JSPM's
JAYAWANTRAO SAWANT POLYTECHNIC,
Handewadi Road, Hadapsar, Pune-28
Department of Computer Engineering
Academic Year 2020-21

MICRO PROJECT

TITLE OF THE PROJECT

Bricks game in c

Program : CO **Program code:** CO5I

Course : Operating System **Course code: :** 22516

Class : TYCO1 **Group No:** 9

Project Guide: Ms. Z.S.Sajjade

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION

Certificate

This is to certify that Mr Ashish Dhananjay Kadam of V Semester of Diploma in Computer Engineering of Institute Jayawantrao Sawant Polytechnic (Code: 0711) has completed the Micro Project satisfactorily in Subject – Operating System (22516) for the academic year 2020- 2021- as prescribed in the curriculum.

Place: Hadapsar, Pune.

Enrollment No: 1807110129

Date: ……

Exam Seat No:-

Subject Teacher Head of the Department Principal

❖ MICROPROJECT GROUP DETAILS

Sr No.	Roll No.	Name	Enrollment No.	Seat No.
1	9	Ashish Dhananjay Kadam	1807110129	

❖ INDEX

❖ **Title of Micro project:** Bricks game in C.

1.0 Brief Introduction: In this project, we have Developed the Bricks game in C by creating objects and design by initializing the graphics system and then calculating the x and y co ordinates along with the center of the screen and calculating the various co-ordinates of the old and new positions of the ball and the bricks and we also created 2 different sounds for the game along with the variations of levels which make the game totally interesting and enjoyable to play.

2.0 Aim of Micro Project: The aim of the project is to create a Bricks game in C.

3.0 Action Plan (Sequence and time required for major activities for 8 week)

Sr. No	Details of activity	Planned start date	Planned Finish date	Name of Responsible Team members
1	Collecting the information of view	4/10/20	4/10/20	Ashish Dhananjay Kadam
2	Sorting the information of view	11/10/20	11/10/20	
3	Compilation of the project	18/10/20	18/10/20	
4	Submission of the project	25/10/20	25/10/20	

4.0 Resources required (major resources such as raw material, some machining facility, software etc.)

Sr. NO	Name of resource / material	Specification	Quantity	Remarks
1	Computer System	Windows 7 ultimate	1	
2	pc	Windows 10	1	
3	Web	Chrome browser to acquire basic knowledge about Operating System .	1	

❖ INTRODUCTION

- **In this microproject we have created the Bricks game in c .**

- **Following are the Simple Algorithm steps for creating Bricks game in C:-**

1. Initialise the graphics system
2. Get the maximum x and y screen coordinates
3. Calculate center of screen
4. Display opening screen and receive player's level
5. Set speed of ball as per the level chosen
6. Draw the bricks, the paddle and the ball
7. Allocate memory for storing the image of the paddle
8. Allocate memory for storing the image of the ball
9. If memory allocation unsuccessful
10. Store the image of the paddle and the ball into allocated memory
11. Store current position of the paddle and ball
12. Display balls in hand (initially 3)
13. Display initial score
14. Select font and alignment for displaying text
15. Save the current x and y coordinates of the ball
16. Update ballx and bally to move the ball in appropriate direction
17. As per the position of ball determine the layer of bricks to check
18. If the ball hits the left boundary, deflect it to the right
19. If the ball hits the right boundary, deflect it to the left
20. If the ball hits the top boundary, deflect it down
21. If the ball is in the area occupied by the bricks
22. If there is no brick present exactly at the top of the ball
23. Determine if the boundary of the ball touches a brick
24. Check whether there is a brick to the right of the ball
25. If there is a brick
26. Check whether there is a brick to the left of the ball
27. If the ball does not touch a brick at the top, left or right
28. Check if the ball has moved above the current layer
29. If so, change current layer appropriately

