INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA

Project Title: Car Persist

Course title : ELEMENTS OF PROGRAMMING
Course code : CSC 1100
Section : 02

Lecturer Name : Dr. Afidalina Tumian

Group members

1. ABID EBNA SAIF UTSHA (1433527)
2. MAHFUZEALAHI NOMAN (1515803)
3. TANVIR HOSSAIN (1437699)
4. TANVIR FORAZI (1432511)
5. MD ABU HAMED MOHIUDDIN RIAD (1428315)

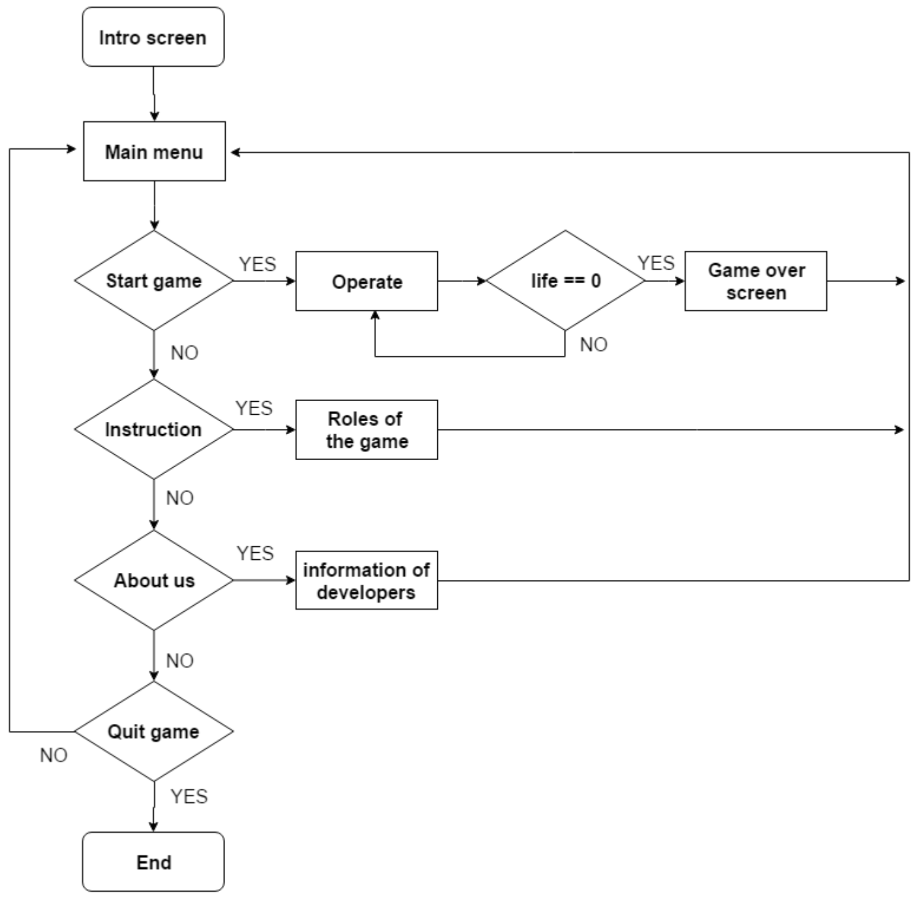
Introduction

We are developing a game named ‘Car Persist’ based on car racing. The theme of our game is to survive as long as possible while trying to avoid bumping to other cars. The player’s goal is to finish levels by staying away from cars those are coming from opposite direction. This game uses board to control player in each turn player press an arrow key to move left, right, up or down. The score will be according to the number of avoided cars and bonus points will be provided through an object. Level will be increasing when the player get a certain score. The speed of car will quicken and add new features such as life destroyer and oncoming cars. Two lives will be provided every time player starts a new game. The game will over if the car coming from opposite direction crash with the player’s car or pick up life destroyer object. When the game is over, it will show the score and save the score if previous score is beaten. Then, main menu will be shown.

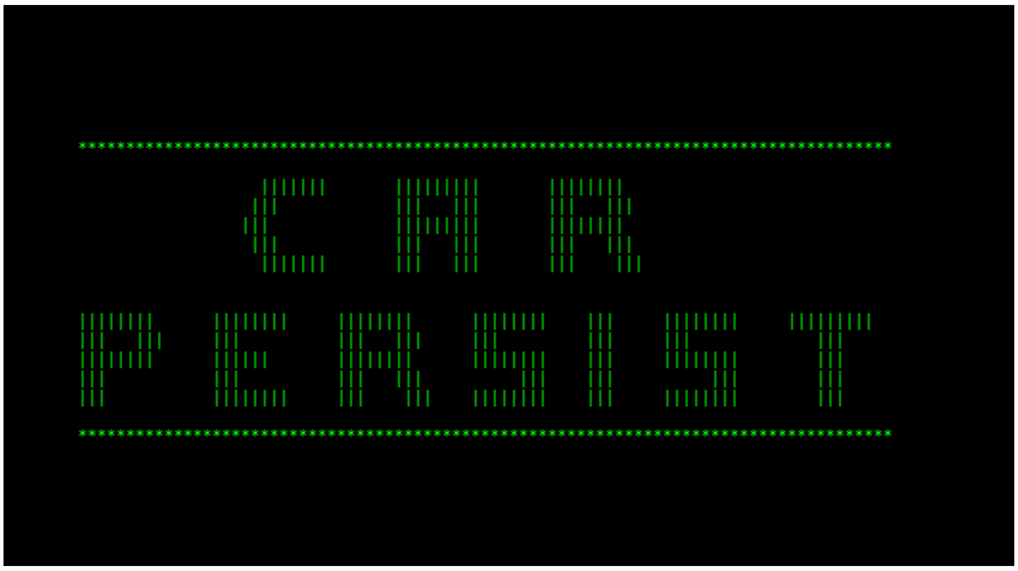
Most of us have played this game in our childhood. We have make a C++ program on it.

