file! ", 124, x, y);
71      PlaySound(NULL, NULL, SND_FILENAME | SND_ASYNC | SND_LOOP);
72      LoadGame(s);
73  }
74  void Help() // Trợ giúp
75  {
76      Box(124, 75, 27, X_CENTER - 34, Y_CENTER - 15);
77      BigText("Help.txt", 124, X_CENTER - 15, Y_CENTER - 10);
78      char press2;
79      press2 = _getch();
80      while (press2 != 'r') press2 = _getch();
81      MenuControl();
82  }
83
84  void ReadPlayerInf(ifstream& f, player& x)
```

```
{
    getline(f, x.name);
    f >> x.score;
    f.ignore();
}
void Ranking()
{
    vector<player> plist;
    plist.resize(0);
    player tmp;
    ifstream f;
    f.open("DSNguoiChoi.txt", ios::in);
    while (!f.eof())
    {
        ReadPlayerInf(f, tmp);
        plist.push_back(tmp);
    }
    plist.resize(plist.size() - 1);
    for (int i = 0; i < plist.size(); i++)
    {
        for (int j = i + 1; j < plist.size(); j++)
        {
            if (plist[i].score < plist[j].score)
            {
                player m = plist[i];
                plist[i] = plist[j];
                plist[j] = m;
            }
        }
    }
    f.close();
    ofstream fb;
    fb.open("Rank.txt", ios::out);
    if (plist.size() <= 5)
```

```
119     {
120         for (int i = 0; i < plist.size(); i++)
121         {
122             fb << plist[i].name;
123             fb << setw(30 - plist[i].name.size());
124             fb << plist[i].score;
125             fb << endl;
126         }
127     }
128     else
129     {
130         for (int i = 0; i <= 4; i++)
131         {
132             fb << plist[i].name;
133             fb << setw(30 - plist[i].name.size());
134             fb << plist[i].score;
135             fb << endl;
136         }
137     }
138 }
```

- **FileSaved():** Lưu tên từng file vào từng phần tử của một vector bằng cách đọc file "FileDaLuu.txt" theo từng dòng.
- **ShowFile():** Hàm dùng để hiện tên 3 file gần nhất (nếu số file đã lưu ít hơn 3 thì hiện tên tất cả các file đã lưu). Hàm còn gọi hàm **FileSave()** và **InputFileName().**
- **FileAvailible():** Hàm dùng để kiểm tra tên file đã tồn tại trong vector **FileSaved** hay chưa.
- **InputFileName():** Hàm dùng để nhập tên file đã Load Game. Nếu file không tồn tại thì chương trình sẽ hiện thông báo người dùng nhập tên một file không tồn tại. Khi đó chương trình sẽ yêu cầu người chơi nhập lại. Người dùng có thể nhấn phím ESC để quay lại màn hình Menu.
- **Help():** Hàm dùng để hiển thị hướng dẫn chơi được ghi trong file "Help.txt".
- **ReadPlayerInf():** Hàm dùng để đọc tên người chơi với điểm số trong một file (cụ thể là file "DSNguoiChoi.txt" để xếp hạng top 5.
- **Ranking():** Hàm dùng để xếp hạng những lần chơi của người dùng. Đầu tiên là đọc dữ liệu trong file "DSNguoiChoi.txt" (dùng hàm **ReadPlayerInf()** và hàm **push_back**) để lưu dữ liệu từng phần tử vào vector (kiểu Player) rồi xếp hạng

người chơi theo số điểm (từ điểm cao xuống điểm thấp, dùng thuật toán **Bubble Sort**) và lưu bảng xếp hạng vào file "Rank.txt"

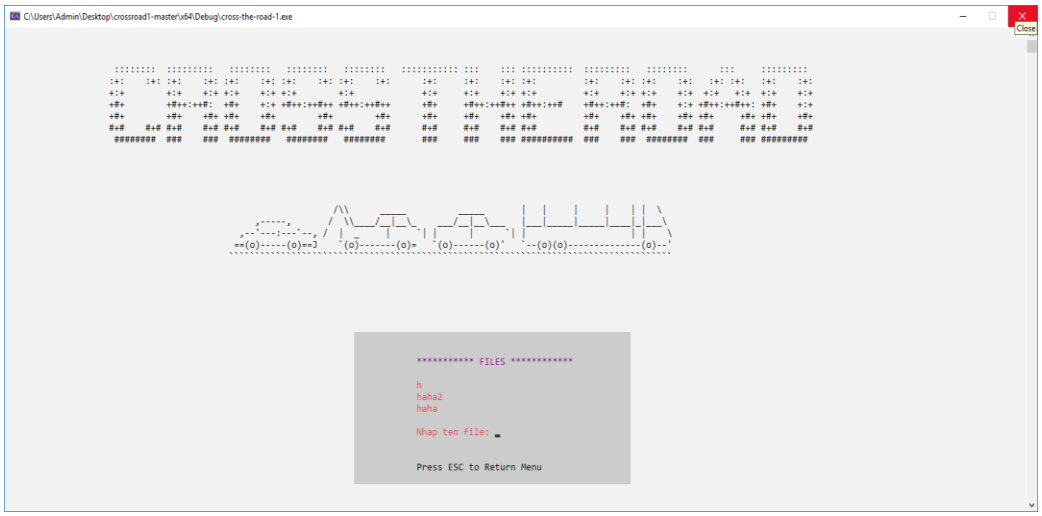
- **Hàm điều khiển chức năng MENU:**