30. Put the image of the ball at the old coordinates
31. Erase the image at the old coordinates
32. Place the image of the ball at the new coordinates
33. Introduce delay
34. Carry on with moving the ball
35. Control comes to this point only if the ball is touching a brick
36. Play music
37. Erase the brick hit by the ball
38. If the brick hit happens to be on the extreme right
39. Redraw right boundary
40. If the brick hit happens to be on the extreme left
41. If the brick hit happens to be in the topmost layer
42. Redraw top boundary
43. Set appropriate array element to 1 to indicate absence of brick */
44. Update the y coordinate
45. Change the direction of the ball
46. Increment score
47. Print latest score
48. If the first brick is hit during a throw
49. For the consecutive bricks hit during the same throw
50. Clear part of the screen used for displaying Well done message
51. If the ball has reached the bottom
52. If the paddle has missed the ball
53. Continue the descent of the ball
54. Erase the image of the ball at the old coordinates
55. Put the image of the ball at the updated coordinates
56. Save the current x and y coordinates of the ball
57. Update ballx and bally to move the ball in appropriate direction
58. Decrement the number of chances
59. Decrement 20 points for each ball lost
60. Print latest score
61. Erase one out of the available balls
62. If the last ball is being played
63. Your last ball... Be careful
64. If all the balls are lost
65. I warned you! Try again
66. If ball is collected on paddle
67. Restore the y coordinate of ball
68. Deflect the ball upwards
69. Put the image of the ball at the old coordinates

70. Erase the image of the ball at the old coordinates
71. Put the image of the ball at the updated coordinates
72. If all the bricks have been destroyed
73. You win
74. You are simply GREAT!");
75. Introduce delay
76. If the user has pressed a key to move the paddle
77. Issue interrupt to obtain the ascii and scan codes of key hit
78. Put the image of the paddle at the old coordinates
79. Erase the image of the paddle at the old coordinates
80. If Esc key has been pressed
81. Right arrow key
82. Left arrow key
83. If paddle goes beyond left boundary
84. If paddle goes beyond right boundary
85. Put the image of the paddle at the proper position
86. Creates opening screen
87. Array showing the positions where a brick is needed to form the figure BRICK
88. Form the word BRICKS
89. Draw pattern at the bottom of the screen
90. Draw the paddle and the ball
91. Display menu
92. Clear the region below the word BRICKS
93. If a special key is hit, flush the keyboard buffer
94. Draws bricks at the start of the game
95. Draw a brick at appropriate coordinates
96. Draws a brick at the proper position
97. Erases the specified brickb - brick number, l - layer
98. Plays different types of music
99. Natural frequencies of 7 note
100. Flush the keyboard buffer

