```
#include<stdio.h>
#include<fstream.h>
#include<iostream.h>
#include<conio.h>
#include<graphics.h>
#include<stdlib.h>
#include<dos.h>
#include<bios.h>
#include<time.h>
#define RIGHT 19712
#define LEFT 19200
#define UP 18432
#define DOWN 20480
//bios.h is for taking a keystroke(up,down etc) to move the car

void *car_1,*car_traffic,*road_divider,*tree;  // pointers to void have
         //been created.  They will store the image in the heap.

int poly[7],area;  //these are variables for creating and savings images

int carArr[6][3];    //the screen is a 2-D array
long int score = 0; //the score for the game

int temp,num,flag=0; //for the main menu

void drawPath()    //to draw the background and road
{
 setcolor(10);
 setfillstyle(1,10);
 bar(440,0,550,480);                   //grass
 setcolor(9);
 setfillstyle(9,9);
 bar(100,0,234,480);                    //water

putimage(310,0,road_divider,1);
putimage(370,0,road_divider,1);
putimage(440,0,tree,1);
 setcolor(4);
 setfillstyle(11,4);
 bar(235,0,245,480);                   //road ends
 bar(430,0,438,480);
}

//this function will draw the car at given position and in given color
void drawCar( int row, int col, int last, void* test )
{
  //here (row, col) refers to the position where the car has go to
  // (r,c) is the old position.
  //before the execution of this function, the car is at (r,c) onscreen.
   int r = row,c = col;
   switch( last )
      {
     case LEFT  : c = col + 1;
              break;
     case RIGHT : c = col - 1;
              break;
     case UP    : r = row + 1;
              break;
     case DOWN  : r = row - 1;
              break;
      }

  setfillstyle( SOLID_FILL, BLACK ); // old position filled with black(erased)
  bar(260+ c*60,r*67,300+c*60,67+r*75);
  putimage(260+col*60,row*67,test,1); //car put in the new position
  carArr[row][col] = 1;
  carArr[r][c] = 0;
}

//this function will erases the last position of a row, with the column
//being the input
void eraseBottom( int c )
{
   int r = 5;
   setfillstyle( SOLID_FILL, BLACK ); // to erase
 bar(260+ c*60,r*67,300+c*60,67+r*75);
  carArr[r][c] = 0;
}

//this function tells us where we have to place the enemy cars
void enemy( int row, int col)
{ //
   if(  carArr[ row ][ col ] == 0 )    //it checks if there is a car where it is to
    {                                 //be put. If there isn't, it is moved there.
      if( row < 6 )
      {
    drawCar( row, col , DOWN, car_traffic );
      }
   }
    else if( carArr[ row ][ col ] == 1 ) //however, if there is a car at (row,col)
    { //then it means that we have crashed.
       setcolor(2);
      settextstyle(4,0,4);
      outtextxy(220,120,"GAME OVER");
      getch();
      return;
   }
}

//this function will instruct the action to be done according to the player's
//inout.
//Here speed will be increasing gradually when the score increases
void play()
{
  score=0; //resets score to zero at the start
 for(int i=0;i<6;i++)   //resets all positions to zero. thus there are
 {                     //no cars onscreen as of now.
   for(int j=0;j<3;j++)
   carArr[i][j]=0;
 }
   unsigned int seedval;
   time_t t;
   seedval=(unsigned)time(&t);
   srand(seedval);
   int key;
   int d;
   int row = 5, col = 1;
   int r = row,c = col;
   unsigned int last;
   int ctr1 = 0, ctr2 = -2,ctr3=-4;
   int col1, col2, col3,col4;
   col1 = rand() % 3;
   col2 = rand() % 3;
   col3 = rand() % 3;
   col4 = rand() % 3;
//at any given time, three enemy cars moving onscreen
//corresponding to each car we have variables ctr and col
//giving the row and column position respectively
   drawCar( row, col, RIGHT, car_1 ); //player car is put on the grid

   while( key != 283 )
   {
      while( bioskey( 1 ) == 0 )
      {
     while( kbhit() == 0 )
     {
        gotoxy(1,5);
        printf("Score : %lu", score);
        if( ctr1 < 6 )   //ctr1 is the position of the lead car
        {//while it is onscreen (ie. ctr1<6), we move all the cars
        //downwards by one unit
           enemy( ctr1 , col1 );
           enemy( ctr2 , col2 );
           enemy( ctr3 , col3 );
           if( ctr1 == 5 ) //the lead car is onscreen but has reached
           {//the end. so a new car is put on screen
          enemy( -1 , col4 );
           }
        }
         else //each time the lead car goes offscreen, the score
         {//increases.