```
//Là nơi gọi lại các hàm chức năng 1 cách khoa học và thẩm mỹ nhất:
```

```
1  void MenuControl()
2  {
3      SetColor(240);
4      DisplayMenu();
5      char press;
6      do { press = _getch(); }
7      while (press != '1' && press != '2' && press != '3' && press != '4' && press != 'e');
8      if (press == '1')
9      {
10         PlaySound(NULL, NULL, SND_FILENAME | SND_ASYNC | SND_LOOP);
11         int scr = 0;
12         system("cls");
13         InGame();
14     }
15     else if (press == '2')
16     {
17         ShowFile();
18     }
19     else if (press == '3')
20     {
21         system("cls");
22         Ranking();
23         Box(124, 75, 27, X_CENTER - 34, Y_CENTER - 15);
24         Text("******  TOP  5  BEST  PLAYERS  ******", 117, X_CENTER - 12, Y_CENTER - 15);
```

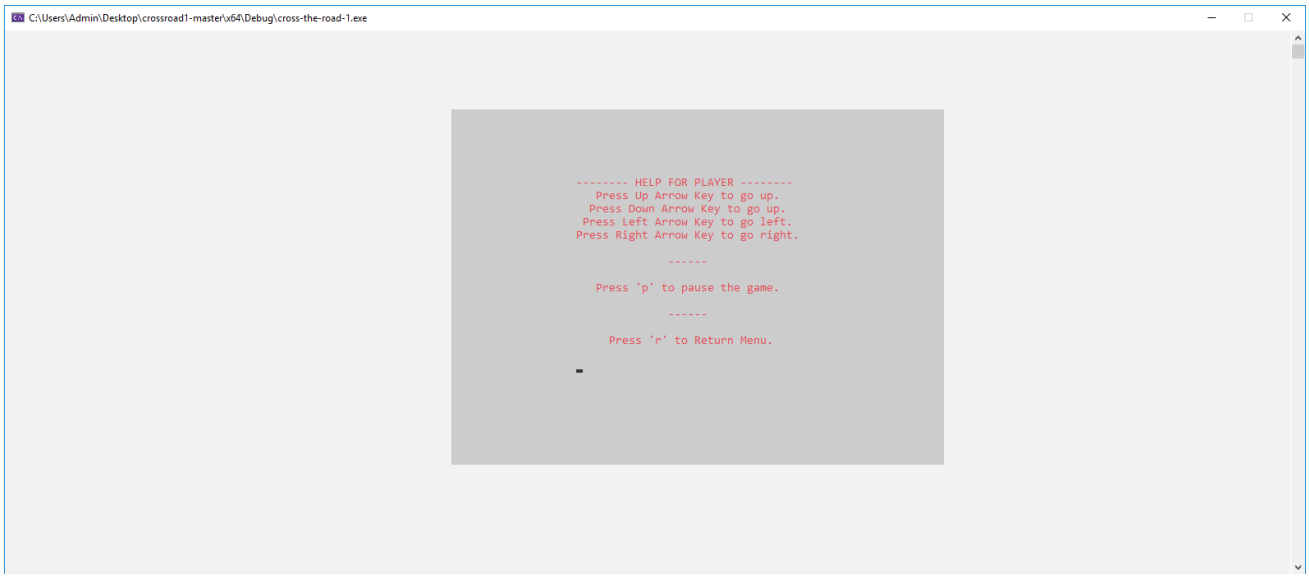
```
25          BigText("Rank.txt", 124, X_CENTER - 15, Y_CENTER - 10);
26          Text("Press 'r' to return to the menu screen.", 124, X_CENTER - 16, Y_CENTER - 3);
27          char press5;
28          press5 = _getch();
29          while (press5 != 'r') press5 = _getch();
30          SetColor(240);
31          MenuControl();
32      }
33      else if (press == '4')
34      {
35          system("cls");
36          Help();
37      }
38      else if (press == 'e')
39      {
40          PlaySound(NULL, NULL, SND_FILENAME | SND_ASYNC | SND_LOOP);
41          system("cls");
42          ExitGame();
43      }
44  }
```

- **Nhấn phím 1 để bắt đầu trò chơi với dữ liệu gốc.**
- **Nhấn phím 2 để Load Game từ một file đã lưu từ trước. Sau đó chương trình sẽ gọi hàm ShowFile() để hiện tên file. Có thể nhấn ESC để quay lại màn hình Menu.**

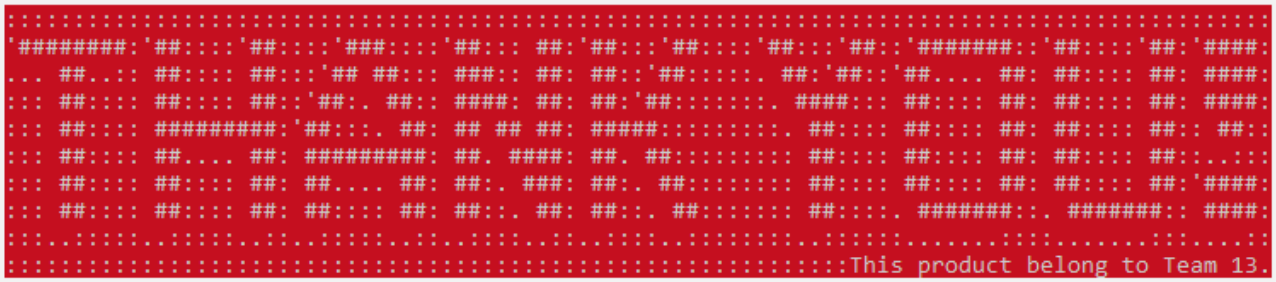


- **Nhấn phím 3 để hiện bảng xếp hạng top 5 người chơi có điểm cao nhất. Chương trình sẽ:**

  - Gọi hàm **Ranking()** để xếp hạng.
  - Gọi các hàm **Text(), BigText()** để hiển thị bảng xếp hạng, bài trí bảng xếp hạng top 5.
  - Có thể nhấn phím 'r' để quay lại màn hình Menu.
- **Nhấn phím 4 để hiện bảng hỗ trợ người chơi về các thao tác phím sử dụng trong trò chơi:**



- **Nhấn phím ‘ e ‘ để thoát game và đưa ra màn hình Exit bằng hàm ExitGame().**



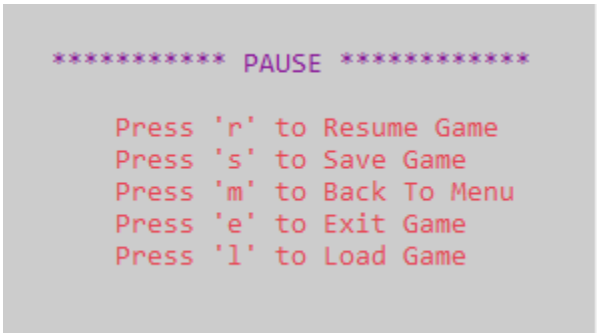
V. Hàm điều khiển trong game:

```
1  void ControlInGame(void)
2  {
3      thread t1(SubThread); //Khởi động thread.
4      while (1)
5      {
6          if (STT)
7          {
8              char press, press1;
9              press = _getch();
10             if (press == 'p')
11             {
12 
SuspendThread((HANDLE)t1.native_handle());
```

