COURSE: FUNDAMENTALS OF PROGRAMMING

GROUP MEMBERS:

ROHA ALI 2112124

SYEDA MAHNOOR HASAN 2112135

AVISHA KATARIA 2112107

ASMA HAFIZ MIRNAWAZ 2112106

RESPECTED TEACHERS:

SHEIKH USAMA KHALID

MUHAMMAD FAIZAN TAHIR

MONTH/YEAR: JANUARY 2022

PACMAN GAME

SUMMARY OF PROJECT:

Pac-Man is an action maze chase game; the player controls the eponymous character through an enclosed maze. The objective of the game is to eat all of the dots placed in the maze while avoiding ghosts — that pursue him.

FEATURES:

1. A Pac-man we can control using arrow keys.
2. A maze.
3. Ghosts roam the maze, trying to catch Pac-Man.
4. If a ghost touches Pac-man, a life is lost.
5. When all lives have been lost, the game ends.

ALGORITHM FOR PAC-MAN:

1. Start.
2. Initialize the structure of Pac-man and Ghost.
3. In play field, we used 2-D array.
4. Void initialize the function, we have initialize all the empty field in the 2-D array to have food particles that is represent by dot.
5. Initialize all the ghost randomly and different position which will move diagonally or horizontally.
6. Now, we used void user input and we used a key for captures the key using keyboard (kbhit).
7. We used if statement, if c1 is equal to -32 then char c2 is getch () and if c1 is not equal to -32 switch.
8. Now, in void display, display the Pac-man and satisfice the blue color.
9. Move figure according the user input either the ghost or Pac-man.
10. If ghost and Pac-man position is equal then the game end and a sound play.

11. If game not end move to end step.
12. If the food calculated is above 250, set the cursor position to specify x and y coordinates.
13. And height cursor function is to height the cursor to reduce the blinking of the screen.
14. Add color in the text where necessary.
15. At the end, the main function which call all the function time up sound is played using the play sound which we used in the first in header files.
16. While condition is called infinite loop until the condition is wrong.
17. We used the sleep function is basically the pause each and every iteration
18. Set background whole at one position and all the operation are perform on the same screen.
19. Stop.