Method

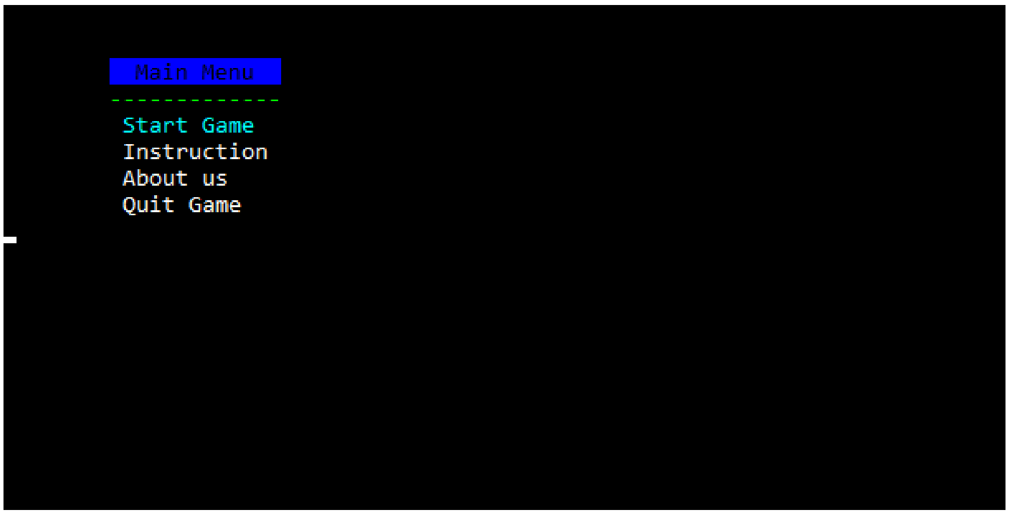
Diagram of ‘Car Persist’ design:



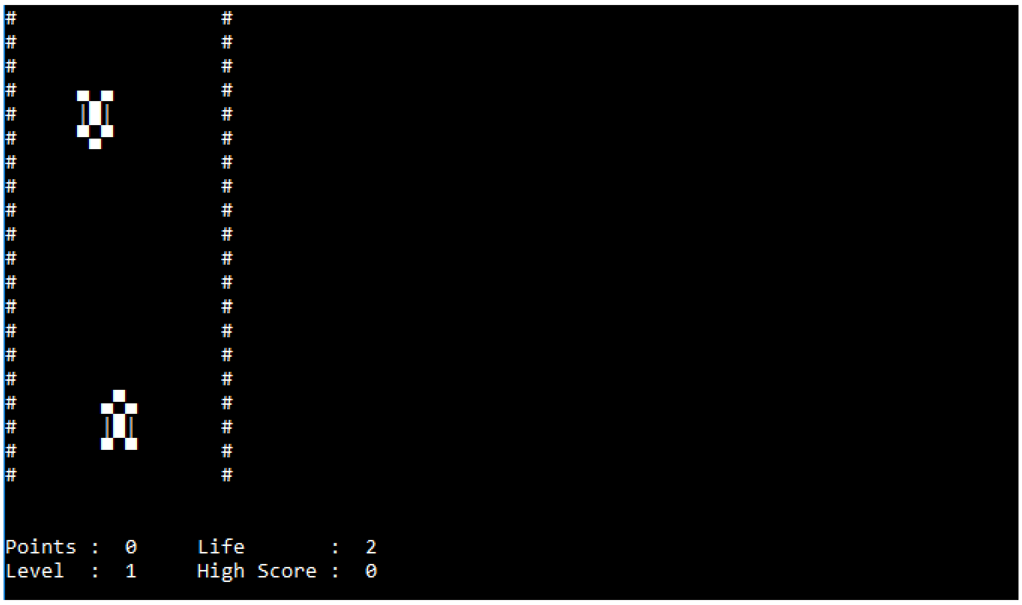
Result



When the game compiles, it will show this screen first.

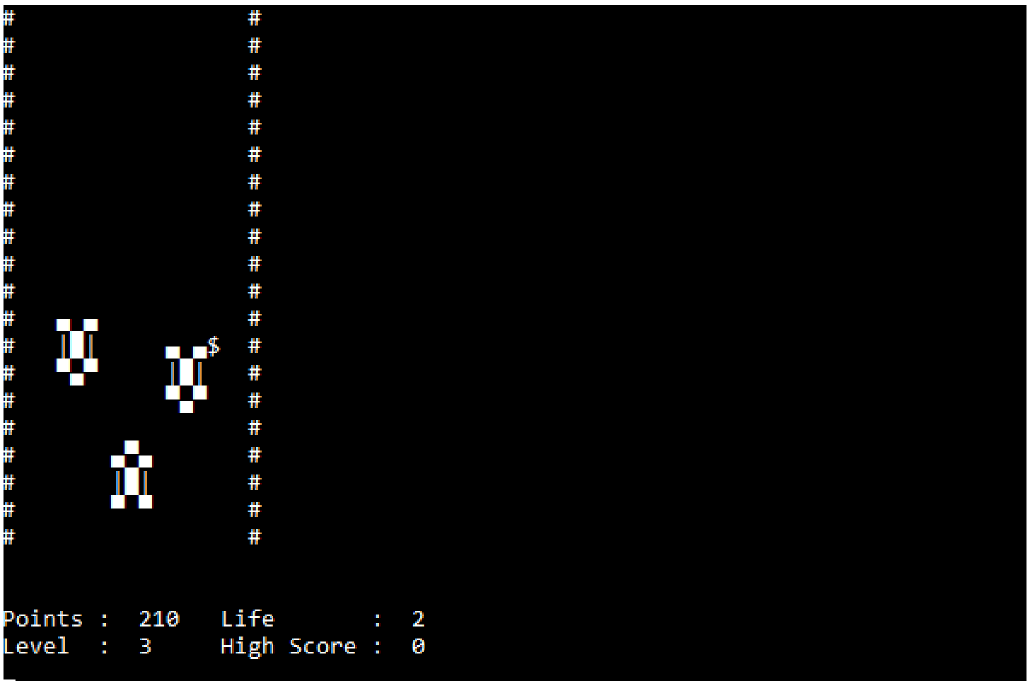


Then, this main menu will be shown. User car choose any option.



