

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



Báo cáo đồ án cuối kỳ môn Kỹ thuật Lập trình



Fast and Safe



Thực hiện bởi:

Nguyễn Trọng Nghĩa	1712615
Nguyễn Ngọc Lan Như	1712644
Nguyễn Đắc Phát	1712651
Mạnh Trọng Lâm Phong	1712658

Ngày 26 tháng 05 năm 2018

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



Báo cáo đồ án cuối kỳ môn Kỹ thuật lập trình



Fast and Safe



Thực hiện bởi:

Nguyễn Trọng Nghĩa	1712615
Nguyễn Ngọc Lan Như	1712644
Nguyễn Đức Phát	1712651
Mạnh Trọng Lâm Phong	1712658

Giáo viên hướng dẫn: Lê Ngọc Thành

Ngày 26 tháng 05 năm 2018

TÀI LIỆU THAM KHẢO

- [1]<https://cachhoc.net/2013/08/22/cc-gotoxy-trong-dev-c-gotoxy-in-dev-c/>
- [2]<https://cachhoc.net/2013/07/16/cc-mau-chu-trong-dev-c-how-do-change-color-of-text-in-dev-c/>
- [3]<https://nguyenvanquan7826.wordpress.com/2013/08/22/cc-gotoxy-trong-dev-c-gotoxy-in-dev-c/>
- [4]<https://daynhauhoc.com/t/an-con-tro-chuot-tren-man-hinh-console/34411>
- [5]<https://daynhauhoc.com/t/lenh-bat-phim-trong-lap-trinh-c/16736/3>

I. QUÁ TRÌNH LÀM VIỆC, PHÂN CÔNG

Thời gian	Người làm	Công việc	Hoàn thành	Chưa hoàn thành
Tuần 1	Cả nhóm	Tìm hiểu về đồ án	Có ý tưởng hoàn thành đồ án	Chưa có code để cụ thể ý tưởng
Tuần 3	Phát	- Code được cho xe và vật cản xuất hiện và chạy được	- Xe và vật cản hiển thị được trên màn hình - Xe chạy được dùng điều khiển là các nút a s d w	- Màn hình chớp nháy liên tục, xe và vật cản chạy không mượt. - Mỗi lần điều khiển chương trình sẽ dừng lại một khoảng thời gian.
	Phong	- Cải tiến để cho xe và vật cản chạy mượt hơn - Xuất hiện nhiều vật cản cùng một lúc	- Màn hình không còn chớp như lúc đầu -> thay clrscr bằng xóa rồi in lại - Xuất hiện được ba vật cản lần lượt trên đường đua một cách ngẫu nhiên	- Đôi khi xe đụng vào vật cản mà không dừng game. - Còn nhiều lần các vật cản xuất hiện cùng một chỗ
	Nghĩa	- Viết hàm để sắp hạng người chơi	- Có file BXH để lưu dữ liệu người chơi. - Sắp xếp theo thứ tự từ cao xuống thấp theo điểm	- Còn bị lỗi ở phần nhập tên người chơi - Nhập rồi không cho chơi nữa
	Như	- Tô màu được đua, xe và vật cản.	- Đường đua được tô màu - Xe và vật cản đã được tô màu	- Đường đua không có cảm giác đang chạy

Tuần 10	Phát	<ul style="list-style-type: none"> - Điều chỉnh hiển thị console cho game - Code auto-pilot mode 	<ul style="list-style-type: none"> - In ra màn điều khiển mà không cần phải nhập để chọn, dùng mũi tên lên xuống để chọn menu - Xe tự động né vật cản được 	<ul style="list-style-type: none"> - Xe chưa tự động ăn bonus được - Vẫn có trường hợp xe chưa tự động né được
	Phong	<ul style="list-style-type: none"> - Cải tiến điều khiển di chuyển của xe. - Làm vật cản có thể di chuyển được 	<ul style="list-style-type: none"> - Thay điều khiển bằng phím a s d w thành các phím mũi tên ←→↑↓ - Cứ 3 vật cản sẽ có một vật cản đột ngột di chuyển. 	
	Nghĩa	<ul style="list-style-type: none"> - Sửa bug của BXH - Làm vật bonus 	<ul style="list-style-type: none"> - Cho người dùng nhập tên được. - Làm vật khi ăn được thì sẽ thêm điểm, hay tốc độ chậm lại, tăng mạng 	
	Như	- Tô màu cho từng vật bonus	- Tô màu được vật bonus	
Tuần 14	Cả nhóm	<ul style="list-style-type: none"> - Xem và sửa lỗi - Xem và chỉnh sửa báo cáo 		
	Như	- Viết báo cáo, ghi chú code và hàm	- Có báo cáo để cả nhóm góp ý vào một tuần trước khi nộp	