           score += 100;
           gotoxy(1,5);
           printf("Score : %lu", score);
           eraseBottom(col1); //as the lead car has gone off screen, the
           ctr1 = ctr2;//car just before the lead car now takes the lead
           ctr2 = ctr3;//the controls for the cars are reassigned
           ctr3 = 0;
           col1 = col2;
           col2 = col3;
           col3 = col4;
           col4 = rand() % 3; //this randomises the column in which
           //the enemy cars will appear
           enemy( ctr1 , col1 );
           enemy( ctr2 , col2 );
           enemy( ctr3 , col3 );
        }
        ctr1++;  //the enemy cars are moved down by one unit
        ctr2++;
        ctr3++;

       d= score/300;    //the delay decreased after every 300 points
       if((750-50*d)>200)
       delay(750-50*d);
       else
       delay(200); //however a delay of 200 is quite challenging enough

     }
      }
      key = bioskey( 0 );  //this takes in a input
    //if no key is hit, then it returns 0
      switch( key )  //it switches the input. Note that if no key is hit
      {//then key=0 and as there is no such case, no action is taken
     case LEFT  : c = col - 1;
              last = LEFT;
              break;
     case RIGHT : c = col + 1;
              last = RIGHT;
              break;
     case UP    : r = row - 1;
              last = UP;
              break;
     case DOWN  : r = row + 1;
              last = DOWN;
              break;
      }
      if( c < 0 )   c = 0; //these four lines ensure that the position of our
      if( r < 0 )   r = 0; //car remains on the grid.However it is to be noted
      if( c > 2)    c = 2; //that if any of these lines are invoked, it means
      if( r > 4 )   r = 5;//we have hit the side

      if( carArr[r][c] != 0 )  //checks if we have hit the sides
      {
       setcolor(2);
      settextstyle(4,0,4);
      outtextxy(220,120,"GAME OVER");
      getch();
      return;
      }
      drawCar(r,c,last, car_1);
      row = r;  //the previous position of our car becomes its
      col = c; //current position
   }
}

void create()    //it creates various graphic objects and stores them
```

```
{ //as images for later use.
   setcolor(7);             //car_1
 setfillstyle(1,7);
 pieslice(40,40,0,360,20);
 bar(23,50,57,70);         //body
 fillellipse(40,75,20,15);
 setcolor(0);
 setfillstyle(1,0);
 bar(20,87,60,95);
 setcolor(9);
 setfillstyle(1,9);
 fillellipse(40,47,12,5);
 poly[0]=28;poly[1]=47;
 poly[2]=52;poly[3]=47;
 poly[4]=50;poly[5]=55;
 poly[6]=30;poly[7]=55;
 fillpoly(4,poly);         //front windshield
 poly[0]=28;poly[1]=75;
 poly[2]=52;poly[3]=75;
 poly[4]=50;poly[5]=75;
 poly[6]=30;poly[7]=75;
 fillpoly(4,poly);         //back windshield
 fillellipse(40,75,12,2);
 poly[0]=55;poly[1]=47;
 poly[2]=55;poly[3]=70;
 poly[4]=52;poly[5]=65;
 poly[6]=54;poly[7]=50;
 fillpoly(4,poly);        //right window
 poly[0]=25;poly[1]=47;
 poly[2]=25;poly[3]=70;
 poly[4]=28;poly[5]=65;
 poly[6]=26;poly[7]=50;
 fillpoly(4,poly);       //left window
 setcolor(12);
 setfillstyle(1,12);
 pieslice(30,24,40,180,3);      //front lamps
 pieslice(50,24,130,0,3);
 bar(28,86,52,87);   //back lamps
 area=imagesize(
 20,20,60,87);
 car_1=malloc(area);
 getimage(20,20,60,87,car_1);
 clearviewport();

 setcolor(12);                            //car_traffic
 setfillstyle(1,12);
 pieslice(40,40,0,360,20);
 bar(23,50,57,70);             //body
 fillellipse(40,75,20,15);
 setcolor(0);
 setfillstyle(1,0);
 bar(20,87,60,95);
 setcolor(8);
 setfillstyle(1,8);
```

```
fillellipse(40,47,12,5);
poly[0]=28;poly[1]=47;
poly[2]=52;poly[3]=47;
poly[4]=50;poly[5]=55;
poly[6]=30;poly[7]=55;
fillpoly(4,poly);      //front windshield
poly[0]=28;poly[1]=75;