CODE:

```
#include<windows.h>

#include <stdio.h>

#include<conio.h>


#define H 30

#define W 60

#define gho1 5


struct coord

{

  int x;

  int y;

};


struct PacMan

{

  struct coord position;
```

```
  int vx;
  int vy;
  int food_coll;
};


struct Ghost
{
  struct coord position;
  int vx;
  int vy;
};


struct Ghost allGhosts[gho1];
struct Ghost allGhosts2[gho1];

struct PacMan myPacMan = {
                        {
                          .x = 1,
                          .y = 5,
                        },
                        .vx = 0,
                        .vy = 0,
                        .food_coll = 0,
                       };
```

```
char playfield[H][W] =
{
  {"#################################################################"},
  {"#                              #"},
  {"#  ###################   ######  ###################  #"},
  {"#         ####   ##   ####         #"},
  {"#  ####  ###  ####   ##   ####  ###  ####  #"},
  {"#  ####  ###  ####  ######  ####  ###  ####  #"},
  {"#  ####                    ####  #"},
  {"#   ###################   ##   ###################   #"},
  {"#              ##              #"},
  {"#  ################  ###  ##  ###  ################  #"},
  {"#  ###       ###     ###       ###  #"},
  {"#  ### ###  ########## ## ## ########## ###  ###  #"},
  {"#  ###       ###     ###       ###  #"},
  {"#  ################  ### ## ## ###  ################  #"},
  {"#                              #"},
  {"#  ####  ###  #######  ####  #######  ###  ####  #"},
  {"#  ####  ###  #######  ####  #######  ###  ####  #"},
  {"#  ####  ###   ###  ####  ###   ###  ####  #"},
  {"#  ####         ####         ####  #"},
  {"#  ####   #############     #############  ####  #"},
  {"#  ####   ###             ###   ####  #"},
  {"#  ####   ###             ###   ####  #"},
  {"#  ####   ###################################  ####  #"},
  {"#  ####                     ####  #"},
  {"#  ####  #############   ##   #############  ####  #"},
```

```
  { "#  ####  ##############   ##   ############## ####  #" },
  { "#  ####    ####     ##     ####    ####  #" },
  { "#        ####   ##########   ####       #" },
  { "#    ####       ##########       ####    #" },
  { "#################################################################" }
};
void display();
void SetColor(int ForgC);
//
void initialize()
{
  int i;

  for(i = 0; i < H; i++)
  {
     int j;
    for( j = 0; j < W; j++)
    {
      if (playfield[i][j]==' ')
         playfield[i][j] = '.';
    }
  }

  for ( i = 0; i <gho1; i++)
  {
    allGhosts[i].vx = 0;
    allGhosts[i].vy = 0;
```

```
  int x,y;
  do
  {
     x = 1 + rand() % (W-1);
     y = 1 + rand() % (H-1);

  } while (playfield[y][x] != '.');
   allGhosts[i].position.x = x;
   allGhosts[i].position.y = y;
   playfield[y][x] = '%';

}
  for ( i = 0; i < gho1; i++)
{
   allGhosts2[i].vx = 0;
   allGhosts2[i].vy = 0;

   int x,y;
   do
  {
     x = 1 + rand() % (W-1);
     y = 1 + rand() % (H-1);

  } while (playfield[y][x] != '.');
   allGhosts2[i].position.x = x;
   allGhosts2[i].position.y = y;
```

```
      playfield[y][x] = '%';

    }




}

void user_input()
{
  if (_kbhit())
  {
    char c1 = _getch();

    if (c1 == -32)
    {
      char c2 = _getch();

      myPacMan.vx = 0;
      myPacMan.vy = 0;
      int i;
      for(i=0;i<gho1;i++)
     {
         allGhosts[i].vx=0;
         allGhosts[i].vy=0;
         allGhosts2[i].vx=0;
         allGhosts2[i].vy=0;
```

```
      }

       switch (c2)
       {
          case 72: myPacMan.vy = -1; break;
          case 80: myPacMan.vy = +1; break;
          case 75: myPacMan.vx = -1; break;
          case 77: myPacMan.vx = +1; break;
       }
       for(i=0;i<gho1;i++)
       {
           allGhosts[i].vx=+1;
           allGhosts[i].vy=+1;
           allGhosts2[i].vx=0;
           allGhosts2[i].vy=+1;
       }
     }


  }

}
void display()
{
   SetColor(1);
printf("                          ___  __   __  __    _  __  _     \n");
printf("                         ||   || ||   ||  ||    ||  |   | || ||   || ||  |   ||\n");
```

```
printf("                          ||    || ||   ||  ||     ||   |  |   || ||   || ||   |   ||\n");
printf("                          ||__|| ||_||  ||     ||   ||   || ||__|| ||    |  ||\n");