```
13                  STT = 0;
14                  PauseGame();
15                  do press1 = _getch();
16                  while (press1 != 'r' && press1 != 'e' &&
press1 != 'm' && press1 != 's' && press1 != 'l');
17                  if (press1 == 'r') {
18                      ClearScreen(50, 20, X_CENTER + 50,
Y_CENTER - 16);
19                      GoTo(0, 0);
20
ResumeThread((HANDLE)t1.native_handle());
21                      STT = 1;
22                  }
23                  else if (press1 == 'e') {
24                      ExitGame();
25                  }
26                  else if (press1 == 's') {
27
TerminateThread((HANDLE)t1.native_handle(), 0);
28                      SaveGame();
29                  }
30                  else if (press1 == 'm') {
31
TerminateThread((HANDLE)t1.native_handle(), 0);
32                      system("cls");
33                      MenuControl();
34                  }
35                  else if (press1 == 'l') {
36                      STT = 0;
37                      ClearScreen(50, 20, X_CENTER + 50,
Y_CENTER - 16);
38                      Box(124, 50, 20, X_CENTER + 50,
Y_CENTER - 16);
39                      Text("********** LOAD **********",
117, X_CENTER + 60, Y_CENTER - 13);
```

```
40                          InputFileName(s, X_CENTER + 55, Y_CENTER - 10, 1);
41                      }
42              else if (press == KEY_LEFT || press == KEY_RIGHT || press == KEY_DOWN || press == KEY_UP) {
43                      Moving = press;
44              }
45          }
46      }
47  }
```

- **ControlInGame(void)** là một hàm có tác dụng chính là quản lý các thao tác trong game:
    - Trường hợp các phím điều hướng (KEY_LEFT, KEY_RIGHT, KEY_DOWN, KEY_UP) được nhấn: biến di chuyển Moving sẽ được gán với biến press được _getch() từ bàn phím.
    - Trong trường hợp ‘ p ‘ được nhấn: một bảng điều khiển PAUSE được hiển thị với các chức năng khác.



        - Nhấn ‘ r ‘ : Game được tiếp tục.
        - Nhấn ‘ s ‘ : Tiến hành lưu trữ file game. Hàm SaveGame() được gọi.
        - Nhấn ‘ m ‘ : Trở về MENU chính.
        - Nhấn ‘ e ‘ : Thoát game.
        - Nhấn ‘ l ‘ : Tải lại một file game khác từ hệ thống.
        - Trường hợp các phím khác được nhấn thì _getch() sẽ được gọi lại bởi vòng lặp do while.

- Các đường hợp các phím khác được nhấn thì vòng lặp sẽ lập lại.

VI. Nhóm hàm dùng để điều khiển người dùng, xe ô tô và vẽ làn đường trong Game.

- **CÁC BIẾN TOÀN CỤC SỬ DỤNG TRONG GAME (ý nghĩa của các biến như comment).**

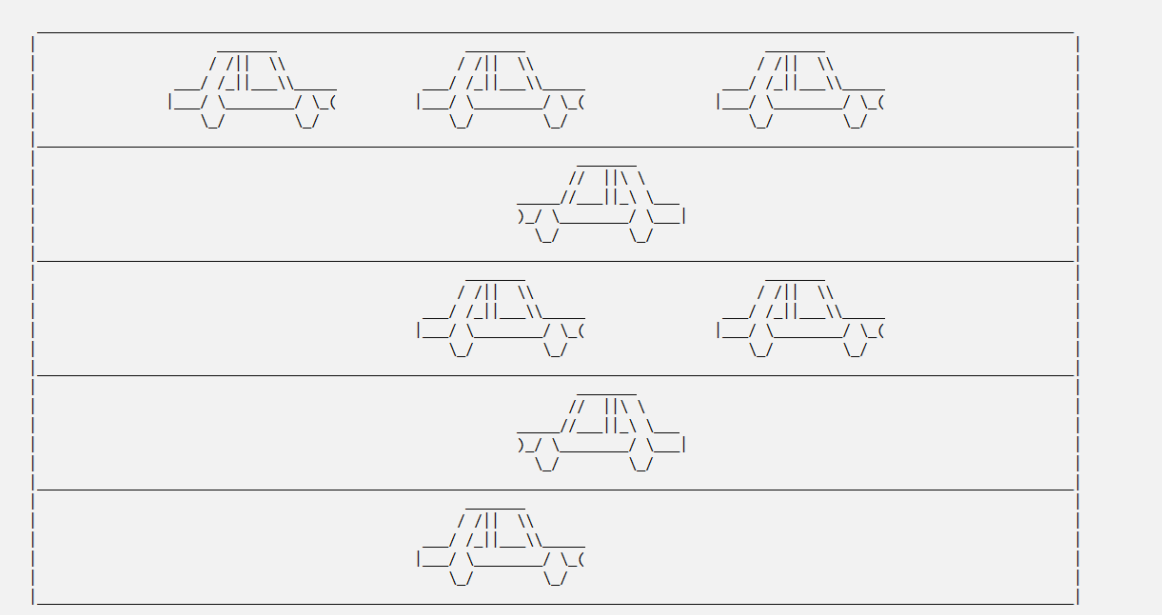
```
1  int timeStart[5], timeCur[5];
   //timeStrat[5]: dùng để random giá trị thời gian tìm làn đường. TimeCur[5]: lưu giá trị từng phần tử timeStrat[5] tương ứng
2  you Y; //Biến có kiểu cấu trúc You, dùng để lưu toạ độ người chơi.
3  Cars a; // Biến có kiểu cấu trúc Cars, dùng để lưu toạ độ xe.
4  int spd = 1; //Tốc độ di chuyển của xe
5  char Moving; //Biến nhận các phím điều khiển người chơi
6  bool STT; //Trạng thái của Thread (1 là cập nhật, 0 là dừng)
7  bool mark[130]; //Đánh dấu vị trí người chơi đã qua bên kia đường
8  int Score = 0; //Điểm số người chơi
9  int c = 50, d = 36; //Toạ độ người chơi ban đầu
```

- **HÀM VẼ CÁC LÀN ĐƯỜNG:**

```
1  void DrawBoard(int row, int col, int x, int y, int width, int height)
2  {
3      SetColor(240);
4      for (int i = 0; i < height; i++)
5      {
6          //Sleep(0.8);
7          //GoTo(x, y + i);
8          //cout << "|";
9          Text("|", 240, x, y + i);
10     }
11     for (int i = 0; i <= height; i += 6)
```