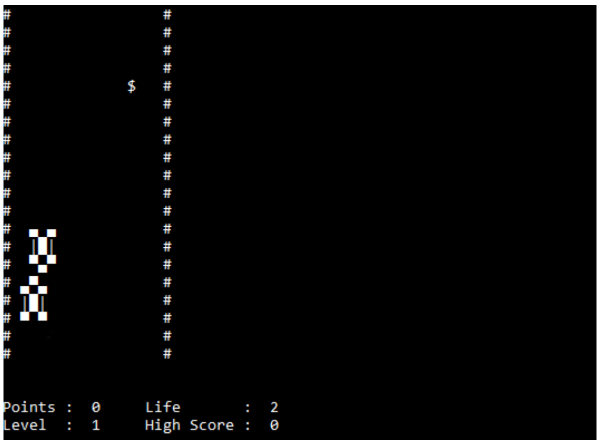
```
Points :  0     Life       :  2
Level  :  1     High Score :  0
```

If user chooses ‘Start game’, it will begin processing the game. On the bottom of screen points, life, level and previous high score will be shown respectively.

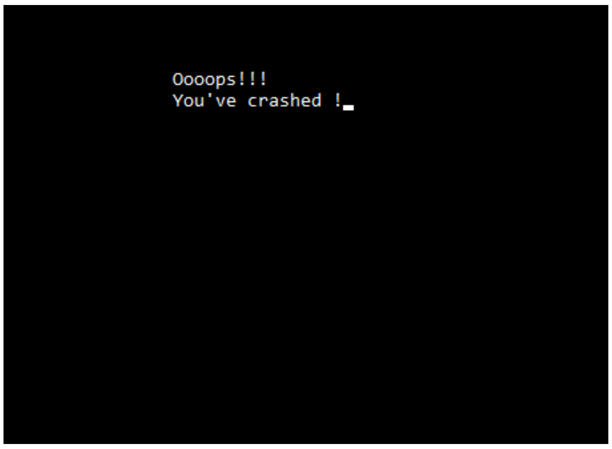


```
Points :  210   Life       :  2
Level  :  3     High Score :  0
```

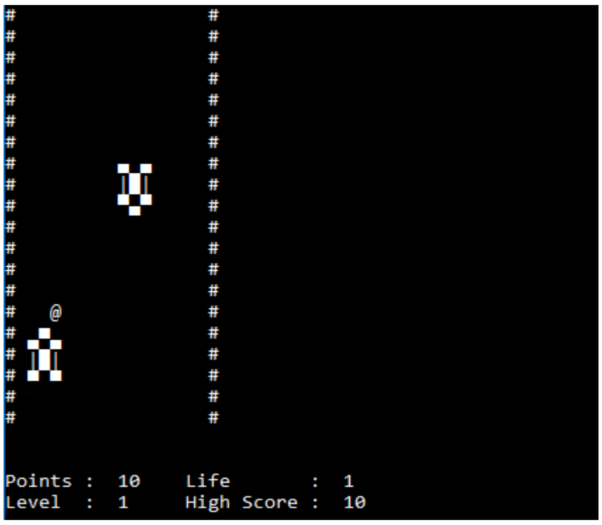
User gets points when he/she avoids each oncoming car. Bonus points will be given through an object. After achieving a certain point, level will be increase gradually. At level three onwards, forthcoming car will be doubled.



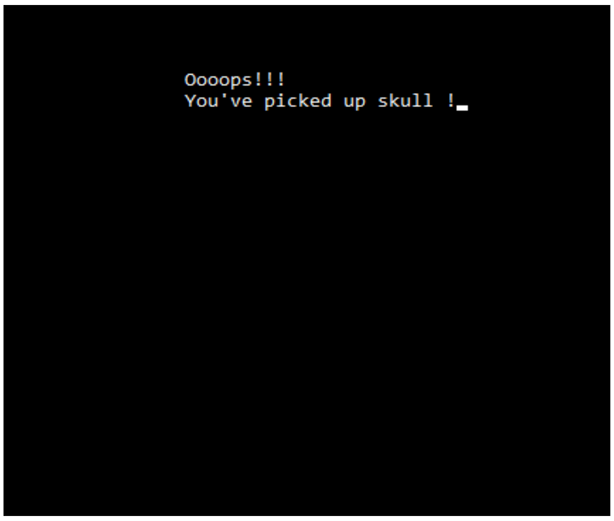
```
Points :  0     Life       :  2
Level  :  1     High Score :  0
```



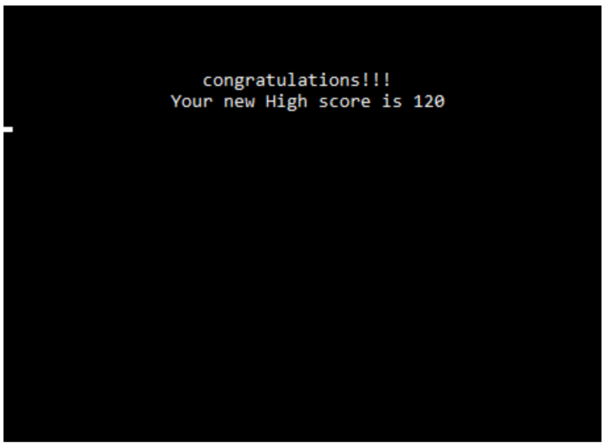
```
Oooops!!!
You've crashed !
```



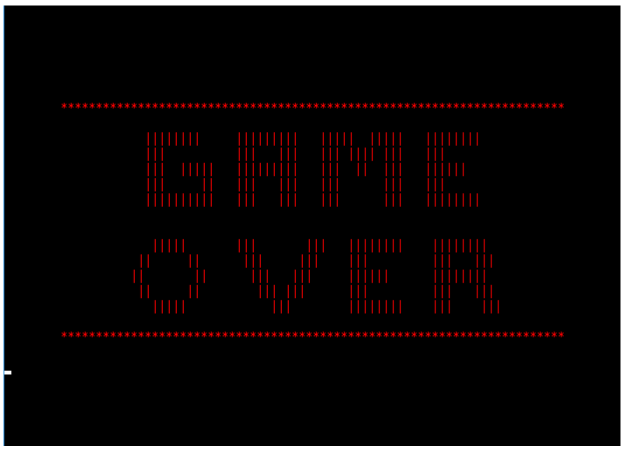
```
Points :  10    Life       :  1
Level  :  1     High Score :  10
```