poly[2]=52;poly[3]=75;
poly[4]=50;poly[5]=75;
poly[6]=30;poly[7]=75;
fillpoly(4,poly);       //back windshield
fillellipse(40,75,12,2);
poly[0]=55;poly[1]=47;
poly[2]=55;poly[3]=70;
poly[4]=52;poly[5]=65;
poly[6]=54;poly[7]=50;
fillpoly(4,poly);       //right window
poly[0]=25;poly[1]=47;
poly[2]=25;poly[3]=70;
poly[4]=28;poly[5]=65;
poly[6]=26;poly[7]=50;
fillpoly(4,poly);       //left window
setcolor(8);
setfillstyle(1,8);
pieslice(30,24,40,180,3);      //front lamps
pieslice(50,24,130,0,3);
bar(28,86,52,87);      //back lamps
area=imagesize(
20,20,60,87);
car_traffic=malloc(area);
getimage(20,20,60,87,car_traffic);
clearviewport();

setcolor(7);                       //road divider
setfillstyle(1,7);
for(int i=0;i<=400;i+=80)
{
 bar(0,i,5,i+40);
}
area=imagesize(0,0,5,440);
road_divider=malloc(area);
getimage(0,0,5,440,road_divider);
clearviewport();

for(i=0;i<=400;i+=100)                     //tree
{
 setcolor(8);
 setfillstyle(1,8);
 poly[0]=20;poly[1]=0+i;
 poly[2]=30;poly[3]=10+i;
 poly[4]=10;poly[5]=10+i;
 fillpoly(3,poly);
 poly[0]=20;poly[1]=5+i;
 poly[2]=35;poly[3]=20+i;
```

```
  poly[4]=5;poly[5]=20+i;
  fillpoly(3,poly);
  poly[0]=20;poly[1]=10+i;
  poly[2]=40;poly[3]=30+i;
  poly[4]=0;poly[5]=30+i;
  fillpoly(3,poly);
  setcolor(9);
  setfillstyle(1,12);
  bar(15,30+i,25,35+i);
 }
 area=imagesize(0,0,40,435);
 tree=malloc(area);
 getimage(0,0,40,435,tree);
 clearviewport();
}

void create2() //it creates the same objects as void create() but this time
{//the size of the objects is bigger
 setcolor(7);                            //car_1
 setfillstyle(1,7);
 pieslice(40,40,0,360,40);
 bar(5,60,75,100);                         //body
 fillellipse(40,110,40,30);
 setcolor(0);
 setfillstyle(1,0);
 bar(0,135,80,150);
 setcolor(9);
 setfillstyle(1,9);
 fillellipse(40,55,25,10);
 poly[0]=15;poly[1]=55;
 poly[2]=65;poly[3]=55;
 poly[4]=60;poly[5]=70;
 poly[6]=20;poly[7]=70;
 fillpoly(4,poly);                      //front windshield
 poly[0]=15;poly[1]=110;
 poly[2]=65;poly[3]=110;
 poly[4]=60;poly[5]=100;
 poly[6]=20;poly[7]=100;
 fillpoly(4,poly);                      //back windshield
 fillellipse(40,110,25,5);
 poly[0]=70;poly[1]=55;
 poly[2]=70;poly[3]=100;
 poly[4]=65;poly[5]=90;
 poly[6]=68;poly[7]=60;
 fillpoly(4,poly);                      //right window
 poly[0]=10;poly[1]=55;
 poly[2]=10;poly[3]=100;
 poly[4]=15;poly[5]=90;
 poly[6]=12;poly[7]=60;
 fillpoly(4,poly);                      //left window
 setcolor(12);
 setfillstyle(1,12);
 pieslice(20,8,40,180,7);              //front lamps
 pieslice(60,8,130,0,7);
 bar(15,132,65,135);                   //back lamps
 area=imagesize(0,0,80,135);
 car_1=malloc(area);
 getimage(0,0,80,135,car_1);
 clearviewport();
 setcolor(12);                             //car_traffic
 setfillstyle(1,12);
 pieslice(40,40,0,360,40);
 bar(5,60,75,100);                         //body
 fillellipse(40,110,40,30);
 setcolor(0);
 setfillstyle(1,0);
 bar(0,135,80,150);
 setcolor(8);
 setfillstyle(1,8);
 fillellipse(40,55,25,10);
 poly[0]=15;poly[1]=55;
 poly[2]=65;poly[3]=55;
 poly[4]=60;poly[5]=70;
 poly[6]=20;poly[7]=70;
 fillpoly(4,poly);                      //front windshield
 poly[0]=15;poly[1]=110;
 poly[2]=65;poly[3]=110;
 poly[4]=60;poly[5]=100;
 poly[6]=20;poly[7]=100;
 fillpoly(4,poly);                      //back windshield
 fillellipse(40,110,25,5);
 poly[0]=70;poly[1]=55;
 poly[2]=70;poly[3]=100;
 poly[4]=65;poly[5]=90;
 poly[6]=68;poly[7]=60;
 fillpoly(4,poly);                      //right windshield
 poly[0]=10;poly[1]=55;
 poly[2]=10;poly[3]=100;
 poly[4]=15;poly[5]=90;
 poly[6]=12;poly[7]=60;
 fillpoly(4,poly);                      //left screen
 setcolor(8);
 setfillstyle(1,8);
 pieslice(20,8,40,180,7);              //front lamps