```
12      {
13          for (int j = 0; j <= width; j++)
14          {
15              //Sleep(0.8);
16              //GoTo(x + j + 1, y + i - 1);
17              //cout << "_";
18              Text("_", 240, x + j + 1, y + i - 1);
19          }
20      }
21      for (int i = 0; i < height; i++)
22      {
23          //Sleep(0.8);
24          //GoTo(x + width + 2, y + i);
25          //cout << "|";
26          Text("|", 240, x + width + 2, y + i);
27      }
28      GoTo(0, 0);
29  }
```

- **DrawBroad()**: Hàm này sẽ vẽ lần lượt đường viền bên ngoài phía tay trái -> các đường ngang chia làn đường -> đường viên bên ngoài phía tay phải.



- **CÁC HÀM LIÊN QUAN ĐẾN XE:**

```
1   void CreateCar()
2   {
3       srand(time(NULL));
4       for (int i = 0; i < 5; i++)
5       {
6           a.n[i] = (rand() % 3) + 1;
7           a.x[i][0] = (i % 2 == 0) ? 12 : 110;
8           for (int j = 1; j < a.n[i]; j++)
9           {
10              int temp = (i % 2 == 0) ? 35 : -35;
11              a.x[i][j] = a.x[i][j - 1] + temp;
12          }
13          a.State[i] = 1;
14          timeStart[i]  = (rand() % 50) + 50;
15          timeCur[i] = timeStart[i];
16      }
17  }
18
19  void StopCar()
20  {
21      for (int i = 0; i < 5; i++)
22      {
23          timeCur[i]--;
24          if (timeCur[i] < 0)
25          {
26              timeCur[i] = timeStart[i];
27              a.State[i] = !a.State[i];
28          }
29          GoTo(4, 5.5 + (i * 6));
30          cout << "   ";
31          SetColor(240);
32          GoTo(4, 5.5 + (i * 6));
33          cout << timeCur[i];
34      }
35
```

```
}
void DrawCar()
{
    for (int i = 0; i < 5; i++)
    {
        if(i%2==0)
        for (int j = 0; j < a.n[i]; j++)
        {
            BigText("Car.txt", 240, a.x[i][j], 5.5 + (i * 6));
        }
        else
        for (int j = 0; j < a.n[i]; j++)
        {
            BigText("ReCar.txt", 240, a.x[i][j], 5.5 + (i * 6));
        }
    }
}
void EraseCar()
{
    for (int i = 0; i < 5; i++)
    {
        for (int j = 0; j < a.n[i]; j++)
        {
            for (int k = 0; k < 5; k++)
            {
                GoTo(a.x[i][j], 5.5 + (i * 6) + k);
                SetColor(255);
                cout << "                    ";
                //Text("Empty.txt", 255, a.x[i][j], 5.5 + (i * 6) + k);
            }
        }
    }
```

```
68  }
69
70  void MoveCar()
71  {
72      for (int i = 0; i < 5; i+=2)
73      {
74          int cnt = 0;
75          do
76          {
77              cnt++;
78              if (a.State[i])
79              {
80                  for (int j = 0; j < a.n[i]; j++)
81                  {
82                      if (a.x[i][j] + 1 > 110)
83                          a.x[i][j] = 12;
84                      else a.x[i][j]++;
85                  }
86              }
87          } while (cnt < spd);
88      }
89      for (int i = 1; i < 5; i += 2)
90      {
91          int cnt = 0;
92          do
93          {
94              cnt++;
95              if (a.State[i])
96              {
97                  for (int j = 0; j < a.n[i]; j++)
98                  {
99                      if (a.x[i][j] - 1 < 12)
100                         a.x[i][j] = 110;
101                     else a.x[i][j]--;
102                 }
```

```
103                 }
104             } while (cnt < spd);
105         }
106 }
```

- **CreateCar()**: Hàm dùng để random số lượng xe mỗi làn đường, thời gian xe chạy và thời gian dừng xe. Sử dụng vòng lặp For, hàm này sẽ random số lượng xe mỗi làn (từ 1 đến 3 xe), nếu làn xe có vị trí chẵn (tính từ 0 và đếm từ trên xuống dưới màn hình), thì vị trí toạ độ x xe bắt đầu chạy là 12, ngược lại là 110. Sau đó tiếp tục lồng 1 vòng lặp For, nếu làn xe có vị trí chẵn thì xe sẽ di chuyển từ trái sang phải 1 đoạn 35 đơn vị, ngược lại là di chuyển từ phải sang trái 1 đoạn 35 đơn vị. Xe sẽ mặc định được chạy ngay từ đầu.
- **DrawCar()**: Hàm dùng để vẽ xe. Nếu làn xe có vị trí chẵn (tính từ 0 và đếm từ trên xuống dưới màn hình), thì hàm sẽ vẽ xe trong file "Car.txt", ngược lại sẽ vẽ xe trong file "ReCar.txt". Khoảng cách giữa 2 xe (nếu làn xe có nhiều xe) sẽ là 35 đơn vị.
- **EraseCar()**: Hàm dùng để xoá xe. Bản chất của hàm sẽ biến các ký tự vẽ trở thành các dấu khoảng trắng " " có màu cùng với màu nền chính của Game.
- **MoveCar()**: Hàm gồm 2 vòng lặp dùng để di chuyển xe. Trong đó: Biến **cnt** để giới hạn cho mỗi lần xe chạy sẽ không chạy nhiều hơn spd hiện tại.
- **CÁC HÀM LIÊN QUAN ĐẾN NGƯỜI CHƠI:**

```
1  void MoveUp()
2  {
3      if (Y.y - 6 > 0)
4      {
5          PlaySound(TEXT("Move.wav"), NULL, SND_FILENAME | SND_ASYNC);
6          ErasePerson();
7          Y.y -= 6;
8          BigText("Person.txt", 240, Y.x, Y.y);
9      }
10     else if (Y.y - 6 == 0)
11     {
12         if (mark[Y.x] == 0)// && mark[Y.x + 1] == 0 && mark[Y.x + 2] == 0)
13         {
14             ErasePerson();
15             Y.y -= 6;
16             BigText("Person.txt", 240, Y.x, Y.y);
```

```
17                  mark[Y.x] = 1;
18                  //mark[Y.x + 1] = 1;
19                  //mark[Y.x + 2] = 1;
20              }
21              else
22              {
23                  STT = 0;
24                  char press4 = ' ';
25                  MENU menu;
26                  menu.x = X_CENTER + 52;
27                  menu.y = Y_CENTER - 11;
28                  Box(124, 40,20, X_CENTER + 50, Y_CENTER - 16);
29                  Text("********** PAUSE **********", 117, menu.x, menu.y - 2);
30                  Text(" Va vao nguoi khac se bi tru 50 diem ", 124, menu.x, menu.y);
31                  Text("    An 'q' de dong y    ", 124, menu.x, menu.y + 1);
32                  Text("    An 's' de tu choi    ", 124, menu.x, menu.y + 2);
33                  do press4 = _getch();
34                  while (press4 != 'q' && press4 != 's');
35                  if (press4 == 'q')
36                  {
37                      ClearScreen(40, 20, X_CENTER + 50, Y_CENTER - 16);
38                      ErasePerson();
39                      Y.y -= 6;
40                      BigText("Person.txt", 240, Y.x, Y.y);
41                      GoTo(0, 0);
42                      mark[Y.x] = 1;
43                      //mark[Y.x + 1] = 1;
44                      //mark[Y.x + 2] = 1;
45                      Score -= 50;
46                      STT = 1;
47                  }
48                  else if(press4 =='s')
```

