C Project Report

STAR WARS

Overall approach & Outline

- ✓ Our C Project is based on Game namely STAR WARS.
- ✓ Star Wars game outline- A young boy from "Circulla" sets out on an adventure Destroy the Death monster built by the Empire which has the power to destroy the entire galaxy.
- ✓ Our game is **Action** category Game. A gun hits various monsters. With Increasing level randomness of monsters leads to difficulty.
- ✓ Game consists of 21 stages, transitions increasing stages leads to difficulty level by randomizing the positions of monsters.
- ✓ By hitting a monster score gets up by one point and sufficient score is required to reach next level.
- ✓ If a player gets a highest score then he/she is congratulated, screen appear with a smiley randomizing along the screen.

DATA TYPES USED

1. Structures
2. Pointers
3. File Management
4. Functions
5. Arrays
6. Graphics
7. Sounds
8. Keyboard Functions

SYNTAXS USED (New)

- Outtextxy- outtextxy function display text or string at a specified point(x,y) on the screen.
- Getpixel- getpixel function returns the color of pixel present at location(x, y).
- Getmaxy() & getmaxx()- returns the maximum Y & X coordinate for current graphics mode and driver.
- khbit-Checks the availability of keystrokes.

- Putpixel() – puts pixel at defined place in defined color.

✓ **<u>Technical sections(Algorithm)/Top-Down</u>:**

- Declaring various structures and functions and passing of structures to a functions and arrays to a functions
- KEYBOARD CONTROLS

The keyboard controls have been assigned by various methods for codes of different keys needed for controlling the game. The main working codes used are the getch(); which is a console function and outputs the ASCII code of any keyboard controlled command entered into the console. The working of the code is very simple and easy to debug. The various values of the keyboard were tested and picked up accordingly to assign it to perform control functions of the game. As visible the game can be controlled by arrow keys,"UP, DOWN, RIGHT, LEFT "

The main functions assigned are as follows:

- '⇧' arrow or 'W' are used to move up the firing ship.
  The function is to reduce the y co-ordinate by 40.
  The function is passed through an 'if' condition to limit the movement till 200 pixel.
- '⇩' arrow or 'S' are used to move down the ship.
  The function is to increase the value of y co-ordinate by 40.
  The function is passed through an 'if' condition to limit the movement till 400 pixel.
- '⇨' arrow or 'D' are used to move the ship to the right.
  The function is to increase the value of x co-ordinate by 40.
  The function is passed through an 'if' condition to limit the movement till 380 pixel.
- ' ⇦ ' arrow or 'A' are used to move the ship to the left.
  The function is to decrease the value of x co-ordinate by 40.
  The function is passed through an 'if' condition to limit the movement till 40 pixel.
- 'SPACEBAR' has been used to shoot the monsters.

The firing mechanisms are working on the putpixel function of graphics.h which runs in a ‘for’ loop similar to matrix , printing pixels first in white color in a limited line and then proceeding to another. Another similar program works in the same loop deleting the already printed white pixels by blending it to the background color.

The other keyboard input function used in the game is ‘kbhit’ which is running a ‘while’ loop, waiting for keyboard to be hit. As soon as the keyboard is hit, the mainscreen(); function jumps to the main game.

The main game runs in infinite loop broken by an ‘ENTER’ or ‘ESC’ keypress.

- File Management for High Scores

Declared pointer that access the filename, through file management and a structure

```
struct player
{
 char name[30];
 int score;
}b,f;
void write()
{
 fp=fopen("PLAYERLIST","r");
 fs=fopen("SCORE","r");
 tp=fopen("TEMP","w");
 t=fopen("TEMPS","w");
 fscanf(fp,"%s",&b.name);
 fscanf(fs,"%d",&b.score);
 if(b.score<f.score)
  {
   fprintf(tp,"%s",f.name);
   fprintf(t,"%d",f.score);
  }
 else
  {
   fprintf(tp,"%s",b.name);
   fprintf(t,"%d",b.score);
  }
 fclose(fp);
 fclose(tp);
```

```
 fclose(t);
 fclose(fs);
 remove("PLAYERLIST");
 rename("TEMP","PLAYERLIST");
 remove("SCORE");
 rename("TEMPS","SCORE");
}
void read()
{
 fp=fopen("PLAYERLIST","r");
 fs=fopen("SCORE","r");
 fscanf(fp,"%s",&b.name);
 fscanf(fs,"%d",&b.score);
 printf("NAME OF THE HIGHSCORER:%s\n",b.name);
 printf("\tHIGHSCORE:%d",b.score);
}
```

- STRUCTURE

Firstly declared a 2D structure which defines integers terms x,y,z (coordinates of gun(circle)), these terms are adjusted according to a loop.

```
struct star
{
 int x,y,z;
 int state;
}a[5][360];
```