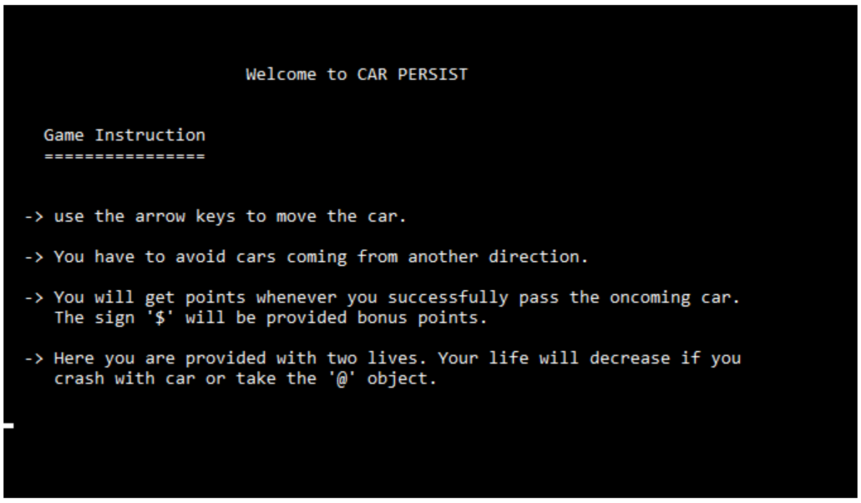
```
Oooops!!!
You've picked up skull !
```



```
   congratulations!!!
Your new High score is 120
```

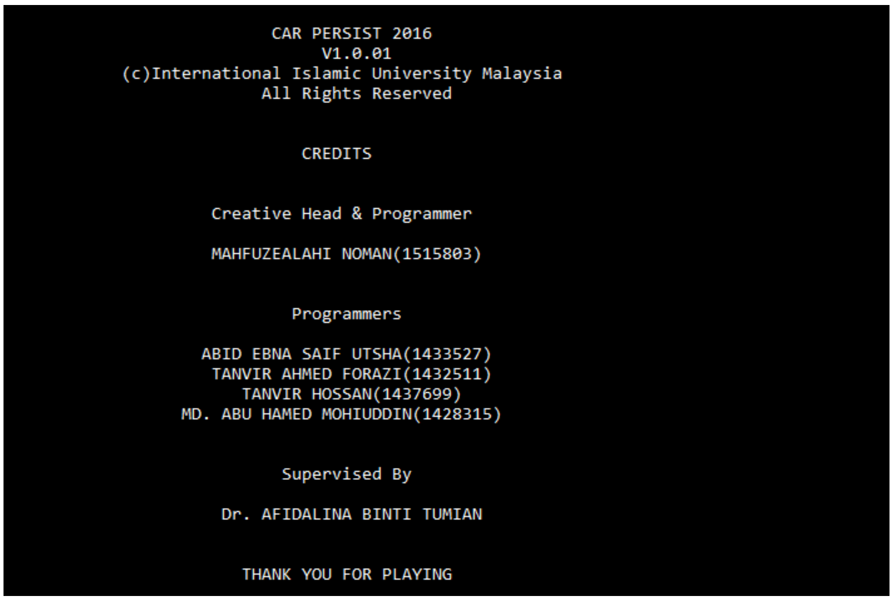


Whenever the user's car crashes with oncoming car or life destroyer object, life will decrease and a warning screen will be shown. If all lives have been used up, final score and game over screen will be shown.



```
                    Welcome to CAR PERSIST


  Game Instruction
  ================


-> use the arrow keys to move the car.

-> You have to avoid cars coming from another direction.

-> You will get points whenever you successfully pass the oncoming car.
   The sign '$' will be provided bonus points.

-> Here you are provided with two lives. Your life will decrease if you
   crash with car or take the '@' object.
```

If user chooses 'Instruction', roles of the game will be displayed.



```
                         CAR PERSIST 2016
                              V1.0.01
            (c)International Islamic University Malaysia
                        All Rights Reserved


                            CREDITS


                    Creative Head & Programmer

                    MAHFUZEALAHI NOMAN(1515803)


                           Programmers

                   ABID EBNA SAIF UTSHA(1433527)
                    TANVIR AHMED FORAZI(1432511)
                       TANVIR HOSSAN(1437699)
                 MD. ABU HAMED MOHIUDDIN(1428315)


                           Supervised By

                     Dr. AFIDALINA BINTI TUMIAN


                       THANK YOU FOR PLAYING
```

If user chooses 'About us', information of game developers will be displayed.

If user chooses 'Quit game', the program will terminate.

Conclusion

We have faced difficulties when we wanted to crash between cars because of matching array positions. As well as, creating main menu was a bit of complex.

As a suggestion, graphic and audio can be added to improve user convenience.