```
49                  {
50                      ClearScreen(40, 20, X_CENTER + 50,
Y_CENTER - 16);
51                      GoTo(0, 0);
52                      STT = 1;
53                  }
54              }
55
56          }
57      }
58      void MoveDown()
59      {
60          if (Y.y + 6 <= 36)
61          {
62              PlaySound(TEXT("Move.wav"), NULL, SND_FILENAME |
SND_ASYNC);
63              ErasePerson();
64              Y.y += 6;
65              BigText("Person.txt", 240, Y.x, Y.y);
66          }
67      }
68      void MoveRight()
69      {
70          if (Y.x + 3 < 130)
71          {
72              ErasePerson();
73              Y.x += 3;
74              BigText("Person.txt", 240, Y.x, Y.y);
75          }
76      }
77      void MoveLeft()
78      {
79          if (Y.x - 3 > 10)
80          {
81              ErasePerson();
82              Y.x -= 3;
83              BigText("Person.txt", 240, Y.x, Y.y);
84          }
```

```
85  }
86
87  bool Impact()
88  {
89
90      int i = 4;
91      while (5.5 + (i * 6) > Y.y) i--;
92      for (int j = 0; j < a.n[i]; j++)
93      {
94          if (a.x[i][j] <= Y.x + 3 && a.x[i][j] + 20 >= Y.x && Y.y >= (5.5 + i * 6) && Y.y < (5.5 + (i + 1) * 6))
95          {
96              return true; // kiem tra nguoi co nam trong doan duong i hay khong va co o vi tri da co xe san hay khong
97          }
98      }
99      return false;
100 }
101 void Finish()
102 {
103     if (spd == 3) spd = 1;
104     else spd++;
105     //PlaySound(TEXT("Up.wav"), NULL, SND_FILENAME | SND_SYNC);
106     Score += 100;
107     Y.x = 50;
108     Y.y = 36;
109     Moving = KEY_RIGHT;
110 }
111 void SubThread(void)
112 {
113     while (1)
114     {
115         if (STT)
116         {
117             switch (Moving)
118             {
```

```
119                 case KEY_LEFT:
120                     MoveLeft(); break;
121                 case KEY_RIGHT:
122                     MoveRight(); break;
123                 case KEY_DOWN:
124                     MoveDown(); break;
125                 case KEY_UP:
126                     MoveUp(); break;
127                 }
128                 Moving = ' ';
129                 StopCar();
130                 EraseCar();
131                 MoveCar();
132                 DrawCar();
133                 if (Impact())
134                 {
135                     YDead();
136                     return;
137                 }
138                 if (Y.y < 5.5)
139                 {
140                     Finish();
141                 }
142                 MENU menu;
143                 menu.x = X_CENTER + 52;
144                 menu.y = Y_CENTER - 11;
145                 Text("Score: ", 240, menu.x, menu.y);
146                 cout << Score;
147                 SetColor(240);
148                 Sleep(150);
149             }
150     }
151
152 }
153 void ErasePerson()
154 {
155     for (int i = 0; i < 4; i++)
156     {
```

```
157             GoTo(Y.x, Y.y + i);
158             cout << "   ";
159         }
160 }
```

- **MoveUp()**: Hàm dùng để di chuyển người dùng lên trên. Nếu Y.y – 6 > 0 thì người sẽ di chuyển lên trên 1 đoạn 6 đơn vị, còn nếu Y.y – 6 = 0 thì sẽ có 2 trường hợp xảy ra
    - Nếu người đụng vào một người khác ở bên kia làn đường thì Game sẽ dừng lại, chương trình sẽ hiển thị 1 bảng thông báo: Nếu nhấn "q" thì người chơi sẽ chỉ được cộng 50 điểm (bình thường là được cộng 100 điểm) và Game được tiếp tục, 1 người mới sẽ được tạo ra; còn nếu nhấn "s" thì người vẫn sẽ ở yên vị trí cũ.
    - Còn ngược lại thì vị trí của người đã qua hết làn đường sẽ được lưu lại (biến Mark[Y.x] = 1).
- **MoveDown()**: Hàm dùng để di chuyển người dùng xuống dưới. Nếu Y.y + 6 <= 36 thì người sẽ di chuyển xuống dưới 1 đoạn 6 đơn vị, ngược lại thì người dùng sẽ đứng yên.
- **MoveLeft()**: Hàm dùng để di chuyển người dùng sang bên trái. Y.x – 3 > 10 thì người sẽ di chuyển sang bên trái 1 đoạn 3 đơn vị, ngược lại thì người dùng sẽ đứng yên.
- **MoveRight()**: Hàm dùng để di chuyển người dùng sang bên trái. Y.x + 3 < 130 thì người sẽ di chuyển sang bên phải 1 đoạn 3 đơn vị, ngược lại thì người dùng sẽ đứng yên.
- **Impact()**: Hàm trả về kiểu Bool, dùng để kiểm tra người dùng có đụng các xe chưa (nếu a.x[i][j] <= Y.x + 3 && a.x[i][j] + 20 >= Y.x && Y.y >= (5.5 + i * 6) && Y.y < (5.5 + (i + 1) * 6)). Nếu đụng thì hàm sẽ trả về TRUE (1), ngược lại là FALSE (0).
- **Finish()**: Hàm dùng để cập nhật dữ liệu và tạo 1 người mới sau khi 1 người qua được bên kia làn đường. Khi có 1 người qua hết tất cả các làn đường thì điểm số sẽ được tăng lên 100, và 1 người mới sẽ được tạo với toạ độ Y.x = 50, Y.y = 36.
- **YDead()**: Hàm dùng để thông báo và lưu lại một số dữ liệu sau khi Game Over. Sau khi Game Over thì sẽ có hiệu ứng và âm thanh xe cứu thương xuất hiện -> Chương trình yêu cầu nhập tên để lưu dữ liệu điểm số để xếp hạng cho các người chơi -> Nhấn phím 'r' để quay lại màn hình Menu sau khi nhập tên. Hàm còn reset lại dữ liệu gốc để người dùng tiếp tục chơi.
- **SubThread()**: Đây là kịch bản chính trong Game. Hàm sử dụng vòng lặp While, liên tục nhận các phím bấm của người dùng và liên tục thực hiện các hàm đã nêu ở trên nếu biến STT = 1. Nếu nhấn phím "▲" thì hàm **MoveUp()** được được gọi, phím "►" thì hàm **MoveRight()** được được gọi**,** phím **"▼"** thì hàm **MoveDown()**

được gọi, phím "◀" thì hàm **MoveLeft()** được gọi. Nếu **Impact() = 1** thì hàm **YDead()** được gọi và dừng việc thực hiện hàm. Ngược lại thì hàm **Finish()** được gọi. Vòng lặp While sẽ được thực hiện mỗi 0.15 giây.

- **ErasePerson()**: Hàm dùng để xoá vị trí của người khi di chuyển sang một vị trí khác. Bản chất là biến cách ký tự vẽ nên người thành dấu khoảng trắng " ".

VII. Nhóm hàm liên quan đến dữ liệu của Game.

- **HÀM THIẾT LẬP DỮ LIỆU BAN ĐẦU CHO GAME:**

