Project 2

LA Traffic

CSC-5-42829

Jesus E. Moran Rivera

05/02/2016

**Introduction**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Title: LA Traffic

LA Traffic is a simple obstacle dodging game where the player takes control of the car represented by ‘<’. The goal of the game is to avoid the oncoming traffic, represented as ‘]’, '}', and ')' by changing lanes. Jumping lanes is done by using the number keys starting from 1 up to 4 with each successful dodge giving the player 10 points. If the player hits an obstacle they lose a life until they hit 0 and cause a game over. After the game over, the player can input their name and it is outputted to a scoreboard along with the points.

**Summary**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project Size: 208 lines

Variables: About 14

I got the idea for the game after remembering some of the old games i played in those cheap 99 cent store handheld games which included a similar game. The second inspiration comes from those endless games like “Flappy Bird” where the only objective is to get a score that is higher than your peers.

In searching for a way to make the grid I found examples of arrays and played around with it, after learning the do's and don'ts in class it was a small fix to get the program to updated and it is now possible to change a few variables to completely change the grid.

While the game itself isn't very fast paced for now, it can be later updated to increase the rate of obstacles appearing and possibly toggle difficulty by increasing the rate and giving the player more points and less lives during the game.

Overall, the project took a lot of work and dedication, it was a big challenge to make time for it and it's the best I can do for now but plan on working on it just to play around with C++ some more and find out what else I can stick on it. I hope that in the future I will be able to stick a graphical interface and make an actual game.

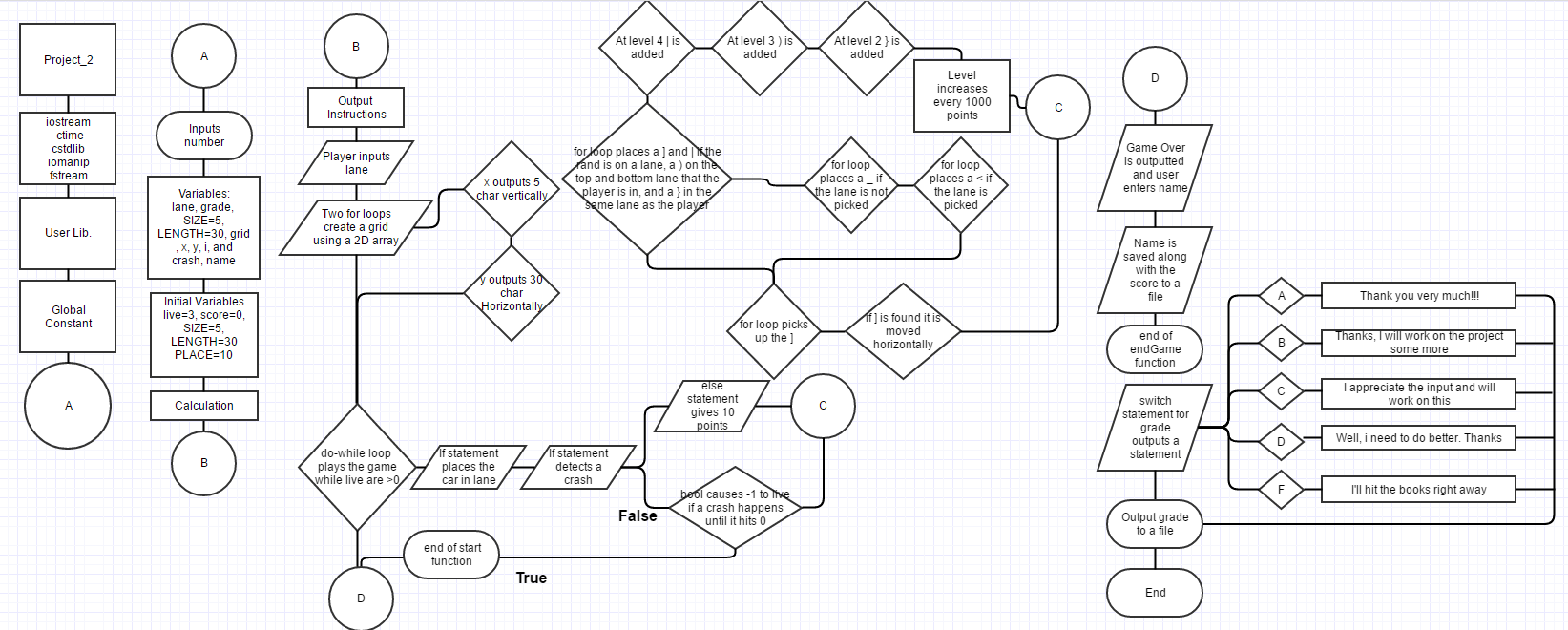
**Description**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

The program demonstrates the use of all learned material: Functions, Strings, Data types, Loops, Decision making, file input /output, and formatting.