Reference

http://pastebin.com/gU2hMXM7

Appendix (Source codes)

```
#include <iostream>
#include <string>
#include <ctime>
#include <conio.h>
#include <fstream>
#include <windows.h>

using namespace std;

void IntroScreen();
void MainMenu();
void Logic();
void AboutUs();
void Instruction();
void EndScreen();

int main()
{
   IntroScreen();
   MainMenu();
   return 0;
}

void IntroScreen(){
         system("COLOR A"); //set color
         system("CLS"); // clear screen for windows
          cout<<"\n\n\n\n\n\n\n";
         cout<<"\t************************************************************
************************\n\n";
   cout<<"\t               ||||||      ||||||||      ||||||| "<<endl;
   cout<<"\t              |||          |||  |||      |||  |||   "<<endl;
   cout<<"\t             |||           ||||||||      |||||||    "<<endl;
   cout<<"\t              |||          |||  |||      |||  |||   "<<endl;
   cout<<"\t               ||||||      |||  |||      |||   |||  "<<endl<<endl<<endl;
   cout<<"\t|||||||     ||||||||    |||||||     |||||||   |||    |||||||    ||||||||"<<endl;
   cout<<"\t|||  |||    |||  |||    |||  |||    |||       |||    |||           |||"<<endl;
   cout<<"\t|||||||     ||||||||    |||||||     |||||||   |||    |||||||       |||"<<endl;;
   cout<<"\t|||         |||  |||    |||  |||        |||   |||        |||       |||"<<endl;
   cout<<"\t|||         |||  |||    |||   |||   |||||||   |||    |||||||       |||\n"<<endl;
cout<<"\t******************************************************************
******************\n\n";
   Sleep(1500);//pause the time
   return ;  }

void MainMenu()
{
   string menu[4] = {"Start Game", "Instruction", "About us", "Quit Game"};
   int pointer = 0;
   //print main menu
```

```
while(true)
   {
      system("CLS");
      HANDLE h = GetStdHandle(STD_OUTPUT_HANDLE);
      SetConsoleTextAttribute(h, BACKGROUND_BLUE | BACKGROUND_INTENSITY);
      cout << "\n\n\t  Main Menu  \n";
      SetConsoleTextAttribute(h,FOREGROUND_GREEN | FOREGROUND_INTENSITY);
      cout << "\t-------------\n";

      for(int i=0; i<4; ++i)
      {
         if(i == pointer)
         {
            SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),11);
            cout << "\t " << menu[i] << endl;
         }
         else
         {
            SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),15);
            cout << "\t " << menu[i] << endl;
         }
      }
      //control main menu
      while(true)
         {
                  if (GetAsyncKeyState(VK_UP) != 0)
                  {
                           pointer -= 1;
                           if (pointer == -1)
```

```
        {
                pointer = 2;
        }
                break;
        }
else if (GetAsyncKeyState(VK_DOWN) != 0)
{
        pointer += 1;
        if (pointer == 4)
        {
                pointer = 0;
        }
        break;
}
if (GetAsyncKeyState(VK_RETURN) != 0)
{
        if(pointer == 0)
        {
        cout << "\n\n\n\tStarting new game...";
        Sleep(1000);
        Logic();
        }


        else if(pointer == 1)
        {
                Instruction();
        }


        else if(pointer == 2)
```

```
                        {
                              AboutUs();
                        }

                        else if(pointer == 3)
                        {
                              return ;
                        }
                  }
                  break;
            }
            Sleep(150);
            }
    }

void Logic()
{
    char map[25][25];

    //loads the map with spaces and borders
    for(int i=0; i < 20; ++i)
    {
      for(int j=0; j < 20; ++j)
      {
            map[i][0] = '#';
            map[i][18] = '#';
            map[i][j] = ' ';
      }
    }
```

```
//read previous high score
int hscore;
ifstream inputFile;
inputFile.open("H.txt");
inputFile >> hscore;
inputFile.close();

int y = 17, x = 9; //the cars coordinates
srand(time(0));
int a = 0, b = rand() % 7 + 2; //the obstacles coordinates
int points = 0; //points that the player has earned
int speed = 160; //determines the speed of the obstacles (and the car)
int q = 0, p = rand() % 15 + 2; //the cash coordinates
int cashcheck = 0; //balances when the cash spawns
int cashpickedup = 0;
int c = 0, d = rand() % 4 + 11; //the second obstacles coordinates
bool startup = true;
int life = 2;
int level = 1;
int k = 0, l = rand()% 15 + 2;//the skull coordinates
char skull = '@';
int skullcheck = 0;

char cash = '$';
char obstacle1 = 219, obstacle2 = 220, obstacle3 = 223;

char car1 = 219, car2 = 220, car3 = 223;
map[y][x] = car1;
```

```
//the game loop
for(;;) {
  system("CLS");

    //places the car at its default location
    map[y][x] = car1;
    map[y][x+1] = '|';
    map[y][x-1] = '|';
    map[y+1][x-1] = car3;
    map[y+1][x+1] = car3;
    map[y-1][x-1] = car2;
    map[y-1][x+1] = car2;
    map[y-1][x] = car3;

  //generates the obstacles
    map[a][b] = ' ';
    map[a][b+1] = ' ';
    map[a][b-1] = ' ';
    map[a+1][b-1] = ' ';
    map[a+1][b+1] = ' ';
    map[a-1][b-1] = ' ';
    map[a-1][b+1] = ' ';
    map[a+2][b] = ' ';

    ++a;

    map[a][b] = obstacle1;
    map[a][b+1] = '|';
    map[a][b-1] = '|';
    map[a+1][b-1] = obstacle3;
```

```
map[a+1][b+1] = obstacle3;
map[a-1][b-1] = obstacle2;