```
1  void InGame()
2  {
3      system("cls");
4      //SetColor(240);
5      Score = 0;
6      for (int i = 0; i < 130; i++)
7      {
8          mark[i] = 0;
9      }
10     CreateCar();
11     DrawBoard(0, 0, 10, 5.5, 120, 30);
12     Y.x = c; Y.y = d;
13     BigText("Person.txt", 240, Y.x, Y.y);
14     Moving = 'd';
15     spd = 1;
16     STT = 1;
17     DrawCar();
18     ControlInGame();
19 }
```

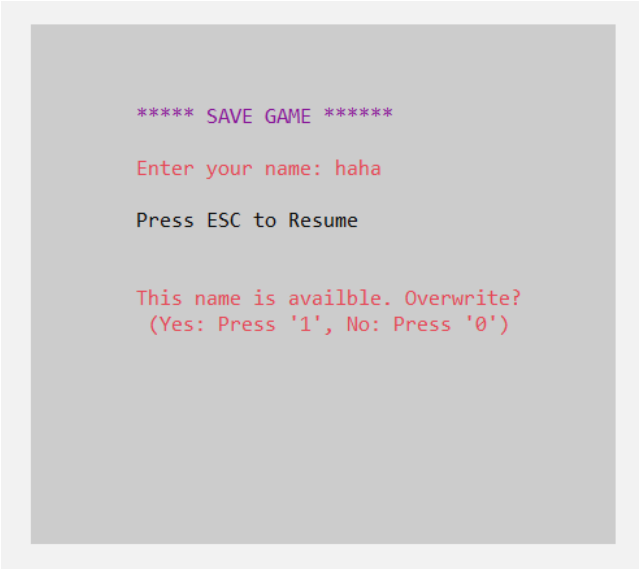
- **HÀM DÙNG ĐỂ LƯU GAME:**

```
1  void SaveGame()
2  {
3      string s;
4      string s4 = "Press ESC to Resume";
5      string s3 = "This name is availble. Overwrite?";
6      string s2 = " (Yes: Press '1', No: Press '0') ";
7      MENU menu;
8      menu.x = X_CENTER + 59;    //X_CENTER + 52
9      menu.y = Y_CENTER - 11;     /*Y_CENTER - 11;*/
10     //Box(124, 10, 10, 56, 16);
```

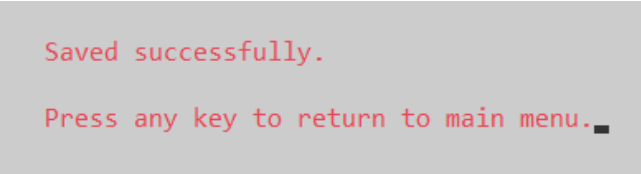
```
11      Box(124, 50, 20, X_CENTER + 50, Y_CENTER - 16); //+50 ,-16
12      Text("***** SAVE GAME ******", 117, menu.x, menu.y - 2);
13      Text(s4, 112, menu.x, menu.y + 2);
14      Text("Enter your name: ", 124, menu.x, menu.y);
15      do {
16          s = InputName(1);
17          if (FileAvailable(s))
18          {
19              Text(s3, 124, menu.x, menu.y + 5);
20              Text(s2, 124, menu.x, menu.y + 6);
21              char press7;
22              do  press7 = _getch(); while (press7 != '1' && press7 != '0');
23              if (press7 == '1') break;
24              else
25              {
26                  Clear(s3, "", menu.x, menu.y + 5);
27                  Clear(s2, "", menu.x, menu.y + 6);
28                  Clear(s, "", menu.x + 17, menu.y);
29                  s.clear();
30                  GoTo(menu.x + 17, menu.y);
31                  //s = InputName(1);
32              }
33          }
34          else
35          {
36              fstream f;
37              f.open("FileDaLuu.txt", ios::app);
38              f << s;
39              f << endl;
40              f.close();
41              break;
42          }
43      } while (FileAvailable(s));
44      string s1 = s + ".txt";
45      ofstream f1;
```