**Flow Chart**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



(See Project\_2\_FlowChart.png in Project\_2 folder)

**Pseudo Code**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//Initial instructions for playing

//Calls the game

//Grading prompt

//Calls the grading scale

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Game mechanics

//Generates initial map of 5 by 30

//Places "car" on the grid if it is above -1

//Do-while loop runs the program as long as the player has lives

//Keeps track of lives for the player

//Changes lanes for the player

//Verifies that the inputs are in the lane range

//If player hits a block they lose a live

//Else if player avoids obstacles it adds 100 points

//Random lane selection for Obstacles

//Places obstacle once a lane is picked

//Place an obstacle on the same lane as player

//Places obstacles on the lanes above and below the player

//Places car in the selected lane

//Places obstacle and moves it across the grid once a new one spawns

//At level 2 the obstacle is added

//Places obstacle and moves it across the grid once a new one spawns

//At level 3 the obstacle is added

//Places obstacle and moves it across the grid once a new one spawns

//At level 4 the obstacle is added

//Places obstacle and moves it across the grid once a new one spawns

//Outputs grid

//Else statement if input is above or below the lanes

//If statement calls the endGame function for the scoreboard

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Scoreboard

//Outputs the game over

//Asks for a player name to input into scoreboard

//Outputs score at the end of the game

//Output the results to a file

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Switch statement for grading

//Outputs grade to a file

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Major Variables**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

unsigned short lane; //Places the car in a lane

char grade; //Input for grade switch statement

const int SIZE=5, LENGTH=30; //Size and Length of the array

char grid[SIZE][LENGTH]; //Array for the lanes

unsigned short x; //Variable increases until it reaches SIZE

unsigned short y; //Variable increases until it reaches LENGTH

unsigned short i; //Variable picks lane from rand

unsigned short live=3; //Lives of the player

unsigned short score=0; //Score starts at 0 and increases by one if no crash is detected

bool crash=false; //Crash is false as long as the player does not cash

const short PLACE=10; //Character limit for name

char name[PLACE]; //Variable for player input name

Program

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//Function Prototype

void start(unsigned short);

void endGame(unsigned short, int);

void grading(char);

//Execution Begins Here

int main(int argc, char\*\* argv) {

//Declare Variables

unsigned short lane;

char grade;

ifstream in;

//Initial instructions for playing

cout<<"Use the number keys to move the car and press enter after each input. \n"

"Use 1 to 4 and press any other key to exit."<<endl<<endl;

//Calls the game

start(lane);

cout<<endl;

//Grading prompt

cout<<"Please take some time to grade the Project."<<endl<<endl;

cout<<"Please place a letter grade in caps:"<<endl;

cin>>grade;

//Calls the grading scale

grading(grade);

return 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void start(unsigned short lane)

{

//Declare Variables

const int SIZE=5, LENGTH=30;

char grid[SIZE][LENGTH];

unsigned short x, y, i, live=3, score=0, level=0;

bool crash=false;

//Generates initial map of 5 by 30

for (x=0;x<SIZE;x++){

for (y=0;y<LENGTH;y++){

grid[x][y]='\_';

grid[x][y];

}

}

//Places "car" on the grid if it is above -1

lane=0;

//Do-while loop runs the program as long as the player has lives

do{

//Keeps track of lives for the player

cout<<"Lives: "<<live<<endl;

//Changes lanes for the player

cout<<"Pick a lane:"<<endl;

cin>>lane;

//Verifies that the inputs are in the lane range

if(lane>=1&&lane<=4){

//If player hits a block they lose a live

if(grid[lane][28]==']'||grid[lane][28]=='}'||grid[lane][28]==')'||

grid[lane][28]=='|'){

crash=true;

live-=1;