map[a-1][b+1] = obstacle2;
map[a+1][b] = obstacle2;
if(a > 20) {
  if(level>2)
  {
    a = 0;
    b = rand() % 7 + 2;
  }
  else
  {
    a = 0;
    b = rand() % 15 + 2;
  }
}

//generates the second obstacles
if(level>2)
{
map[c][d] = ' ';
map[c][d+1] = ' ';
map[c][d-1] = ' ';
map[c+1][d-1] = ' ';
map[c+1][d+1] = ' ';
map[c-1][d-1] = ' ';
map[c-1][d+1] = ' ';
map[c+2][d] = ' ';

++c;
```

```
        map[c][d] = obstacle1;
        map[c][d+1] = '|';
        map[c][d-1] = '|';
        map[c+1][d-1] = obstacle3;
        map[c+1][d+1] = obstacle3;
        map[c-1][d-1] = obstacle2;
        map[c-1][d+1] = obstacle2;
        map[c+1][d] = obstacle2;
        if(c > 20) {
          c = 0;
          d = rand() % 4 + 11;
        }
        }
    //displays the map
    for(int i=0; i < 20; ++i)
{
      for(int j=0; j < 20; ++j)
{
        cout << map[i][j];
          if(j >= 19) {
            cout << endl;
          }
        }
      }
    cout << "\n\nPoints :  " << points <<"\tLife       :  " << life << endl;
    cout << "Level  :  " << level << "\tHigh Score :  " << hscore << endl;

    if(startup)
    {
      _getch();
      startup = false;
    }

    //moves the car to the left
    if(GetAsyncKeyState(VK_LEFT))
    {
        if((map[y][x-4] == obstacle1) || (map[y][x-4] == obstacle2) || (map[y][x-4] == obstacle3) ||
     (map[y-1][x-4] == obstacle1) || (map[y-1][x-4] == obstacle2) || (map[y-1][x-4] == obstacle3) ||
   (map[y+1][x-4] == obstacle1) || (map[y+1][x-4] == obstacle2) || (map[y+1][x-4] == obstacle3) ||
   (map[y-2][x-4] == obstacle1) || (map[y-2][x-4] == obstacle2) || (map[y-2][x-4] == obstacle3))
      {
          --life;
          if(life>0)
          {
             system("CLS");
             cout << "\n\n\n\t\tOooops!!!\n\t\tYou've crashed !";
             Sleep(1000);
             restart :
             {
             system("ClS");
             map[a][b] = ' ';
             map[a][b+1] = ' ';
             map[a][b-1] = ' ';
             map[a+1][b-1] = ' ';
             map[a+1][b+1] = ' ';
             map[a-1][b-1] = ' ';
    map[a-1][b+1] = ' ';
    map[a+2][b] = ' ';
    map[a+1][b] = ' ';

    map[c][d] = ' ';
    map[c][d+1] = ' ';
    map[c][d-1] = ' ';
    map[c+1][d-1] = ' ';
    map[c+1][d+1] = ' ';
    map[c-1][d-1] = ' ';
    map[c-1][d+1] = ' ';
    map[c+2][d] = ' ';
    map[c+1][d] = ' ';

    map[y][x] = ' ';
    map[y][x+1] = ' ';
    map[y][x-1] = ' ';
    map[y+1][x-1] = ' ';
    map[y+1][x+1] = ' ';
    map[y-1][x-1] = ' ';
    map[y-1][x+1] = ' ';
    map[y-1][x] = ' ';

    map[k][l] = ' ';

    y = 17;
    x = 9;
    a = 0;
    c = 4;
    k = 6;
} }
```

```
  if(life==0)
  {
      goto lost;
  }
}

else if((map[y][x-4] == skull) || (map[y-1][x-4] == skull) ||
        (map[y+1][x-4] == skull) ||(map[y-2][x-4] == skull) ||
        (map[y][x-3] == skull) || (map[y-1][x-3] == skull) ||
        (map[y+1][x-3] == skull) ||(map[y-2][x-3] == skull) ||
        (map[y][x-2] == skull) || (map[y-1][x-2] == skull) ||
        (map[y+1][x-2] == skull) ||(map[y-2][x-2] == skull))
{
      --life;
      if(life > 0)
      {
          system("CLS");
          cout << "\n\n\n\t\tOooops!!!\n\t\tYou've picked up skull !";
          Sleep(1000);
          goto restart;
      }
  if(life==0)
  {
      goto lost;
  }
}

else if(map[y][x-3] != '#') {
  map[y][x] = ' ';
```

```
            map[y][x+1] = ' ';
            map[y][x-1] = ' ';
            map[y+1][x-1] = ' ';
            map[y+1][x+1] = ' ';
            map[y-1][x-1] = ' ';
            map[y-1][x+1] = ' ';
            map[y-1][x] = ' ';
            x -= 3;
            map[y][x] = car1;
            map[y][x+1] = '|';
            map[y][x-1] = '|';
            map[y+1][x-1] = car3;
            map[y+1][x+1] = car3;
            map[y-1][x-1] = car2;
            map[y-1][x+1] = car2;
            map[y-1][x] = car3;
          }
        }
        //moves the car to the right
        if(GetAsyncKeyState(VK_RIGHT))
        {
 if((map[y][x+4] == obstacle1) || (map[y][x+4] == obstacle2) || (map[y][x+4] == obstacle3) || (map[y-1][x+4] == obstacle1) || (map[y-1][x+4] == obstacle2) || (map[y-1][x+4] == obstacle3) || (map[y+1][x+4] == obstacle1) || (map[y+1][x+4] == obstacle2) || (map[y+1][x+4] == obstacle3) || (map[y-2][x+4] == obstacle1) || (map[y-2][x+4] == obstacle2) || (map[y-2][x+4] == obstacle3))
          {
              --life;
              if(life > 0)
              {
```

```
        system("CLS");
        cout << "\n\n\n\t\tOooops!!!\n\t\tYou've crashed !";
        Sleep(1000);
        goto restart;
      }
  if(life==0)
  {
      goto lost;
  }
}

else if((map[y][x+4] == skull) || (map[y-1][x+4] == skull) ||
        (map[y+1][x+4] == skull) ||(map[y-2][x+4] == skull) ||
        (map[y][x+3] == skull) || (map[y-1][x+3] == skull) ||
        (map[y+1][x+3] == skull) ||(map[y-2][x+3] == skull) ||
        (map[y][x+2] == skull) || (map[y-1][x+2] == skull) ||
        (map[y+1][x+2] == skull) ||(map[y-2][x+2] == skull))
{
      --life;
      if(life > 0)
      {
        system("CLS");
        cout << "\n\n\n\t\tOooops!!!\n\t\tYou've picked up skull !";
        Sleep(1000);
        goto restart;
      }
  if(life==0)
  {
      goto lost;
```

```
    }
  }



  else if(map[y][x+3] != '#') {
    map[y][x] = ' ';
    map[y][x+1] = ' ';
    map[y][x-1] = ' ';