❖ **Program**

```
#include <stdio.h>
#include <conio.h>
#include <process.h>
#include <dos.h>
#include <stdlib.h>
#include <graphics.h>
#include <ctype.h>

#define NULL 0
#define YES 1
#define NO 0

int maxx, maxy, midx, midy;
int bri[5][20];

void music(int);
char mainscreen();
void drawbrick(int, int);
void erasebrick(int, int);
void bricks(void);


int main() {
union REGS ii, oo;
int ballx, bally, paddlex, paddley, dx = 1, dy = -1, oldx, oldy;
int gm = CGAHI, gd = CGA, playerlevel;
int i, flag = 0, speed = 25, welldone = NO, score = 0, chance = 4, area;
int layer[5] = {
10,
20,
30,
40,
50
}, limit = 50, currentlayer = 4;

char * p1, * p2;


/* initialise the graphics system */
```

```
initgraph( &gd, &gm, "C:\\TURBOC3\\BGI");


/* get the maximum x and y screen coordinates */
maxx = getmaxx();
maxy = getmaxy();

/* calculate center of screen */
midx = maxx / 2;
midy = maxy / 2;

/* display opening screen and receive player's level */
playerlevel = mainscreen();

/* set speed of ball as per the level chosen */
switch (playerlevel) {
case 'A':
case 'a':
speed = 15;
break;

case 'E':
case 'e':
speed = 10;
}

/* draw the bricks, the paddle and the ball */
rectangle(0, 0, maxx, maxy- 12);
bricks();
rectangle(midx- 25, maxy- 7- 12, midx + 25, maxy- 12);
floodfill(midx, maxy- 1- 12, 1);
circle(midx, maxy- 13- 12, 12);
floodfill(midx, maxy- 10- 12, 1);

/* allocate memory for storing the image of the paddle */
area = imagesize(midx- 12, maxy- 18, midx + 12, maxy- 8);


p1 =(char *) malloc(area);
```

```
/* allocate memory for storing the image of the ball */


area = imagesize(midx- 25, maxy- 7, midx + 25, maxy- 1);
p2 =(char *) malloc(area);

/* if memory allocation unsuccessful */
if (p1 == NULL || p2 == NULL) {

puts("Insufficient memory!!");
exit(1);
}

/* store the image of the paddle and the ball into allocated memory */
getimage(midx- 12, maxy- 7- 12- 12 + 1, midx + 12, maxy- 8- 12, p1);
getimage(midx- 25, maxy- 7- 12, midx + 25, maxy- 1- 12, p2);

/* store current position of the paddle and ball */
paddlex = midx- 25;
paddley = maxy- 7- 12;
ballx = midx- 12;
bally = maxy- 7- 12 + 1- 12;

/* display balls in hand ( initially 3 ) */
gotoxy(45, 25);
printf("Balls Remaining: ");
for (i = 0; i < 3; i++) {
circle(515 + i * 35, maxy - 5, 12);
floodfill(515 + i * 35, maxy - 5, 1);
} /* display initial score */
gotoxy(1, 25);
printf("Your Score: %4d", score); /* select font and alignment for displaying text */
settextjustify(CENTER_TEXT, CENTER_TEXT);
settextstyle(SANS_SERIF_FONT, HORIZ_DIR, 4);
while (1) {
flag = 0; /* save the current x and y coordinates of the ball */
oldx = ballx;
oldy = bally; /* update ballx and bally to move the ball in appropriate direction */
ballx = ballx + dx;
```

```
bally = bally + dy; /* as per the position of ball determine the layer of bricks to check */

if (bally > 40) {
limit = 50;


currentlayer = 4;
} else {
if (bally > 30) {
limit = 40;
currentlayer = 3;
} else {
if (bally > 20) {

limit = 30;
currentlayer = 2;
} else {
if (bally > 10) {
limit = 20;
currentlayer = 1;
} else {
limit = 10;
currentlayer = 0;
}
}
}
}

/* if the ball hits the left boundary, deflect it to the right */
if (ballx < 1) {
music(5);
ballx = 1;
dx = -dx;
} /* if the ball hits the right boundary, deflect it to the left */
if (ballx > (maxx- 24- 1)) {
music(5);
ballx = maxx- 24- 1;
dx = -dx;
}
```

```
/* if the ball hits the top boundary, deflect it down */
if (bally < 1) {
music(5);
bally = 1;
dy = -dy;



} /* if the ball is in the area occupied by the bricks */
if (bally < limit) {