}

//Else if player avoids obstacles it adds 100 points

else{

score+=100;

//Every 1000 points the level is increased

if(score==1000){

level=2;

}

if(score==2000){

level=3;

}

if(score==3000){

level=4;

}

if(score==4000){

level=5;

}

//Outputs current score and level

cout<<"Score: "<<score<<endl;

cout<<"Level: "<<level<<endl;

}

//Random lane selection for Obstacles

srand(static\_cast<int>(time(0)));

i=rand()%5+1;

//Places obstacle once a lane is picked

grid[i][0]=']';

grid[i][0]='|';

//Place an obstacle on the same lane as player

grid[lane][0]='}';

//Places obstacles on the lanes above and below the player

if(lane>1&&lane<4){

grid[lane-1][0]=')';

grid[lane+1][0]=')';

}

for(x=0;x<SIZE;x++){

grid[x][29]='\_';

}

//Places car in the selected lane

grid[lane][29]='<';

//Places obstacle and moves it across the grid once a new one spawns

for (x=0;x<SIZE;x++){

for (y=0;y<LENGTH;y++){

if(y<=29){

if(grid[x][y]==']'){

grid[x][y+1]=grid[x][y];

grid[x][y]='\_';

y++;

}

}

}

}

//At level 2 the obstacle is added

if(level>=2)

//Places obstacle and moves it across the grid once a new one spawns

for (x=0;x<SIZE;x++){

for (y=0;y<LENGTH;y++){

if(y<=29){

if(grid[x][y]=='}'){

grid[x][y+1]=grid[x][y];

grid[x][y]='\_';

y++;

}

}

}

}

//At level 3 the obstacle is added

if(level>=3)

//Places obstacle and moves it across the grid once a new one spawns

for (x=0;x<SIZE;x++){

for (y=0;y<LENGTH;y++){

if(y<=29){

if(grid[x][y]==')'){

grid[x][y+1]=grid[x][y];

grid[x][y]='\_';

y++;

}

}

}

}

//At level 4 the obstacle is added

if(level>=4)

//Places obstacle and moves it across the grid once a new one spawns

for (x=0;x<SIZE;x++){

for (y=0;y<LENGTH;y++){

if(y<29){

if(grid[x][y]=='|'){

grid[x][y+1]=grid[x][y];

grid[x][y]='\_';

y++;

}

}

}

}

//Outputs grid

for (x=0;x<SIZE;x++){

for (y=0;y<LENGTH;y++){

cout<<grid[x][y];

}

cout<<endl;

}

//Else statement if input is above or below the lanes

}else

cout<<"Stay in the lane!!!"<<endl;

}while(live>0);

//If statement calls the endGame function for the scoreboard

if (live<1)

endGame(live, score);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void endGame(unsigned short live, int score)

{

//Declare Variables

const short PLACE=10;

char name[PLACE];

//Outputs the game over

cout<<"Game Over!!!"<<endl<<endl;

//Asks for a player name to input into scoreboard

cout<<"Enter your name: (max of 10 characters)\n";

cin>>name;

cout<<endl;

//Outputs score at the end of the game

cout<<"Name Score"<<endl;

cout<<name<<setw(13)<<score<<endl;

ofstream out;

//Output the results to a file

out.open("LA\_Traffic\_Score.dat",ios::app);

out<<"\r\n"<<name<<setw(13)<<score<<endl;

out.close();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void grading(char grade)

{

ofstream out;

//Switch statement for grading

switch (grade){

case 'A':{

cout<<"Thank you very much!!!"<<endl;

break;

}

case 'B':{

cout<<"Thanks, I will work on the project some more."<<endl;

break;

}

case 'C':{

cout<<"I appreciate the input and will work on this."<<endl;

break;

}

case 'D':{

cout<<"Well, i need to do better. Thanks."<<endl;

break;

}

case 'F':{

cout<<"I'll hit the books right away."<<endl;

break;

}

default:{

cout<<"Enter a valid letter grade or use capitols."<<endl;

}

}

//Outputs grade to a file

out.open("Grade.dat",ios::app);

out<<"\r\n"<<grade<<endl;

out.close();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/