    map[y+1][x-1] = ' ';
    map[y+1][x+1] = ' ';
    map[y-1][x-1] = ' ';
    map[y-1][x+1] = ' ';
    map[y-1][x] = ' ';
    x += 3;
    map[y][x] = car1;
    map[y][x+1] = '|';
    map[y][x-1] = '|';
    map[y+1][x-1] = car3;
    map[y+1][x+1] = car3;
    map[y-1][x-1] = car2;
    map[y-1][x+1] = car2;
    map[y-1][x] = car3;
  }
}

//moves the car to the up
if(GetAsyncKeyState(VK_UP))
{
    if((map[y-3][x] == obstacle1) || (map[y-3][x] == obstacle2) || (map[y-3][x] == obstacle3) ||
(map[y-3][x] == '|') ||

       (map[y-3][x+1]  ==  obstacle1)  ||  (map[y-3][x+1]  ==  obstacle2)  ||  (map[y-3][x+1]  ==
obstacle3)|| (map[y-3][x+1] == '|') ||

       (map[y-3][x-1]  ==  obstacle1)  ||  (map[y-3][x-1]  ==  obstacle2)  ||  (map[y-3][x-1]  ==
obstacle3) || (map[y-3][x-1] == '|'))

    {

        --life;

      if(life > 0)

        {

            system("CLS");

            cout << "\n\n\n\t\tOooops!!!\n\t\tYou've crashed !";

            Sleep(1000);

            goto restart;

        }

      if(life==0)

        {

            goto lost;

        }

    }


    else if((map[y-3][x] == skull) || (map[y-3][x+1] == skull) ||

          (map[y-3][x-1] == skull) || (map[y-2][x] == skull) ||

          (map[y-2][x+1] == skull) || (map[y-2][x-1] == skull))

    {

        --life;

        if(life > 0)

        {

            system("CLS");

            cout << "\n\n\n\t\tOooops!!!\n\t\tYou've picked up skull !";
            Sleep(1000);
            goto restart;
        }
    if(life==0)
    {
        goto lost;
    }
  }

  else if(y>3) {
    map[y][x] = ' ';
    map[y][x+1] = ' ';
    map[y][x-1] = ' ';
    map[y+1][x-1] = ' ';
    map[y+1][x+1] = ' ';
    map[y-1][x-1] = ' ';
    map[y-1][x+1] = ' ';
    map[y-1][x] = ' ';
    y -= 3;
    map[y][x] = car1;
    map[y][x+1] = '|';
    map[y][x-1] = '|';
    map[y+1][x-1] = car3;
    map[y+1][x+1] = car3;
    map[y-1][x-1] = car2;
    map[y-1][x+1] = car2;
    map[y-1][x] = car3;
  }
}
```

```
   //moves the car to the down
   if(GetAsyncKeyState(VK_DOWN))
   {
     if((map[y+3][x] == obstacle1) || (map[y+3][x] == obstacle2) || (map[y+3][x] == obstacle3)
||
        (map[y+3][x+1] == obstacle1) || (map[y+3][x+1] == obstacle2) || (map[y+3][x+1] ==
obstacle3)||
        (map[y+3][x-1] == obstacle1) || (map[y+3][x-1] == obstacle2) || (map[y+3][x-1] ==
obstacle3))
     {
          --life;
       if(life > 0)
       {
          system("CLS");
          cout << "\n\n\n\t\tOoooops!!!\n\t\tYou've crashed !";
          Sleep(1000);
          goto restart;
       }
       if(life==0)
       {
          goto lost;
       }
     }

   else if((map[y+3][x] == skull) || (map[y+3][x+1] == skull) ||
          (map[y+3][x-1] == skull) || (map[y+2][x] == skull) ||
          (map[y+2][x+1] == skull) || (map[y+2][x-1] == skull))
     {
          --life;
        if(life > 0)
        {
            system("CLS");
            cout << "\n\n\n\t\tOooops!!!\n\t\tYou've picked up skull !";
            Sleep(1000);
            goto restart;
        }
    if(life==0)
    {
        goto lost;
    }
  }

  else if(y<=16) {
    map[y][x] = ' ';
    map[y][x+1] = ' ';
    map[y][x-1] = ' ';
    map[y+1][x-1] = ' ';
    map[y+1][x+1] = ' ';
    map[y-1][x-1] = ' ';
    map[y-1][x+1] = ' ';
    map[y-1][x] = ' ';
    y += 3;
    map[y][x] = car1;
    map[y][x+1] = '|';
    map[y][x-1] = '|';
    map[y+1][x-1] = car3;
    map[y+1][x+1] = car3;
    map[y-1][x-1] = car2;
```

```
        map[y-1][x+1] = car2;

        map[y-1][x] = car3;

      }

    }

    //checks if the car crashed

    if((map[y-2][x] == obstacle2 && map[y-2][x+1] == obstacle3) || (map[y-2][x-1] ==
obstacle2  && map[y-2][x] == obstacle3) ||

       (map[y-2][x+1] == obstacle2 && map[y-2][x+2] == obstacle3) || (map[y-2][x-2] ==
obstacle2 && map[y-2][x-1] == obstacle3)

        || (map[y-2][x+2] == obstacle2 && map[y-2][x+3] == obstacle3))

      {

        --life;

        if(life > 0)

        {

            system("CLS");

            cout << "\n\n\n\t\tOoooops!!!\n\t\tYou've crashed !";

            Sleep(1000);

            goto restart;

        }

      }


    if(life == 0)

    {

    lost:

        //check if the high score has been beaten

        if(points > hscore)

    {

        ofstream outFile;

        outFile.open("H.txt");

        outFile << points;
    outFile.close();


    system("CLS");
    cout << "\n\n\n\t\t   congratulations!!!\n\t\tYour new High score is " << points <<endl;
    Sleep(2500);
}
    else
    {
        system("CLS");
    cout << "\n\n\n\t\tYour score is " << points <<endl;
    Sleep(2500);
    }


  EndScreen();
  Sleep(1500);
  return ;
}
//checks if the player picked up cash
if(map[y-2][x] == cash || map[y-2][x-1] == cash || map[y-2][x+1] == cash) {


  map[y-2][x] = ' ';
  map[y-2][x-1] = ' ';
  map[y-2][x+1] = ' ';
  ++cashpickedup;
  points += 20;
  q = 0;
  p = rand() % 15 + 2;
}
```

```
   else if(map[y-1][x] == cash || map[y-1][x-1] == cash || map[y-1][x+1] == cash )
  {
     map[y-1][x] = ' ';
     map[y-1][x-1] = ' ';
     map[y-1][x+1] = ' ';
     ++cashpickedup;
     points += 20;
     q = 0;
     p = rand() % 15 + 2;
   }

   else if(map[y][x] == cash || map[y][x-1] == cash || map[y][x+1] == cash )
{
     map[y][x] = ' ';
     map[y][x-1] = ' ';
     map[y][x+1] = ' ';
     ++cashpickedup;