 pieslice(60,8,130,0,7);
 bar(15,132,65,135);                   //back lamp
 setcolor(15);
 area=imagesize(0,0,80,135);
 car_traffic=malloc(area);
 getimage(0,0,80,135,car_traffic);
 clearviewport();
 }

class menu
{
    char name[20];
    int pass;  //password
    int hscore;
```

```
  public:

  void create()
  {
      cout<<"enter your name\n";
      gets(name);
      cout<<"create password\n";
      cin>>pass;
      cout<<"PRESS ANY KEY TO CONTINUE!!!\n";
  }

int login(int temp)
{
 if(temp==pass)
 return 1;
 return 0;
}

void score()
{
cout<<hscore;
}

int retscore()
{
return hscore;
}

void display()
{
cout<<"Name : ";
puts(name);
cout<<"Best Score : ";
cout<<hscore;
cout<<"\n";
}

void hiscore(int s)
{
if(s>hscore)
hscore=s;
}

int showpass()
{
return pass;
}

void sethigh()   //sets the highscore to -1, and is  used in the
{                //implementation of the leaderboard
hscore=-1;
}
```

```
};
menu hiscore[10];     //array of 10 objects created

void insert(menu s)
{
menu t; //to temporarily store an object
int p; //for temporarily storing password to compare with.
int temp; //for temporarily storing password to compare with.
p=s.showpass();
ifstream iff2;
ofstream off2;
iff2.open("car.txt",ios::in|ios::binary);
off2.open("temp.txt",ios::app|ios::binary);

while(!iff2.eof())
{
iff2.read((char*)&t,sizeof(t));
if(iff2.eof())
break;
temp=t.showpass();
if(temp==p)
off2.write((char*)&s,sizeof(s));
else
off2.write((char*)&t,sizeof(t));
}

iff2.close();
off2.close();
remove("car.txt");
rename("temp.txt","car.txt");
}

void main()
{
    ifstream iff;
    ofstream off;
    char name[20];
    int opt;  //for switch
    int i=0,j=0,k=0; //loop variables for the leaderboard
    menu select;  //object instance to hold data

    do
    {
        clrscr();
        cout<<
        "\n\n\t\tUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUSUS
        USUSUSUSUSUSUSUSUSUS\n";
        cout<<"\t\t\t\t\t\t\t\t\t\t\t\t\tSTX....WELCOME TO ROAD KILL....STX\n";
        cout<<
        "\t\tRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRSRS
        RSRSRSRSRSRSRSRS\n\n\n";
        cout<<"\t\tSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTX
        STXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTX\n";
        cout<<"\t\tGSGSGSGSGSGSGSGSGSGSGSGSGSGSGSGSGSGSGS MAIN MENU
```

```
GSGSGSGSGSGSGSGSGSGSGSGSGSGSGSGSGSGSGS\n";
cout<<"\t\tSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTX
STXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTXSTX\n\n";
cout<<"\n\t\t\tPlease select a kind of user:";
cout<<"\n\n\t\t\t\t1.Create
profile\n\n\t\t\t\t2.Login\n\n\t\t\t\t3.Leaderboard\n\n\t\t\t\t4.Play\n\n\t\t\t\t5.How
to play";
cout<<"\n\n\t\t\t\t6.Exit";
cout<<"\n\n\t\t\t\tDLE ";
cin>>opt;

 switch(opt)
{
    case 1:
    {     off.open("car.txt",ios::app|ios::binary);
            clrscr();
            cout<<"\n\n\n\t\t\t\t NEW USER\n";
            cout<<"\t\t\t\t    ********\n\n";
            select.create();
            off.write((char *)&select,sizeof(select));
            getch();
            off.close();
            break;
    }

    case 2:
    {       flag=0;
        clrscr();
        iff.open("car.txt",ios::in|ios::binary);
        cout<<"\n\n\t\t\tSTX***** EXISTING USER *****STX\n\n\t\t";
        cout<<"Enter password\n";
        cin>>temp;
        while(!iff.eof())
           {
           iff.read((char*)&select,sizeof(select));
           flag=select.login(temp);
           if(flag==1)
          { select.display();
        cout<<"Press 5 to play, 0 to go back\n";
        cin>>num;
        if(num==5)
        {
        clrscr();
        int gdriver=DETECT,gmode;
        initgraph(&gdriver,&gmode,"c:\\turboc3\\bgi");
        create();