printf("                          ||      ||   ||  ||     ||       || ||   || ||    | ||\n");
printf("                          ||      ||   ||  ||__   ||       || ||   || ||     |||\n");
SetColor(15);
}

void move_figures()
{

  playfield[myPacMan.position.y][myPacMan.position.x] = ' ';
  int i;
  for(i=0;i<gho1;i++)
  {
     playfield[allGhosts[i].position.y][allGhosts[i].position.x] = ' ';
      playfield[allGhosts2[i].position.y][allGhosts2[i].position.x] = ' ';
  }

  int nx = myPacMan.vx + myPacMan.position.x;
  int ny = myPacMan.vy + myPacMan.position.y;
  int mx[5];
  int my[5];
  int mx1[5];
  int my1[5];
 for(i=0;i<gho1;i++)
  {
      mx[i] = allGhosts[i].vx + allGhosts[i].position.x;
```

```
    my[i] = allGhosts[i].vy + allGhosts[i].position.y;
    mx1[i] = allGhosts2[i].vx + allGhosts2[i].position.x;
    my1[i] = allGhosts2[i].vy + allGhosts2[i].position.y;
}

if (playfield[ny][nx] == '#')
{
  myPacMan.vx = 0;
  myPacMan.vy = 0;
}
for(i=0;i<gho1;i++)
{

if(playfield[my[i]][mx[i]]=='#')
  {
     if(allGhosts[i].vx>0 || allGhosts[i].vy>0)
     {

         allGhosts[i].vx=-1;
         allGhosts[i].vy=-1;
      }
      else
      {
         allGhosts[i].vx=+1;
         allGhosts[i].vy=+1;
      }
  }
```

```
    if(playfield[my1[i]][mx1[i]]=='#')
    {
       if(allGhosts2[i].vy<0)
        {

            allGhosts2[i].vx=0;
            allGhosts2[i].vy=+1;
         }
         else
         {
            allGhosts2[i].vx=0;
            allGhosts2[i].vy=-1;
         }
    }



}

   myPacMan.position.x += myPacMan.vx;
   myPacMan.position.y += myPacMan.vy;
    for(i=0;i<gho1;i++)
     {
        allGhosts[i].position.x+=allGhosts[i].vx;
        allGhosts[i].position.y+=allGhosts[i].vy;
        allGhosts2[i].position.x+=allGhosts2[i].vx;
        allGhosts2[i].position.y+=allGhosts2[i].vy;
```

```
  }

if (playfield[ny][nx] == '.')
{
  myPacMan.food_coll++;
}
for(i=0;i<gho1;i++)

{

 if(playfield[my[i]][mx[i]] == '.')
 {

       playfield[my[i]-allGhosts[i].vy][mx[i]-allGhosts[i].vx]='.';
 }
 if(playfield[my1[i]][mx1[i]] == '.')
 {

       playfield[my1[i]-allGhosts2[i].vy][mx1[i]-allGhosts2[i].vx]='.';
 }
 }

playfield[myPacMan.position.y][myPacMan.position.x] = '@';
 for(i=0;i<gho1;i++)
 {
    playfield[allGhosts[i].position.y][allGhosts[i].position.x]='%';
```

```
        playfield[allGhosts2[i].position.y][allGhosts2[i].position.x]='%';
    }


}

void show_playfield()
{
    int i;
   for ( i = 0; i < H; i++)
   {
       printf("                                    ");
       int j;
      for ( j = 0; j < W; j++)
      {
         if(playfield[i][j]=='.')
         {
            SetColor(4);
          printf("%c", playfield[i][j]);
             SetColor(15);
         }
           else if(playfield[i][j]=='%')
           {
              SetColor(9);
              printf("%c", playfield[i][j]);
              SetColor(15);
           }
           else if(playfield[i][j]=='@')
```

```
            {
               SetColor(14);
               printf("%c", playfield[i][j]);
               SetColor(15);
            }
            else
            {
               printf("%c", playfield[i][j]);
            }
      }
      printf("\n");
   }

   printf("                              Score: %d\n", myPacMan.food_coll);
}
void check_coll()
{
    int i=0;
     for(i=0;i<gho1;i++)
   {
       if((allGhosts[i].position.x==myPacMan.position.x &&
allGhosts[i].position.y==myPacMan.position.y) ||
(allGhosts2[i].position.x==myPacMan.position.x &&
allGhosts2[i].position.y==myPacMan.position.y))
        {
            system("cls");
            display();
            printf("\n\n\n");
```

```
            printf("     \t\t\t\t\t\t\t\tYOUR SCORE IS:%d",myPacMan.food_coll);

            getchar();
            exit(0);
        }
        if(myPacMan.food_coll>=250)
        {
            system("cls");
            display();
            printf("\n\n\n");
            printf("     \t\t\t\t\t\t\t\tYOUR SCORE IS:%d",myPacMan.food_coll);

            getchar();
            exit(0);
            }
    }
}

void set_cursor_position(int x, int y)
{

   COORD coord = { x, y };
   SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coord);

}

void hidecursor()
```

