Car Race Game

Aim: Developing car race game using concepts of computer graphics in C.

Theory:
Header Files Included:

1. **<stdio.h>**
   The C programming language provides many standard library functions for file input and output. These functions make up the bulk of the C standard library header <stdio.h>.

2. **<conio.h>**
   conio.h is a C header file used mostly by MS-DOS compilers to provide console input/output. It is not part of the C standard library or ISO C, nor is it defined by POSIX.

3. **<alloc.h>**
   C dynamic memory allocation refers to performing manual memory management for dynamic memory allocation in the C programming language via a group of functions in the C standard library, namely malloc, realloc, calloc and free.
   The C++ programming language includes these functions for compatibility with C; however, the operators new and delete provide similar functionality.

4. **<graphics.h>**
   C graphics using graphics.h functions or WinBGIM (Windows 7) can be used to draw different shapes, display text in different fonts, change colors and many more. Using functions of graphics.h in turbo c compiler you can make graphics programs, animations, projects and games. You can draw circles, lines, rectangles, bars and many other geometrical figures. You can change their colors using the available functions and fill them.

5. **<stdlib.h>**
   stdlib.h is the header of the **general purpose standard library** of C programming language which includes functions involving memory allocation, process control, conversions and others. It is compatible with C++ and is known as cstdlib in C++. The name "stdlib" stands for "standard library".

6. **<time.h>**
   The C date and time functions are a group of functions in the standard library of the C programming language implementing date and time manipulation operations.[1] They provide support for time acquisition, conversion between date formats, and formatted output to strings.

7. **<string.h>**

   string.h is the header in the C standard library for the C programming language which contains macro definitions, constants, and declarations of functions and types used not only for string handling but also various memory handling functions; the name is thus something of a misnomer.

Functions:

1. **outtextxy()**

   These functions are used to display text on the screen in graphics mode. Function outtext displays text at the current position while outtextxy displays text at the specified coordinates (x, y) on the screen.

   Syntax:
   void outtext(char *string);
   void outtextxy(int x, int y, char *string);
   where x,y are coordinates of the point and , third argument contains the address of the string to be displayed.

2. **setcolor()**

   The header file graphics.h contains setcolor() function which is used to set the current drawing color to the new color.

Syntax :

void setcolor(int color);

In Graphics, each color is assigned a number. Total number of colors available are 16.

3. **line()**

   Line function is used to draw a line from a point(x1,y1) to point(x2,y2) i.e. (x1,y1) and (x2,y2) are end points of the line.

   Syntax:

   void line(int x1, int y1, int x2, int y2);

4. **delay()**

   Delay function is used to suspend execution of a program for a particular time.

   Syntax:-

   void delay(unsigned int);

   Here unsigned int is the number of milliseconds (remember one second = 1000 milliseconds).

5. **rectangle()**

   Syntax:
   void rectangle(int left, int top, int right, int bottom);

   rectangle function is used to draw a rectangle. Coordinates of left top and right bottom corner are required to draw the rectangle. left specifies the X-coordinate of top left corner, top specifies the Y-coordinate of top left corner, right specifies the X-coordinate of right bottom corner, bottom specifies the Y-coordinate of right bottom corner.

6. **gotoxy()**

   gotoxy function places cursor at a desired location on screen i.e. we can change cursor position using gotoxy function.

   Syntax :

   ```
   void gotoxy( int x, int y);
   ```

   where (x, y) is the position where we want to place the cursor.

7. **settextstyle()**

   Settextstyle function is used to change the way in which text appears, using it we can modify the size of text, change direction of text and change the font of text.

   Syntax:

   ```
   void settextstyle( int font, int direction, int charsize);
   ```

8. **getmaxx()**

   getmaxx function returns the maximum X coordinate for current graphics mode and driver.

   Syntax:

   ```
   int getmaxx();
   ```

9. **getmaxy()**

   getmaxy function returns the maximum Y coordinate for current graphics mode and driver.