```
46      f1.open(s1, ios::out);
47      f1 << Score << " " << spd << " " << Y.x << " " << Y.y;
48      f1 << endl;
49      for (int i = 0; i < 5; i++)
50      {
51          f1 << a.n[i] << " ";
52          for (int j = 0; j < a.n[i]; j++)
53          {
54              f1 << a.x[i][j] << " ";
55          }
56          f1 << timeStart[i] << " " << timeCur[i] << " " << a.State[i] << endl;
57      }
58      for (int i = 0; i < 130; i++)
59      {
60          if (mark[i] == 1)
61          {
62              f1 << i << " ";
63          }
64      }
65      f1.close();
66      Clear(s4, "", menu.x, menu.y + 2);
67      Text("Saved successfully.", 124, menu.x, menu.y + 4);
68      Clear(s3, "", menu.x, menu.y + 5);
69      Text("Press any key to return to main menu.",124, menu.x, menu.y + 6);
70      _getch();
71      MenuControl();
72  }
```

- Hàm sẽ yêu cầu người dùng nhập tên File để lưu trước. Nếu tên file đã tồn tại thì hàm sẽ hỏi người dùng có lưu đè dữ liệu lên không. Nếu có thì nhấm phím '1', chương trình sẽ lưu đè dữ liệu lên file đó, nếu nhấn không thì nhấn phím "0", chương trình sẽ yêu cầu người dùng nhập lại tên file.



- Khi lưu file thành công sẽ có thông báo như hình bên dưới xuất hiện.



- Khi lưu file thành công, tên file lưu sẽ là chuỗi s người dùng nhập + đuôi ".txt".
- Cấu trúc tệp tin lưu file:
  - **Dòng thứ 1:** lưu lần lượt điểm, giá trị biến **spd** (tốc độ di chuyển của xe), toạ độ người hiện tại (Y.x và Y.y). Các giá trị cách nhau bởi dấu khoảng trắng " ".
  - **Dòng thứ 2, 3, 4, 5, 6:** lưu lần lượt số xe trong làn, vị trí (trục x) của các xe, giá trị timeStrat, timeCur của từng làn xe tại thời điểm dừng, tình trạng của xe (đang đứng yên hay di chuyển). Các giá trị cách nhau bởi dấu khoảng trắng " ".
  - **Dòng thứ 7 (nếu có)**: vị trí của những người đã qua được hết các làn đường, và ở bên kia làn đường. Các giá trị cách nhau bởi dấu khoảng trắng " ".
- **HÀM DÙNG ĐỂ LOAD GAME TỪ MỘT TỆP TIN ĐÃ LƯU TỪ TRƯỚC:**

```
1  void LoadGame(string s)
2  {
3      for (int i = 0; i < 130; i++)
4      {
5          mark[i] = 0;
6      }
7      ifstream fb;
```

```
8       fb.open(s + ".txt");
9       fb >> Score >> spd >> c >> d;
10      SetColor(240);
11      system("cls");
12      for (int i = 0; i < 5; i++)
13      {
14          fb >> a.n[i];
15          for (int j = 0; j < a.n[i]; j++)
16          {
17              fb >> a.x[i][j];
18          }
19          fb >> timeStart[i] >> timeCur[i] >> a.State[i];
20      }
21      if (Score != 0)
22      while (!fb.eof())
23      {
24          int i;
25          fb >> i;
26          {
27              BigText("Person.txt", 240, i, 0);
28          }
29          mark[i] = 1;
30      }
31      fb.close();
32      DrawBoard(0, 0, 10, 5.5, 120, 30);
33      Y.x = c; Y.y = d;
34      BigText("Person.txt", 240, Y.x, Y.y);
35      Moving = 'd';
36      STT = 1;
37      DrawCar();
38      ControlInGame();
39  }
```

- Đầu tiên, hàm này sẽ reset lại mảng Mark về giá trị 0 (để xoá vị trí người dùng đã lưu trước đó ở game cũ).
- Sau đó, hàm sẽ đọc dữ liệu các dòng thứ 1, 2, 3, 4, 5, 6 được lưu trong tệp tin (cấu trúc tệp tin đã được đề cập ở trên).
- Nếu điểm số (Score) khác 0 thì tiếp tục đọc dữ liệu dòng thứ 7 trong cấu trúc tệp tin cho đến khi con trỏ đã trỏ xuống vị trí cuối cùng của tệp tin.

- Các biến toàn cục sẽ có giá trị như các đoạn code trên, và cuối cùng là gọi hàm **DrawCar()** và **ControlInGame()**.

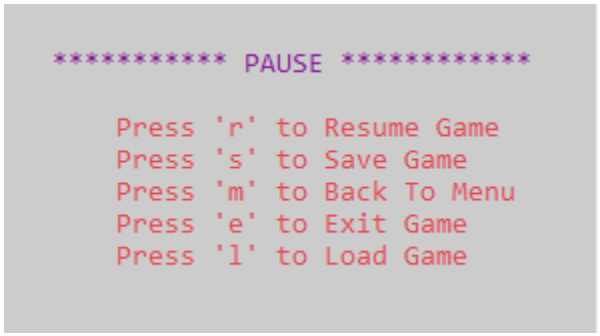
VIII. Các hàm phụ trợ khác

```
1  void PauseGame()
2  {
3      MENU menu;
4      menu.x = X_CENTER + 59; //X_CENTER + 52;
5      menu.y = Y_CENTER - 11; //Y_CENTER - 11;
6      //DrawBoard(2, 1, 54, 15, 45, 10);
7      Box(124, 50, 20, X_CENTER + 50, Y_CENTER - 16);
8      Text("********** PAUSE **********", 117, menu.x,
menu.y - 2);
9      Text("    Press 'r' to Resume Game  ", 124, menu.x,
menu.y);
10     Text("    Press 's' to Save Game    ", 124, menu.x,
menu.y + 1);
11     Text("    Press 'e' to Exit Game    ", 124, menu.x,
menu.y + 3);
12     Text("    Press 'm' to Back To Menu ", 124, menu.x,
menu.y + 2);
13     Text("    Press 'l' to Load Game ", 124, menu.x,
menu.y + 4);
14     SetColor(255);
15 }
16 void ExitGame()
17 {
18     system("cls");
19     SetColor(255);
20     BigText("ThankYou.txt", 71, 45, 15);
21 }
22 void Ambulance()
23 {
24     for (int i = 0; i <= 190; i++)
25     {
26         BigText("Ambulance.txt", 252, i, 15);
27         Sleep(20);
28         for (int j = 0; j <= 5; j++)
29         {
```

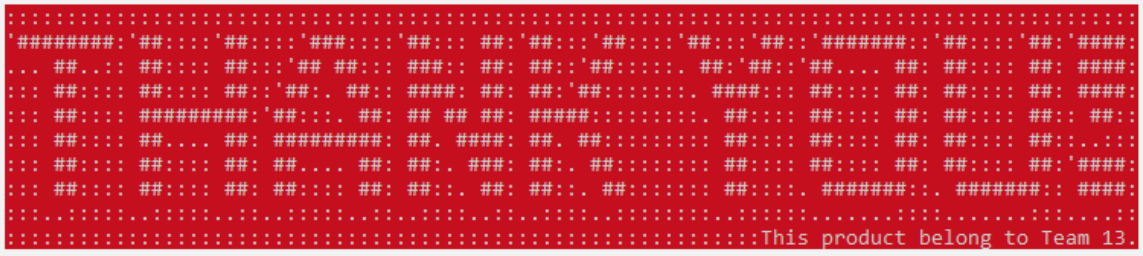
```
30                  Text("                                        ",
255, i, 15 + j);
31              }
32          }
33      }
34
35      string InputName(int n)
36      {
37          string Name;
38          for (char c; (c = _getch()); )
39          {
40              if (c == '\n' || c == '\r') { //Enter
41                  cout << "\n";
42                  break;
43              }
44              else if (c == '\b' && Name.size() > 0) {
//backspace
45                  cout << "\b \b";
46                  if (!Name.empty())
47                      Name.erase(Name.size() - 1);
48              }
49              else if (c == -32) { //Up,Down,Right,Left
50                  _getch();
51              }
52              else if (isprint(c)) {
53                  cout << c;
54                  Name += c;
55              }
56              else if (c == 27 && !isprint(c) && n == 0)
57                  MenuControl();
58              else if (c == 27 && !isprint(c) && n == 1)
59              {
60                  ClearScreen(50, 20, X_CENTER + 50, Y_CENTER
- 16);
61                  GoTo(0, 0);
62                  //ResumeThread((HANDLE)t1.native_handle());
63                  STT = 1;
64                  ControlInGame();
```