/* if there is no brick present exactly at the top of the ball */
if (bri[currentlayer][(ballx + 10) / 32] == 1) {
/* determine if the boundary of the ball touches a brick */
for (i = 1; i <= 6; i++) {
/* check whether there is a brick to the right of the ball */
if (bri[currentlayer][(ballx + i + 10) / 32] == 0) {
/* if there is a brick */
ballx = ballx + i;

flag = 1;
break;
} /* check whether there is a brick to the left of the ball */
if (bri[currentlayer][(ballx - i + 10) / 32] == 0) {
ballx = ballx - i;
flag = 1;
break;
}
} /* if the ball does not touch a brick at the top, left or right */
if (!flag) {
/* check if the ball has moved above the current layer */
if (bally < layer[currentlayer - 1]) {
/* if so, change current layer appropriately */
currentlayer--;
limit = layer[currentlayer];
} /* put the image of the ball at the old coordinates */
putimage(oldx, oldy, p1, OR_PUT); /* erase the image at the old coordinates */
putimage(oldx, oldy, p1, XOR_PUT); /* place the image of the ball at the new coordinates
*/
putimage(ballx, bally, p1, XOR_PUT); /* introduce delay */
delay(speed); /* carry on with moving the ball */
```

```
continue;
}
} /* control comes to this point only if the ball is touching a brick */
music(4); /* play music */ /* erase the brick hit by the ball */
erasebrick((ballx + 10) / 32, currentlayer); /* if the brick hit happens to be on the extreme
right */
if ((ballx + 10) / 32 == 19) line(maxx, 0, maxx, 50); /* redraw right boundary */ /* if the
brick hit happens to be on the extreme left */


if ((ballx + 10) / 32 == 0) line(0, 0, 0, 50); /* redraw left boundary */ /* if the brick hit
happens to be in the topmost layer */


if (currentlayer == 0) line(0, 0, maxx, 0); /* redraw top boundary */ /* set appropriate
array element to 1 to indicate absence of brick */
bri[currentlayer][(ballx + 10) / 32] = 1;
bally = bally + 1; /* update the y coordinate */
dy = -dy; /* change the direction of the ball */
score += 5; /* increment score */
gotoxy(16, 25);
printf("%4d", score); /* print latest score */ /* if the first brick is hit during a throw */
if (welldone == NO) welldone = YES;
else {

/* for the consecutive bricks hit during the same throw */
outtextxy(midx, midy, "Well done!");
music(1);
}
} /* clear part of the screen used for displaying Well done message */
if (bally > 50 && welldone == YES) {
setviewport(midx- 32 * 2.5, midy- 32 / 2, midx + 32 * 2.5, midy + 32 / 2, 1);
clearviewport();
setviewport(0, 0, maxx, maxy, 1);
welldone = NO;
}

/* if the ball has reached the bottom */
if (bally > 180- 12) {
welldone = NO;
```

```
/* if the paddle has missed the ball */
if (ballx < paddlex - 20 || ballx > paddlex + 50) {
/* continue the descent of the ball */
while (bally < 177) {
/* erase the image of the ball at the old coordinates */
putimage(oldx, oldy, p1, XOR_PUT); /* put the image of the ball at the updated
coordinates */
putimage(ballx, bally, p1, XOR_PUT); /* introduce delay */
delay(speed); /* save the current x and y coordinates of the ball */


oldx = ballx;
oldy = bally; /* update ballx and bally to move the ball in appropriate direction */
ballx = ballx + dx;
bally = bally + dy;
}

chance--; /* decrement the number of chances */
score -= 20; /* decrement 20 points for each ball lost */
gotoxy(16, 25);
printf("%4d", score); /* print latest score */
music(2); /* erase one out of the available balls */
if (chance) putimage(515 + (chance - 1) * 35 - 12, maxy - 10, p1, XOR_PUT); /* if the last
ball is being played */
if (chance == 1) {
gotoxy(45, 25);