II. THIẾT KẾ CHƯƠNG TRÌNH

2.1 Library.h

- Chứa tất cả các hàm được dùng trong chương trình

2.1.1. struct QUEUE:

- Định nghĩa:

```
struct node
{
    int data;
    struct node *pNext;
};
typedef struct node NODE;
struct queue
{
    NODE *pHead;
    NODE *pTail;
};
typedef struct queue QUEUE;
```

- Ứng dụng: Dùng trong chương trình để làm chế độ auto mode

- Các hàm:

```
void KhoiTaoQueue(Queue &q);
```

Input: một biến kiểu Queue.

Output: khởi tạo biến với head với tail đều trở vào NULL.

```
Node *KhoiTaoNode(int x);
```

Input: giá trị x

Output: Node mang giá trị x và đang trở vào NULL

```
bool IsEmpty(Queue q);
```

Input: biến Queue

Output: Trả về true nếu q là hàng đợi rỗng, còn ngược lại trả về false.

```
bool Push(Queue &q, Node *p);
```

(push_back)

Input: biến Queue q và một Node mang giá trị p.

Output: Nếu thêm vào được hàng đợi thì trả về true, còn nếu không trả về false.

```
bool Pop(Queue &q, int &x);
```

Input: biến Queue q và một biến để chứa giá trị x.

Output: Nếu lấy ra được giá trị thì trả về true, còn nếu không thì trả về false.

(pop_back)

```
bool Top(Queue &q, int &x);
```

Input: biến Queue q và một biến để chứa giá trị x.

Output: Nếu lấy ra được giá trị thì trả về true, còn nếu không thì trả về false.

(pop_front).

2.1.2. Vật cản

- Định nghĩa:

```
struct Can
```

```
{
```

```
    int c;
```

```
    int d;
```

```
    int sign;
```

```
};
```

- Dùng để làm vật cản.

- Các hàm:

```
void Xoavatcan(char a[50][30], Can &x);
```

- Hàm dùng để xóa vật cản x

Input: Truyền vào mảng a có chiều dài và chiều rộng ứng với chiều dài và chiều rộng của màn hình console game.

Output: Vật cản được xóa.

```
void VeCan(char a[50][30], int &x, int y);
```

- Hàm dùng để vẽ lại vật cản và đổ màu cho vật cản

- Input: truyền vào là mảng console a và vị trí tọa độ của vật cản.

- Output: được vật cản như hình (include hình)

2.1.3. Xe

- Các hàm liên quan đến Xe: Init, kiểm tra xem xe có đụng vào vật cản

- Các hàm:

```
void XeF1(char a[50][30], int x, int y);
```

- Hàm dùng để in xe, không có tô màu.

- Input: mảng console a, giá trị tọa độ của xe.

- Output: được xe như hình (include hình)

```
void XoaXeF1(char a[50][30], int x, int y);
```

- Hàm dùng để xóa xe, thường dùng để xóa rồi in lại tại vị trí mới tạo cảm giác di chuyển.
- Input: mảng console a và tọa độ của xe.
- Output: Xóa xe tại vị trí x, y.

```
int test(char a[50][30], Can vatcan[10], int x, int y, int k, int &o);
```

- Hàm kiểm tra xem xe đã đụng vào vật cản chưa
- Input: mảng console a, tọa độ vật cản, số vật cản, tọa độ của xe
- Output: hàm trả về 1 nếu xe đã va phải vật cản và trả về 0 nếu xe vẫn chưa đụng vào vật cản.

```
int Xuli(int xux, int xuy, int x, int y);
```


- Hàm kiểm tra xem xe có ăn được bonus ko
- Input: tọa độ của xe và tọa độ của bonus
- Output: Nếu ăn được thì trả về 1 còn nếu không thì trả về 0

2.1.4. Bonus

- Các vật thưởng có thể nhặt trên đường đi
- Định nghĩa:


```
struct xu
```

```
{
    char c = '@';
    int x;
    int y;
};
```

- Vật thưởng thêm tiền 


```
struct Slow
```

```
{
    int x;
    int y;
}; (include hình)
```

- làm cho tốc độ chậm lại 

```
struct Life
```

```
{
    int h;
    int t;
}; (include hình)
```

- giúp tăng mạng cho người chơi 

2.1.5. Người chơi

- Định nghĩa: người chơi gồm có tên (tối đa 29 ký tự) và điểm khi chơi.

```
struct player
```

```
{
    char ten[30];
    int diem;
};
```

- Các hàm:

```
void XepHang(player choi[20], int &nplayer);
```

- Input: mảng người chơi, và số lượng người chơi hiện tại.
- Output: Xếp hạng từ trên xuống thấp theo điểm.

```
void BXH();
```

- In mảng BXH từ trong file ra ngoài màn hình console.