```
65              }
66          }
67          return Name;
68      }
```

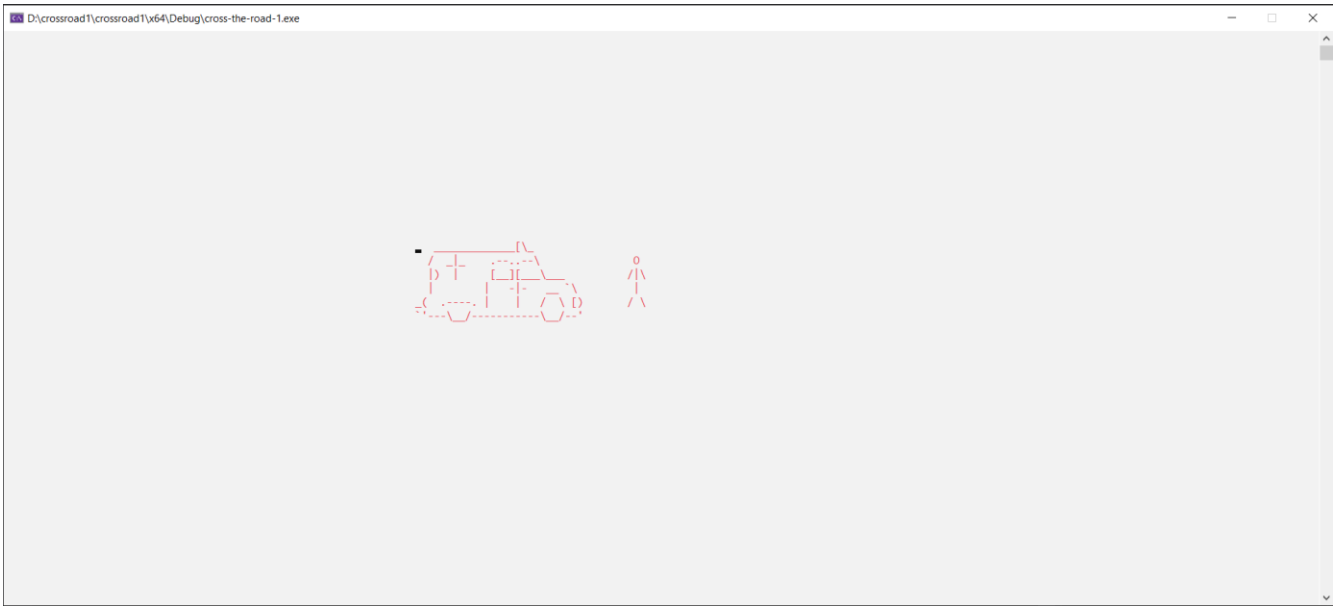
- **PauseGame()**: Hàm hiển thị phần màn hình khi người dùng tạm dừng Game.



- **ExitGame()**: Hàm hiển thị màn hình khi người dùng thoát Game.



- **Ambulance()**: Hàm dùng để tạo hiệu ứng xe cứu thương khi người chơi thua cuộc.



- **InputName()**: Hàm dùng để nhập một xâu ký tự từ bàn phím, nhưng có điểm khác biệt so với cách nhập xâu ký tự thông thường là có tham số n: là chức năng của phím ESC trong 2 trường hợp n = 0 và n = 1
    - n = 0: nhập 1 xâu ở màn hình Menu (Load Game từ màn hình Menu), khi đó nếu nhấn ESC thì hàm **MenuControl()** sẽ được gọi.
    - n = 1: nhập 1 xâu trong Game (Save và Load Game ngay khi Game diễn ra), khi đó nếu nhấn ESC thì Game sẽ được tiếp tục.

IX. Hàm main()

```
1  int main()
2  {
3      //int cnt = 0;//Biến hỗ trợ trong quá trình tăng tốc độ xe di chuyển
4  //int MOVING;//Biến xác định hướng di chuyển của người
5      //int SPEED;// Tốc độ xe chạy (xem như level)
6      //bool STATE; // Trạng thái sống/chết của người qua đường
7      CreateConsoleWindow(WIDTH, HEIGHT);
8      FixConsoleWindow();
9      BoxLoading(32, 100, 1, 43, 30);
10     Sleep(1000);
11     MenuControl();
12     return 0;
13 }
```

- Đây là nơi viết nên tiến trình chính của Game. Bắt đầu từ cố định màn hình Console, kích thước màn hình, sau đó là gọi hàm **BoxLoading()** và cuối cùng là gọi hàm **MenuControl()** để bắt đầu một tiến trình Game.

C. LỜI KẾT

- Mặc dù đã cố gắng hết sức, các thành viên đã rất cố gắng để hoàn thành tốt nhiệm vụ của mình được giao nhưng chắc chắn không thể tránh khỏi những thiếu sót như code chưa tối ưu, trình bày chưa đẹp, … Nhưng chúng em đã cố gắng hết sức để hoàn thành tốt nhất có thể đồ án này.
- Chúng em xin cảm ơn thầy Trương Toàn Thịnh đã đưa ra những hướng dẫn, hướng đi cụ thể để chúng em có thể hoàn thành tốt đồ án này. Chúng em xin cảm ơn thầy!
- Nguồn tài liệu tham khảo:
    - Đồ án cờ caro sau: https://github.com/dungxibo123/caro
    - Stack Overflow.
    - Hướng dẫn chay âm thanh trong C++: https://www.stdio.vn/article/chay-file-wav-voi-windows-h-Y4lmL
    - Tài liệu về hướng dẫn đồ án của thầy Trương Toàn Thịnh.