   Syntax:

   ```
   int getmaxy();
   ```

10. cleardevice()

Syntax:

void cleardevice();

cleardevice function clears the screen in graphics mode and sets the current position to (0,0). Clearing the screen consists of filling the screen with current background color.

11. setfillstyle()

setfillstyle function sets the current fill pattern and fill color.

Syntax :

void setfillstyle(int pattern, int color);

12. fillpoly()

Fillpoly function draws and fills a polygon. It require same arguments as drawpoly.

Syntax:

void drawpoly(int num, int *polypoints);
Fillpoly fills using current fill pattern and color which can be changed using setfillstyle.

13. getimage()

getimage function saves a bit image of specified region into memory, region can be any rectangle.

Syntax:

void getimage(int left, int top, int right, int bottom, void *bitmap);

getimage copies an image from screen to memory. Left, top, right, and bottom define the area of the screen from which the rectangle is to be copied, bitmap points to the area in memory where the bit image is stored.

14. putimage()

putimage function outputs a bit image onto the screen.

Syntax:

void putimage(int left, int top, void *ptr, int op);

putimage puts the bit image previously saved with getimage back onto the screen, with the upper left corner of the image placed at (left, top). ptr points to the area in memory where the source image is stored. The op argument specifies a operator that controls how the color for each destination pixel on screen is computed, based on pixel already on screen and the corresponding source pixel in memory.

15.srand()

The function srand() is used to initialize the pseudo-random number generator by passing the argument seed.
Often the function time is used as input for the seed.

If the seed is set to 1 then the generator is reinitialized to its initial value. Then it will produce the results as before any call to rand and srand.

Syntax:

void srand (unsigned int seed);

16.rand()

The C library function int rand(void) returns a pseudo-random number in the range of 0 to RAND_MAX.

RAND_MAX is a constant whose default value may vary between implementations but it is granted to be at least 32767.

Syntax:

int rand(void);

17. sound()

Sound function produces the sound of a specified frequency. Used for adding music to c program, try to use some random values in loop, vary delay and enjoy.

Syntax:

void sound(unsigned frequency);

18. nosound()

nosound function turn off the PC speaker.

Syntax :

void nosound();