2.1.6. Menu điều khiển

```
enum TRANGTHAI { up, down, enter, esc, backspace };
```

- định nghĩa các trạng thái điều khiển.
- up, down thường dùng để điều khiển lên xuống trong chọn menu.

Enter: dùng để chọn.

Backspace: trở lại menu.

Esc: Thoát game.

```
TRANGTHAI key(int a);
```

- Input: nhập phím từ bàn phím
- Output: trả về một trạng thái ứng với phím đó

```
int DieuKien(str Menu[], int n);
```

- Input: biến kiểu str đã được định nghĩa tại dòng:

```
str Menu[5] = { "====FASR AND SAFE====", " 1.PLAY", " 2.AUTOPLAY", " 3.RANK", "4.EXIT" };
```

trong hàm main, và kích thước của mảng.

- Output: trả về thứ tự ứng với lựa chọn trên menu.

1. PLAY: chơi đua xe

2. AUTOPLAY: chế độ chơi tự động

3. RANK: xem bảng xếp hạng

4. EXIT: thoát game.

Cũng được áp dụng với Menu Continue được định nghĩa ở dòng

```
str Continue[3] = { "CONTINUE?", "YES", "NO" }; trong main.
```

2.1.7. Tô màu

```
void InitColor(int color[50]);
```

- Hàm set các giá trị ban đầu cho mảng color.
- Dùng để tạo cảm giác đường đua di chuyển bằng cách thay đổi màu lần lượt.

```
void XuatMang(char a[50][30], int n, int m, int color[50]);
```

- In toàn bộ màn hình trò chơi, cùng với màu như được chỉ định.

2.2 Console.h

- Sưu tầm từ trên mạng của Nguyễn Trung Thanh
- Header chứa các hàm đơn giản liên quan đến console.
- Các hàm:

```
void clrscr();
```

- Hàm dùng để xóa màn hình

```
void gotoXY (int column, int line);
```

- Hàm dùng để đi đến tọa độ trên màn hình
- Input: vị trí cột muốn đến và vị trí dòng muốn đến
- Output: con trỏ sẽ trở đến vị trí (column, line).

```
int whereX();
```

- Hàm trả về tọa độ X của vị trí con trỏ đang trỏ

```
int whereY();
```








- Hàm trả về tọa độ y của vị trí con trỏ đang trỏ.

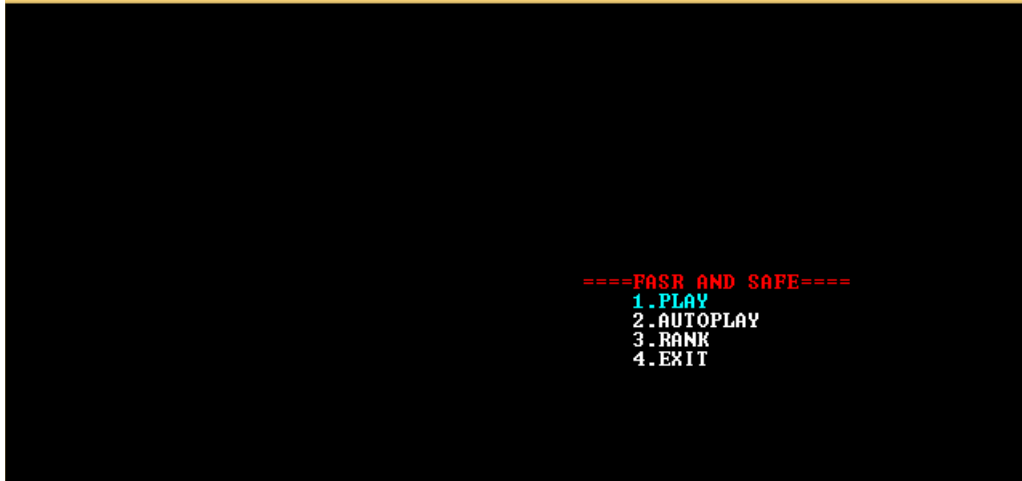
```
void TextColor (int color);
```

- Hàm chuyển màu của text sang màu được quy định bởi biến color nhập vào
- Input: biến số nguyên color quy định màu.
- Output: include hình