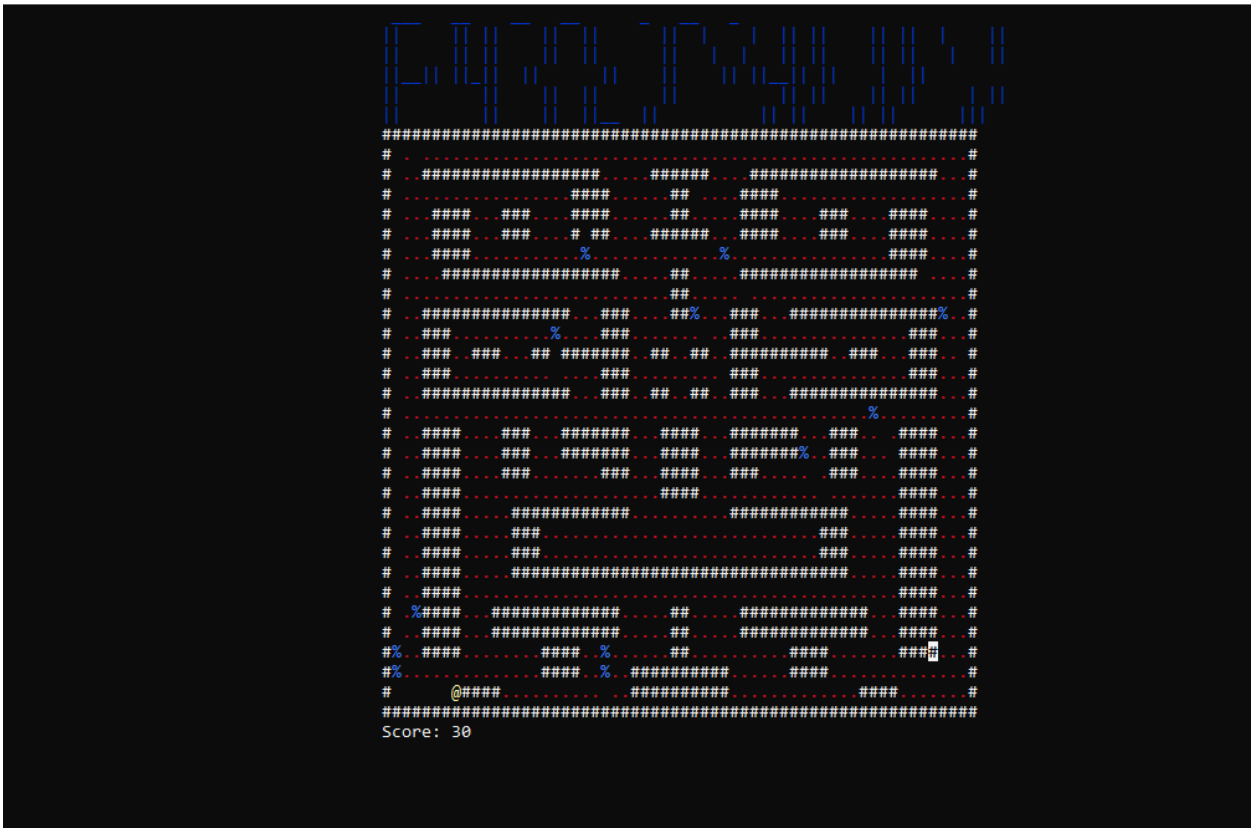
```
{
  HANDLE consoleHandle = GetStdHandle(STD_OUTPUT_HANDLE);
  CONSOLE_CURSOR_INFO info;
  info.dwSize = 100;
  info.bVisible = FALSE;
  SetConsoleCursorInfo(consoleHandle, &info);
}

void SetColor(int ForgC)
{
    WORD wColor;

    HANDLE hStdOut = GetStdHandle(STD_OUTPUT_HANDLE);
    CONSOLE_SCREEN_BUFFER_INFO csbi;

    if(GetConsoleScreenBufferInfo(hStdOut, &csbi))
    {
        wColor = (csbi.wAttributes & 0xF0) + (ForgC & 0x0F);
        SetConsoleTextAttribute(hStdOut, wColor);
    }
    return;
}


int main()
{
  system("cls");
```

```
  hidecursor();
  initialize();



  while (1)
  {
    user_input();
    move_figures();
    display();
    show_playfield();
    check_coll();
    Sleep( 10/ 30 );
    set_cursor_position(0,0);
  }

}
```

OUTPUT:

"C:\Users\Hp\Documents\avisha kataria\avisha kataria\bin\Debug\avisha kataria.exe"

"C:\Users\Hp\Documents\avisha kataria\avisha kataria\bin\Debug\avisha kataria.exe"



"C:\Users\Hp\Documents\avisha kataria\avisha kataria\bin\Debug\avisha kataria.exe"