code:

```
#include <stdio.h>
#include <conio.h>
#include <alloc.h>
#include <graphics.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>
#define maxx 640
#define maxy 480
int max_x, max_y;
int road[] = {150, 0, maxx-150, 0, maxx-150, maxy, 150, maxy, 150, 0};
int road_sides[] = {120, 0, maxx-120, 0, maxx-120, maxy, 120, maxy, 120, 0};
int rect3[]={150+103,1,150+103+15,1,150+103+15,50,150+103,50,150+103,1};
int     rect4[]={maxx-150-119,1,maxx-150-119+15,1,maxx-150-119+15,50,maxx-
150-119,50,maxx-150-119,1};
int car_size[] = {165, 390, 170, 375, 233, 375, 238, 390, 238, 450, 165, 450, 165,
390};
int wheels[] = {155, 390, 165, 390, 165, 410, 155, 410, 155, 390};
int fullscreen[] = {0, 0, 639, 0, 639, 479, 0, 479, 0, 0};
int cover[] = {maxx-110, 50, maxx, 50, maxx, 100, maxx-110, 100, maxx-110, 50};
void game();
void loading()
{
    int i,a, per = 0;
     for(i=0;i<=200;i++)
    {
     setcolor(7);
     outtextxy(300,244,"LOADING...");
     setcolor(WHITE);
     line(224+i,238,224+i,256);
     setcolor(WHITE);
     rectangle(224,238,425,256);
     delay(25);
     gotoxy(55,16);
     setcolor(RED);
     if(i % 2 == 0)
     {
          printf("%d%",per);
          per++;
     }

     }
     settextstyle(1, 0, 1);
     setcolor(WHITE);
     outtextxy(getmaxx()/2-100, getmaxy()/2+20, "Press any key to start");
     getch();
     cleardevice();
     game();
 }

void exitScreen(int displayscore)
{
          char op, string[100];
          sprintf(string, "%d", displayscore);
          outtextxy(getmaxx()/2-100, getmaxy()/2-30, "Your score : ") ;
          outtextxy(getmaxx()/2+30, getmaxy()/2-30, string);
          outtextxy(getmaxx()/2-100, getmaxy()/2, "Do you want to play again?");
          outtextxy(getmaxx()/2-100, getmaxy()/2+30, "Y: Yes   N: No");
          op = getch();
          if(op=='Y'|| op=='y')
          {
                    cleardevice();
```

```
                game();
        }
        else if(op=='N' || op=='n')
        {
                cleardevice();
                exit(1);
        }else if(op==27)
        {
                cleardevice();
                exit(1);
        }
}
void game()
{
        int x=1, position=1, interval=75, enemy=1, carspeed=5, score=0;
        int displayscore=0, value1=0, value2=0;
        char ch;
        char string[100];
        void *buff1; // for road strips
        void *buff2; //for car wheels
        void *buff3; //for player car
        void *buff4; //for enemy car wheels
        void *buff5; //for enemy car
        max_x = getmaxx();
        max_y = getmaxy();
        cleardevice();
        /* setting up white strips */
        setfillstyle(SOLID_FILL,WHITE);
        fillpoly(5,rect3);//draw strips on road
        fillpoly(5,rect4);//as above
        buff1=malloc(imagesize(150+103,1,150+103+15,50)); //to store left strip on
memory
        getimage(150+103,1,150+103+15,50,buff1);//stored left strip on memory
        putimage(150+103,1,buff1,XOR_PUT);//strips on road deleted from the
memory//
        cleardevice();
        /* setting up the player car */
        setfillstyle(SOLID_FILL, 8);
        fillpoly(5, road);
        setcolor(RED);
        setfillstyle(SOLID_FILL, RED);
        fillpoly(7, car_size);
        setfillstyle(SOLID_FILL, BLACK);
```