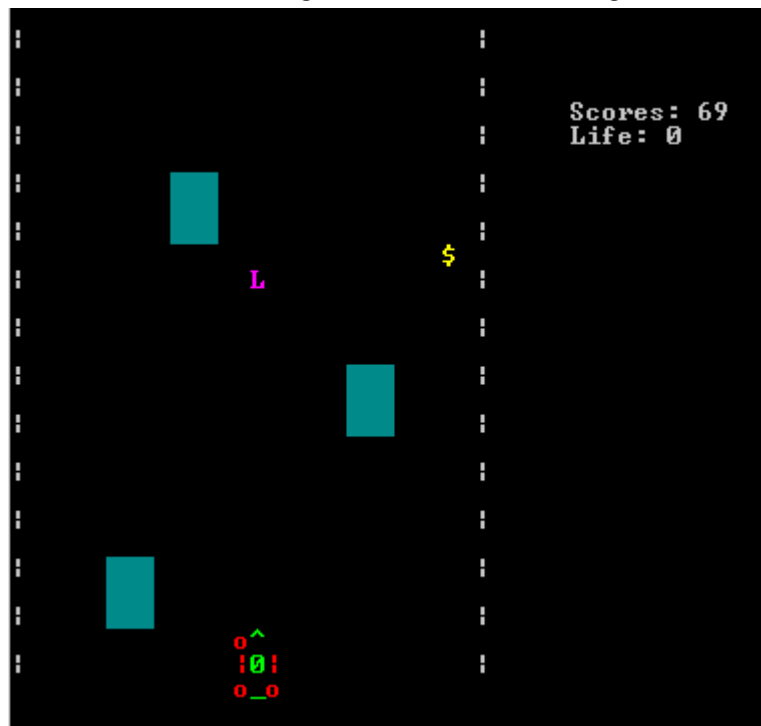
III. MÔ TẢ GAME

TÊN GAME: Fast and Safe

MÔ TẢ: Điều khiển xe  (sau được cải tiến thành ) để né các vật cản  (sau được cải tiến thành ) trong khi vẫn tranh thủ ăn các bonus (  ) để được thêm điểm. Nếu đụng phải các vật cản thì game kết thúc. Tốc độ của xe và vật cản sẽ ngày càng nhanh theo thời gian.

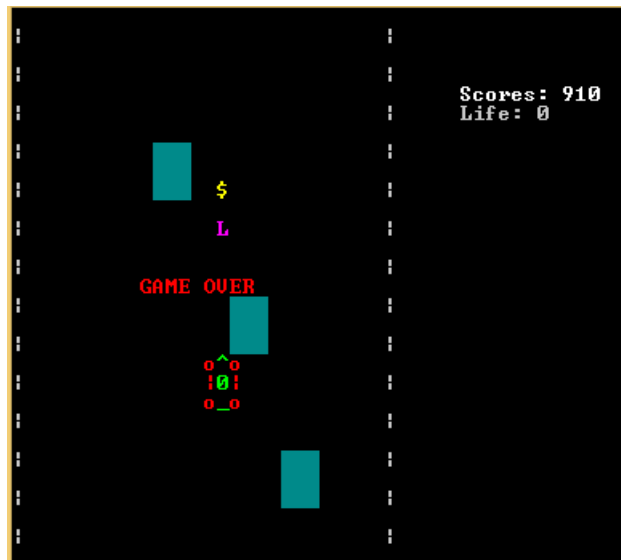


Màn hình Menu mới Load, dùng mũi tên để lên xuống và Enter để chọn



Màn hình Game Lúc chơi

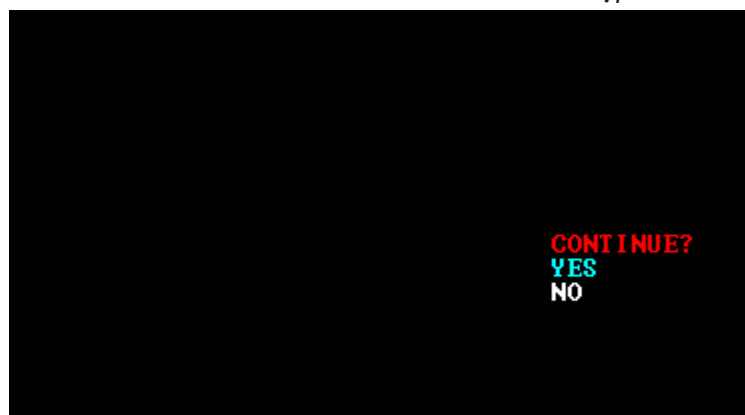
Dùng các phím mũi tên ←→↑↓ để di chuyển



Màn hình game khi kết thúc



Sau khi kết thúc nhấn Enter để nhập tên



Link Video demo: <https://drive.google.com/open?id=1Gc60S8FLvGMoaqFWswNauyUDfBMGqVJs>