printf("Your last ball... Be careful!");

} /* if all the balls are lost */
if (!chance) {
gotoxy(45, 25);
printf("Press any key... ");
outtextxy(midx, midy, "I warned you! Try again");
music(3);
closegraph();
restorecrtmode();
exit(0);
}
} /* if ball is collected on paddle */
music(5);
```

```
bally = 180 - 12; /* restore the y coordinate of ball */
dy = -dy; /* deflect the ball upwards */
} /* put the image of the ball at the old coordinates */
putimage(oldx, oldy, p1, OR_PUT); /* erase the image of the ball at the old coordinates */
putimage(oldx, oldy, p1, XOR_PUT); /* put the image of the ball at the upadted
coordinates */
putimage(ballx, bally, p1, XOR_PUT); /* if all the bricks have been destroyed */
if (score == 500 - ((4 - chance) * 20)) {
outtextxy(midx, midy, "You win !!!");
if (score < 500) outtextxy(midx, midy + 30, "Try scoring 500");


else outtextxy(midx, midy + 30, "You are simply GREAT!");
music(3);
closegraph();
restorecrtmode();
exit(0);
} /* introduce delay */
delay(speed); /* if the user has pressed a key to move the paddle */
if (kbhit()) {
/* issue interrupt to obtain the ascii and scan codes of key hit */
ii.h.ah = 0;
int86(22, & ii, & oo); /* put the image of the paddle at the old coordinates */
putimage(paddlex, paddley, p2, OR_PUT); /* erase the image of the paddle at the old
coordinates */
putimage(paddlex, paddley, p2, XOR_PUT); /* if Esc key has been pressed */
if (oo.h.ah == 1) exit(0); /* right arrow key */
if (oo.h.ah == 75) paddlex = paddlex - 20; /* left arrow key */
if (oo.h.ah == 77) paddlex = paddlex + 20; /* if paddle goes beyond left boundary */
if (paddlex < 0) paddlex = 0; /* if paddle goes beyond right boundary */

if (paddlex > 589)
paddlex = 589;

/* put the image of the paddle at the proper position */
putimage(paddlex, paddley, p2, XOR_PUT);
}
}
}
```

```
/* creates opening screen */
char mainscreen() {
/* array showing the positions where a brick is needed to form the figure BRICKS */
int ff[12][40] = {
1,1,1,1,0,0,0,1,1,1,1,0,0,0,1,1,1,1,1,0,0,0,1,1,1,0,0,1,0,0,0,0,1,0,0,0,1,1,1,0,
1,0,0,0,1,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,0,1,0,0,0,1,0,1,0,0,0,1,0,0,0,1,0,0,0,1,
1,0,0,0,0,1,0,1,0,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,0,0,0,1,0,0,1,0,0,0,1,0,0,0,0,0,
1,0,0,0,0,1,0,1,0,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,0,0,0,1,0,1,0,0,0,0,1,0,0,0,0,0,
1,0,0,0,1,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,1,1,0,0,0,0,0,1,0,0,0,0,0,
1,1,1,1,0,0,0,1,1,1,1,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1,1,1,0,0,
1,0,0,0,1,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,1,0,
1,0,0,0,0,1,0,1,0,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,1,

1,0,0,0,0,1,0,1,0,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,0,0,1,
1,0,0,0,1,0,0,1,0,0,0,0,1,0,0,0,1,0,0,0,0,1,0,0,0,1,0,1,0,0,0,1,0,0,1,0,0,0,0,1,
1,1,1,1,0,0,0,1,0,0,0,0,1,0,1,1,1,1,1,0,0,0,1,1,1,0,0,1,0,0,0,0,1,0,0,1,1,1,1,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
} ;
int i, j, lx = 0, ly = 0, ch;

/* draw boundary */
rectangle(0, 0, maxx, maxy);

/* form the word BRICKS */
for (i = 0; i < 12; i++) {
for (j = 0; j < 40; j++) {
if (ff[i][j]) rectangle(lx, ly, lx + 15, ly + 9);
lx = lx + 16;
}
lx = 0;
ly = ly + 10;
} /* draw pattern at the bottom of the screen */
line(0, maxy - 12, maxx, maxy - 12);

setfillstyle(XHATCH_FILL, WHITE);
floodfill(2, maxy - 2, WHITE); /* draw the paddle and the ball */
setfillstyle(SOLID_FILL, WHITE);
rectangle(midx - 25, maxy - 7 - 12, midx + 25, maxy - 12);
floodfill(midx, maxy - 1 - 12, 1);
```