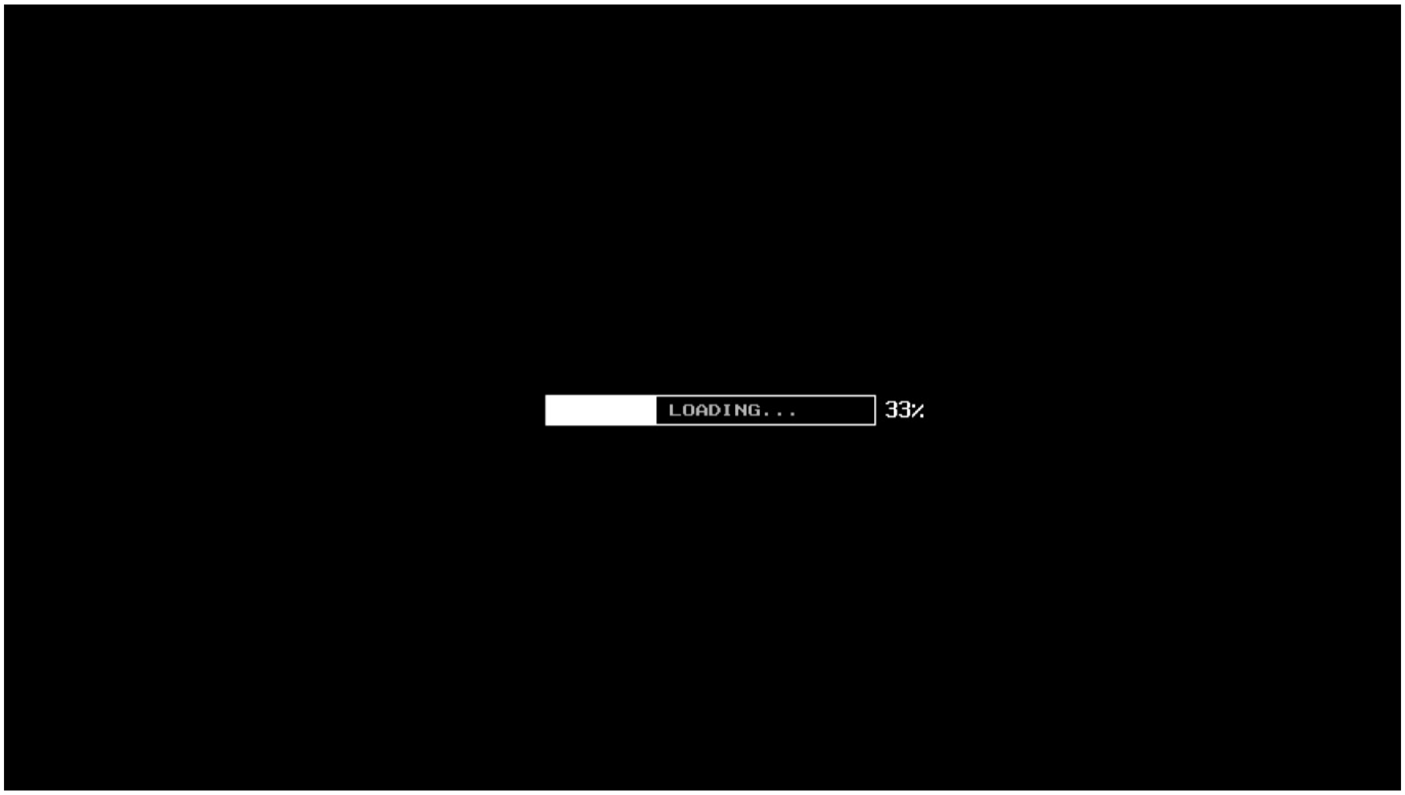
```
fillpoly(5, wheels);
buff2 = malloc(imagesize(155, 390, 165, 410));
getimage(155, 390, 165, 410, buff2);
putimage(155, 390, buff2, XOR_PUT);
putimage(155, 390, buff2, COPY_PUT);
putimage(238, 390, buff2, COPY_PUT);
putimage(155, 428, buff2, COPY_PUT);
putimage(238, 428, buff2, COPY_PUT);
buff3 = malloc(imagesize(155, 375, 248, 450));
getimage(155, 375, 248, 450, buff3);
putimage(155, 375, buff3, XOR_PUT);
cleardevice();
/* setting up the enemy cars */
setfillstyle(SOLID_FILL, 8);
fillpoly(5, road);
setcolor(YELLOW);
setfillstyle(SOLID_FILL, YELLOW);
fillpoly(7, car_size);
setfillstyle(SOLID_FILL, BLACK);
fillpoly(5, wheels);
buff4 = malloc(imagesize(155, 390, 165, 410));
getimage(155, 390, 165, 410, buff4);
putimage(155, 390, buff4, XOR_PUT);
putimage(155, 390, buff4, COPY_PUT);
putimage(238, 390, buff4, COPY_PUT);
putimage(155, 428, buff4, COPY_PUT);
putimage(238, 428, buff4, COPY_PUT);
buff5 = malloc(imagesize(155, 375, 248, 450));
getimage(155, 375, 248, 450, buff5);
putimage(155, 375, buff3, XOR_PUT);
cleardevice();
setfillstyle(SOLID_FILL, BLUE);
fillpoly(5, fullscreen);
setfillstyle(SOLID_FILL, 7);
fillpoly(5, road_sides);
setfillstyle(SOLID_FILL, 8);
fillpoly(5, road);
setcolor(WHITE);
srand(time(NULL));
while(value1==value2)
{
        value1 = rand()%3+1;
        value2 = rand()%3+1;
```