     points += 20;
     q = 0;
     p = rand() % 15 + 2;
   }

   else if(map[y+1][x] == cash || map[y+1][x-1] == cash || map[y+1][x+1] == cash)
{
     map[y+1][x] = ' ';
     map[y+1][x-1] = ' ';
     map[y+1][x+1] = ' ';
     ++cashpickedup;
     points += 20;
```

```
    q = 0;
    p = rand() % 15 + 2;
  }
  //generates the cash
  else if(q > 20)
 {
    q = 0;
    p = rand() % 15 + 2;
  }
  //does so the cash doesnt appear next to the obstacle
  if(a > 8)
 {
     ++cashcheck;
  }
  //places the cash

  if(points % 3 == 0)
  {
      if(cashcheck)
    {
      map[q][p] = ' ';
      ++q;
      map[q][p] = cash;
     }
  }
  else if(map[q][p]==cash)
  {
      map[q][p] = ' ';
      ++q;
      map[q][p] = cash;
```

```
}

//checks if the player picked up skull
if((map[y-2][x+1]==skull) || (map[y-2][x]==skull) || (map[y-2][x-1]==skull) ||
   (map[y-1][x+1]==skull) || (map[y-1][x]==skull) || (map[y-1][x-1]==skull) ||
   (map[y][x+1]==skull) || (map[y][x]==skull) || (map[y][x-1]==skull))
{
    --life;
    if(life > 0)
    {
        system("CLS");
        cout << "\n\n\n\t\tOooops!!!\n\t\tYou've picked up skull !";
        Sleep(1000);
        goto restart;
    }
    if(life==0)
    {
        goto lost;
    }
}

//generates the skull
else if(k > 20)
{
    k = 0;
    l = rand() % 15 + 2;
}

//does so the skull does not appear next to the obstacle
if(a > 4)
```

```
{
   ++skullcheck;
}

//places the skull
if(points % 4 !=0)
{
if(skullcheck)
{
   map[k][l] = ' ';
   ++k;
   map[k][l] = skull;
}
}

else if(map[k][l]==skull)
{
   map[k][l] = ' ';
   ++k;
   map[k][l] = skull;
}

//increase points
if((y+1)==(a-2))
  {
   points += 10;
  }

//speeds up the obstacles each time the player gets another 100 points
```

```
if(points > 100 && points <=200)
{
   speed = 140;
   level = 2;
}
else if(points > 200 && points <=300)
{
   speed = 120;
   level = 3;
}
else if(points > 300 && points <=400)
{
   speed = 100;
   level = 4;
}
else if(points > 400 && points <=500)
{
   speed = 80;
   level = 5;
}
else if(points > 500 && points <=600)
{
   speed = 60;
   level = 6;
}
else if(points > 600 && points <=700)
{
   speed = 40;
   level = 7;
```

```
    }
    else if(points > 700 && points <=800)
    {
        speed = 20;
        level = 8;
    }
    else if(points > 800)
    {
        speed = 10;
        level = 9;
    }
    Sleep(speed);
    }
}

void AboutUs()
{
    system("CLS");

    cout << "\n\n\n                              Hello !!!\n\n\n";
    cout << "                          CAR PERSIST 2016\n";
    cout << "                               V1.0.01\n";
    cout << "            (c)International Islamic University Malaysia\n";
    cout << "                         All Rights Reserved\n\n\n";
    cout << "                             CREDITS\n\n\n";
    cout << "                     Creative Head & Programmer\n\n"
         << "                    MAHFUZEALAHI NOMAN(1515803)\n\n\n";
    cout << "                            Programmers\n\n"
         << "                   ABID EBNA SAIF UTSHA(1433527)\n"
        << "                    TANVIR AHMED FORAZI(1432511)\n"
        << "                       TANVIR HOSSAN(1437699)\n"
        << "                 MD. ABU HAMED MOHIUDDIN(1428315)\n\n\n";
    cout << "                          Supervised By\n\n"
        << "                     Dr. AFIDALINA BINTI TUMIAN\n\n\n";
    cout << "                       THANK YOU FOR PLAYING"<<endl;

    Sleep(6000);
}

void Instruction()
{
    system("CLS");
    cout << "\n\n\n                       Welcome to CAR PERSIST\n\n\n";
    cout << "    Game Instruction\n";
    cout << "    ================\n\n\n";
    cout << "  -> use the arrow keys to move the car.\n\n";
    cout << "  -> You have to avoid cars coming from another direction.\n\n";
    cout << "  -> You will get points whenever you successfully pass the oncoming car.\n";
    cout << "     The sign '$' will be provided bonus points.\n\n";
    cout << "  -> Here you are provided with two lives. Your life will decrease if you\n";
    cout << "     crash with car or take the '@' object.\n\n";

    Sleep(4000);
}

void EndScreen()
{
    system("COLOR C");
```

```
            system("CLS");
    cout<<"\n\n\n\n\n\n\n";
            cout<<"\t***********************************************************************\n\n";
    cout<<"\t           |||||||     ||||||||   |||||  |||||   ||||||| "<<endl;
    cout<<"\t           |||         |||   |||   ||| |||| |||   |||      "<<endl;
    cout<<"\t           |||  |||||   ||||||||   |||  ||  |||   |||||    "<<endl;
    cout<<"\t           |||     ||   |||   |||   |||      |||   |||      "<<endl;
    cout<<"\t           |||||||||   |||   |||   |||      |||   |||||||  "<<endl<<endl<<endl;
    cout<<"\t            |||||       |||       |||   |||||||    |||||||           "<<endl;
    cout<<"\t          ||     ||      |||     |||    |||         |||   |||            "<<endl;
    cout<<"\t         ||       ||      |||   |||     ||||||      |||||||            "<<endl;;
    cout<<"\t          ||     ||        ||| |||      |||         |||   |||              "<<endl;
    cout<<"\t            |||||           |||        |||||||    |||    |||             \n"<<endl;

cout<<"\t***********************************************************************\n\n";


    return;
}
```