- GRAPHICS

Declared a Function void monster (int m ,int n) through which outlook of monster is made . Monster consists of simply two arcs, circle and lines coordinates a set accordingly(+m or n+) .

```
void monsters(int m,int n)
{
 arc(30+m,40+n,0,90,5);
 arc(50+m,40+n,90,180,5);
 circle(35+m,45+n,2);
 circle(45+m,45+n,2);
 arc(40+m,55+n,0,180,5);
 line(35+m,60+n,25+m,65+n);
 line(45+m,60+n,55+m,65+n);
 circle(40+m,50+n,10);
```

```
}
```

-
-
- Stages of The Game

- Declared a Function void stage1() etc , this function defines various stages and randomise the stages by randomising the monsters. As game consists of 21 stages 21 functions are made.

```
void stage2()
{
 int i,j,m,n,k,max=360;
 m=0;
 n=0;
 state();
 for(i=0;i<5;i++)
  {
   j=0;
   if(i>2)
    {
    for(k=0;k<i;k++)
     {
       m+=40;
       j+=40;
     }
    max=360-40*i;
    }
   for(;j<max;j+=40)
   {
    a[i][j].state=1;
    a[i][j].x=m;
    a[i][j].y=n;
    m+=40;
    c[j/40]=i;
   }
   n+=40;
   m=0;
  }
}
```

- Welcome Screen

Declared a Function void mainscreen , this functions act as a welcome screen to a game through various syntax outtextxy ,setcolor,putpixel and other C graphics functions ,breaks on keyboard hit.

```
void mainscreen()
{
  while(!kbhit())
    {
      putpixel(random(getmaxx()),random(getmaxy()),random(16));
      rectangle(100,30,520,220);
      settextstyle(3,HORIZ_DIR,8);
      setcolor(random(16));
      outtextxy(120,45,"STARWARS");
      settextstyle(0,HORIZ_DIR,4);
      outtextxy(0,280,"WELCOME TO THE GAME PRESS");
      outtextxy(0,350,"PRESS A KEY TO PLAY.....");
      delay(3);
    }
}
```

- Firing in the Game

Declared a Function void firing , this function defines sound. when space key is pressed then a firing sound is made(set frequency & delay of sound accordingly)

```
void firing()
{
  int count=100;
  while(count--)
    {
      sound(count*100);
      delay(2);
      nosound();
     }
}
```

While the firing appearance is given by the loop

```
for(k=0;k<(8-c[d])*40+30;k++)
        {
          for(l=0;l<4;l++)
              {
                putpixel(x+l,y-k,0);
                putpixel(x+l,y-k-10,15);
              }
```

```
        }
```

- All these declared functions are called in MAIN functions accordingly with declaration of various variables.

```
void main()
{
 int e,i,l,d,j,k,gdriver=DETECT,gmode,errorcode,ch,counter=0;
 int x=320,y=420,z,m,n;
 initgraph(&gdriver,&gmode,"C:\\TC\\BGI");
 printf("ENTER YOUR NAME");
 scanf("%s",f.name);
 f.score=0;
 j=40;
 mainscreen();
```

- Declared a Main function which contains the main logic of the game. When space key is pressed(ASCII CODE =32) and coordinates of gun and monster meet then function CLEARDEVICE acts and it clear the monster, and at the same time all other monsters are reprinted thanks to the stages function. Likewise other monster are cleared and when monster clear the score is counted up by +100.

```
for(i=0;;)
   {
    setcolor(y);
    ch=getch();
    if(ch==80||ch==115)
    {
     if(y<400)
       y+=j;
     cleardevice();
     printf("%d",f.score);
     stages(counter);
     circle(x,y,10);
     circle(x,200,2);
    }
    if(ch==72||ch==119)
    {
     if(y>280)
       y-=j;
     cleardevice();
     printf("%d",f.score);
     stages(counter);
     circle(x,y,10);
```

```
  circle(x,200,2);
 }
 if(ch==75||ch==97)
 {
  if(x>40)
    x-=j;
  cleardevice();
  printf("%d",f.score);
  stages(counter);
  circle(x,y,10);
  circle(x,200,2);
 }
 if(ch==77||ch==100)
 {
  if(x<360)
    x+=j;
  cleardevice();
  printf("%d",f.score);
  stages(counter);
  circle(x,y,10);
  circle(x,200,2);
 }
```

Transitions

- Declared a Function for transitions between games, BREAK function makes it come out of the loop and state GAME OVER.

- ✓ **Learning Outcomes:** Undergoing this C project was nice experience. As we developed a Short game in C it involve of various concepts of C language and we learned about the applicative part of various syntaxes in C , various concepts such as Graphics, sounds, pointers, passing arrays & structure to a functions was quite helpful and backbone of the game.