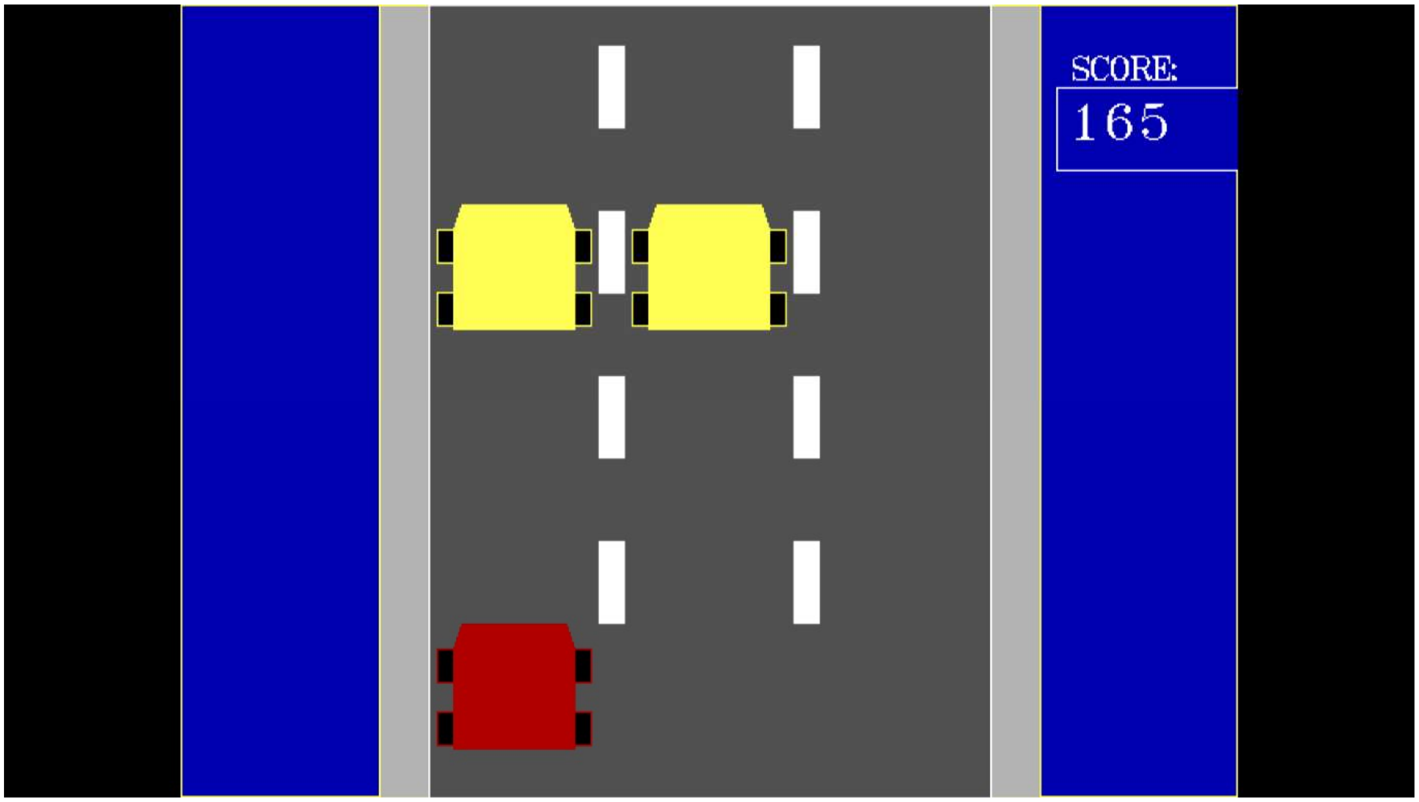
```
}
while(1)
{

        sound(100);
        if(score<=5)
                carspeed=15;
        else if(score>5 && score<=10)
                carspeed=20;
        else if(score>10 && score<=15)
                carspeed=20;
        else if(score>15 && score<=20)
                carspeed=25;
        else if(score>20 && score<=30)
                carspeed=30;
        else if(score>30)
                carspeed=40;

        if(value1!=1 && value2!=1)
        {
                enemy = enemy + carspeed;
                if((enemy>=300) && (position==2 || position==3))
                {
                        sound(50);
                        delay(100);
                        nosound();
                        cleardevice();
                        exitScreen(displayscore);
                }
                putimage(273, enemy, buff5, COPY_PUT);
                putimage(391, enemy, buff5, COPY_PUT);
                if(enemy>=450)
                {
                        enemy=1;
                        score = score+5;
                        srand(time(NULL));
                        do
                        {
                                value1 = rand()%3+1;
                                value2 = rand()%3+1;
                        }while(value1==value2);
                }
        }
```