        drawPath();
        play();
           select.hiscore(score);
           insert(select);
        closegraph();
        }
        cout<<"PRESS ANY KEY TO CONTINUE!!!\n";
        getch();
        break;
         }
           }
        if(!flag)
        cout<<"Invalid user\n";
        iff.close();
        getch();
        break;
    }

    case 3:
    { clrscr();
      int maxpass; //variable to store the password
      // of the guy with the highest score
      int maxscore; //stores the maximum score in an
             //iteration
      int tempscore; //holds a score temporarily
     cout<<"\n\n\t\t\tSTX***** HIGHSCORES *****STX\n\n\t\t";
     iff.open("car.txt",ios::in|ios::binary);
     while(!iff.eof())
     {
     iff.read((char*)&hiscore[i],sizeof(hiscore[i]));
     if(iff.eof())
     break;
     i++;
     }
     for(j=0;j<i;j++)
     {
      maxscore=hiscore[0].retscore();
      for(k=0;k<i;k++)
      {
       tempscore=hiscore[k].retscore();
       if(tempscore>maxscore)
        {
        maxscore=tempscore;
        flag=k;
        }
      }
     cout<<"\n";
     hiscore[flag].display();
     hiscore[flag].sethigh(); //the score has been set to -1.
     flag=0;     //now the highest score of this
     }//iteration wont interfere with the next iteration
     getch();
     i=0;  //resets player count to zero
     iff.close();
    }
    break;

    case 4:
    {clrscr();
    int gdriver=DETECT,gmode;
    initgraph(&gdriver,&gmode,"c:\\turboc3\\bgi");
    create();
```

```
            drawPath();
            play();
            closegraph();
            break;
            }

            case 5:
            { clrscr();
            int gdriver=DETECT,gmode;
            initgraph(&gdriver,&gmode,"c:\\turboc3\\bgi");
             create2();
             setfillstyle(1,8);
            floodfill(0,0,1);
            setcolor(2);
            settextstyle(4,0,5);
            outtextxy(200,0,"ROAD KILL");
            settextstyle(1,0,1);
            outtextxy(0,50,"HOW TO PLAY:-");
               putimage(0,100,car_1,1);
               outtextxy(150,120,"PLAYER:");
        outtextxy(150,140,"Use up and down to navigate along the road");
       outtextxy(150,290,"Use (<-) and (->) to switch lanes");
         outtextxy(200,400,"AND DON'T HIT THE TRAFFIC !");
          putimage(500,320,car_traffic,1);
            getch();
             clearviewport();
            closegraph();
             break;
             }

            case 6:
            {clrscr();
         cout<<"\n\n";
         cout<<"             _/_/_/_/_/  _/                            _/            \n";
         cout<<"                _/      _/_/_/      _/_/_/  _/_/_/    _/  _/         \n";
         cout<<"               _/      _/    _/  _/    _/  _/    _/  _/_/            \n";
         cout<<"              _/      _/    _/  _/    _/  _/    _/  _/  _/           \n";
         cout<<"             _/      _/    _/    _/_/_/  _/    _/  _/    _/          \n";
         cout<<"                                                                  \n";
         cout<<"                                                                  \n";
         cout<<"                                                                  \n";
         cout<<"                 _/     _/                                       \n";
         cout<<"                  _/  _/    _/_/    _/    _/                     \n";
         cout<<"                    _/    _/    _/  _/    _/                     \n";
         cout<<"                    _/    _/    _/  _/    _/                     \n";
         cout<<"                   _/      _/_/      _/_/_/                     \n\n";
         getch();
         exit(0);
    }
    getch();
        }
    }while(1);
}
```