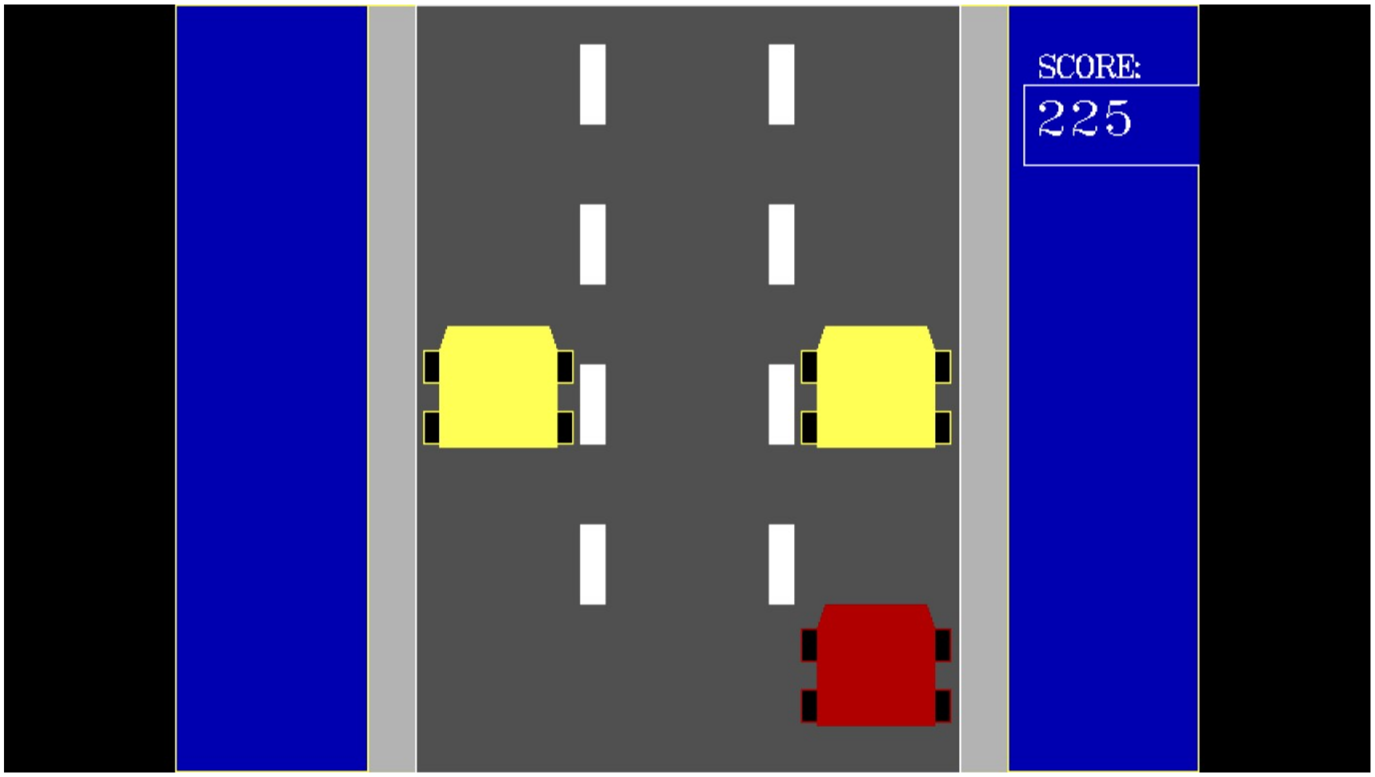
```
if(value1!=2 && value2!=2)
{
	enemy = enemy + carspeed;
	if(enemy>=300 && (position==1 || position==3))
	{
		sound(50);
		delay(100);
		nosound();
		cleardevice();
		exitScreen(displayscore);
	}
	putimage(155, enemy, buff5, COPY_PUT);
	putimage(391, enemy, buff5, COPY_PUT);
	if(enemy>=450)
	{
		enemy=1;
		score = score+5;
		srand(time(NULL));
		do
		{
			value1 = rand()%3+1;
			value2 = rand()%3+1;
		}while(value1==value2);
	}
}
if(value1!=3 && value2!=3)
{
	enemy = enemy + carspeed;
	if(enemy>=300 && (position==2 || position==1))
	{
		sound(50);
		delay(100);
		nosound();
		cleardevice();
		exitScreen(displayscore);
	}
	putimage(155, enemy, buff5, COPY_PUT);
	putimage(273, enemy, buff5, COPY_PUT);
	if(enemy>=450)
	{
		enemy=1;
		score = score+5;
		srand(time(NULL));
```

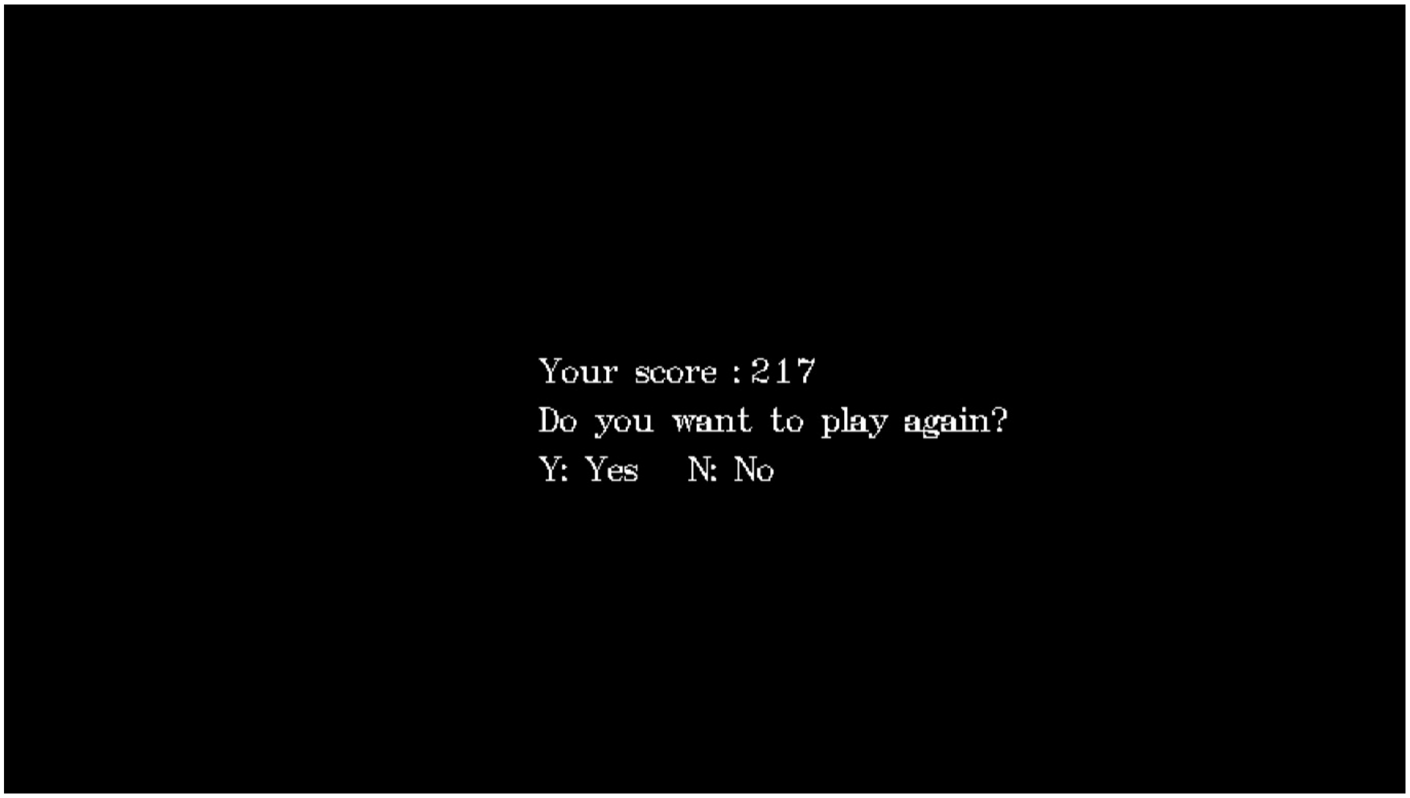
```
                    do
                    {
                            value1 = rand()%3+1;
                            value2 = rand()%3+1;
                    }while(value1==value2);
            }
        }
        putimage(150+103,(x-1)*25,buff1,COPY_PUT);
        putimage(150+103+15+103,(x-1)*25,buff1,COPY_PUT);
        putimage(150+103,100+(x-1)*25,buff1,COPY_PUT);
        putimage(150+103+15+103,100+(x-1)*25,buff1,COPY_PUT);
        putimage(150+103,200+(x-1)*25,buff1,COPY_PUT);
        putimage(150+103+15+103,200+(x-1)*25,buff1,COPY_PUT);
        putimage(150+103,300+(x-1)*25,buff1,COPY_PUT);
        putimage(150+103+15+103,300+(x-1)*25,buff1,COPY_PUT);
        if(x!=5)
        {
                delay(interval);
                putimage(150+103, 400+(x-1)*25, buff1, COPY_PUT);
                sprintf(string, "%d", displayscore);
                displayscore++;
                putimage(150+103+15+103,        400+(x-1)*25,        buff1,
COPY_PUT);
                setfillstyle(SOLID_FILL, BLUE);
                fillpoly(5, cover);
        }
        if(x==5)
        {
                delay(interval);
                setfillstyle(SOLID_FILL, BLUE);
                fillpoly(5, cover);
                putimage(150+103, 1, buff1, COPY_PUT);
                putimage(150+103+15+103, 1,buff1, COPY_PUT);
                sprintf(string, "%d", displayscore);
                displayscore++;

        }
        setfillstyle(SOLID_FILL, 8);
        fillpoly(5, road);
        x++;
        settextstyle(1, DEFAULT_FONT, 4);
        outtextxy(maxx-100, 50, string);
        if(x==6)x=1;
```

```
                if(position==1)
                putimage(155, 375, buff3, COPY_PUT);
                else if(position==2)
                putimage(273, 375, buff3, COPY_PUT);
                else if(position==3)
                putimage(391, 375, buff3, COPY_PUT);
                if(kbhit())
                {
                        ch=getch();
                        switch(ch)
                        {
                                case 75: //left arrow key
                                        position--;
                                        break;
                                case 77:
                                        position++;
                                        break;
                                case 27:
                                        nosound();
                                        exit(1);
                                        break;
                                default:
                                        break;
                         }
                }
                if(position>3)position=3;
                else if(position<1)position=1;
                settextstyle(1, DEFAULT_FONT, 2);
                outtextxy(maxx-100, 25, "SCORE: ");

        }
}



void main()
{
        int gd=DETECT, gm;
        initgraph(&gd, &gm, "..//BGI");
        loading();

}
```


LOADING...
33%


SCORE:
165





Your score : 217
Do you want to play again?
Y: Yes   N: No