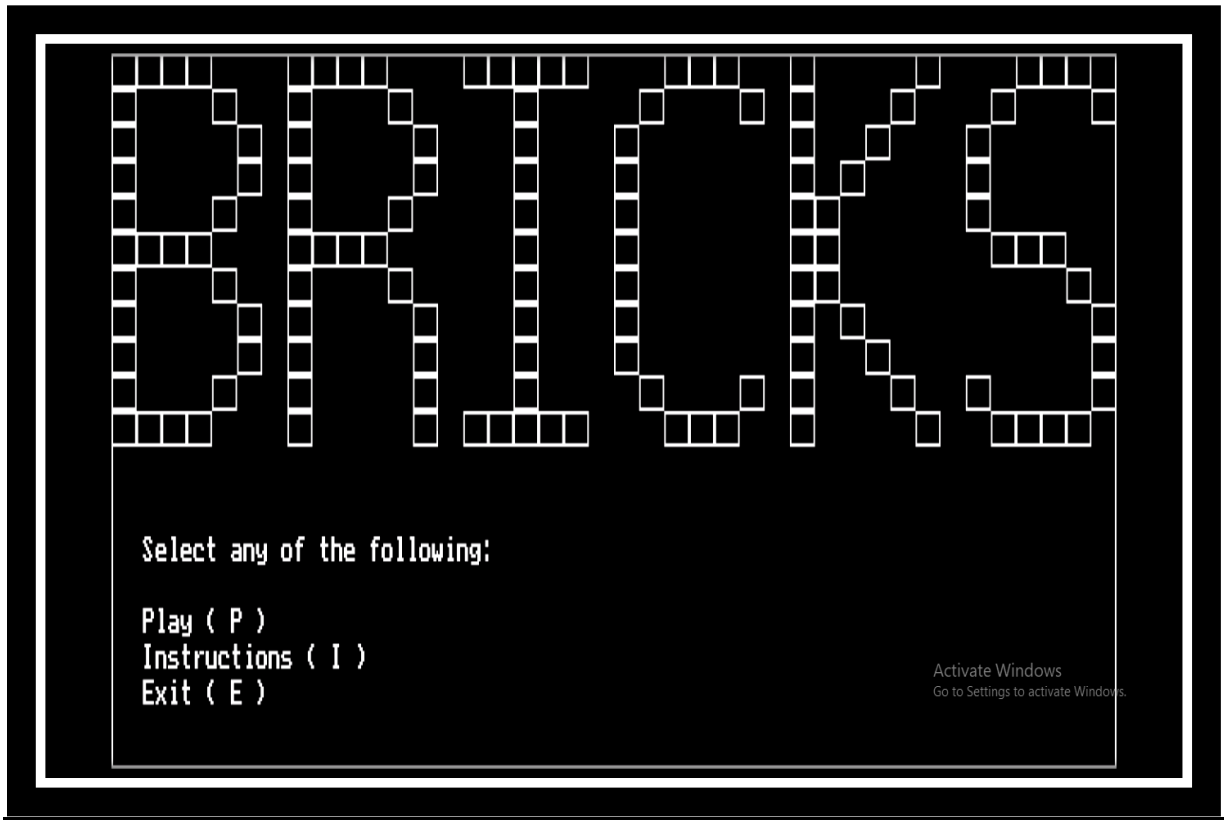
```
circle(midx, maxy - 13 - 12, 12);
floodfill(midx, maxy - 10 - 12, 1);
music(3); /* play music */ /* display menu */
while (1) {
/* clear the region below the word BRICKS */
setviewport(1, 125 - 12, maxx - 1, maxy - 1, 1);
clearviewport();
setviewport(0, 0, maxx, maxy, 1);
outtextxy(20, 135, "Select any of the following:");
outtextxy(20, 155, "Play ( P )");
outtextxy(20, 165, "Instructions ( I )");
outtextxy(20, 175, "Exit ( E )");
ch = 0; /* continue till the correct choice is made */
while (!(ch == 'E' || ch == 'I' || ch == 'P')) {


fflush(stdin); /* if a special key is hit, flush the keyboard buffer */
if ((ch = getch()) == 0) getch();
else ch = toupper(ch);
}
if (ch == 'P') break;
switch (ch) {
case 'I':
setviewport(1, 125 - 12, maxx - 1, maxy - 1, 1);
clearviewport();
setviewport(0, 0, maxx, maxy, 1);
settextstyle(DEFAULT_FONT, HORIZ_DIR, 1);
outtextxy(20, 125, " Instructions ");
settextstyle(DEFAULT_FONT, HORIZ_DIR, 0);
outtextxy(20, 140, "Use left and right arrow keys to move paddle.");
outtextxy(20, 150, "If you don't collect the ball on the paddle, you lose the ball.");
outtextxy(20, 160, "On loosing a ball you loose 20 points.");
outtextxy(20, 170, "On taking a brick you gain 5 points.");
outtextxy(20, 185, "Press any key...");
fflush(stdin);
if (getch() == 0) getch();
break;
case 'E':

closegraph();
```

```
restorecrtmode();
exit(0);
}
}
setviewport(1, 125 - 12, maxx - 1, maxy - 1, 1);
clearviewport(); /* prompt the user for the level desired */
setviewport(0, 0, maxx, maxy, 1);
outtextxy(20, 135, "Select any of the following levels:");
outtextxy(20, 155, "Novice ( N )");
outtextxy(20, 165, "Advanced ( A )");
outtextxy(20, 175, "Expert ( E )"); /* get user's choice */
fflush(stdin);
if ((ch = getch()) == 0) getch();
clearviewport(); /* return the choice made by the user */
return (ch);
}


/* draws bricks at the start of the game */
void bricks() {
int i, j, lx = 0, ly = 0;
for (i = 0; i < 5; i++) /* 5 rows */ {
for (j = 0; j < 20; j++) /* 20 columns */ {
/* draw a brick at appropriate coordinates */
drawbrick(lx, ly);
lx = lx + 32;
}
lx = 0;
ly = ly + 10;
}
}

/* draws a brick at the proper position */
void drawbrick(int lx, int ly) {
rectangle(lx, ly, lx + 31, ly + 9);
rectangle(lx + 2, ly - 2, lx + 31 - 2, ly + 9 - 2);
floodfill(lx + 1, ly + 1, 2);
}

/* erases the specified brick */
```

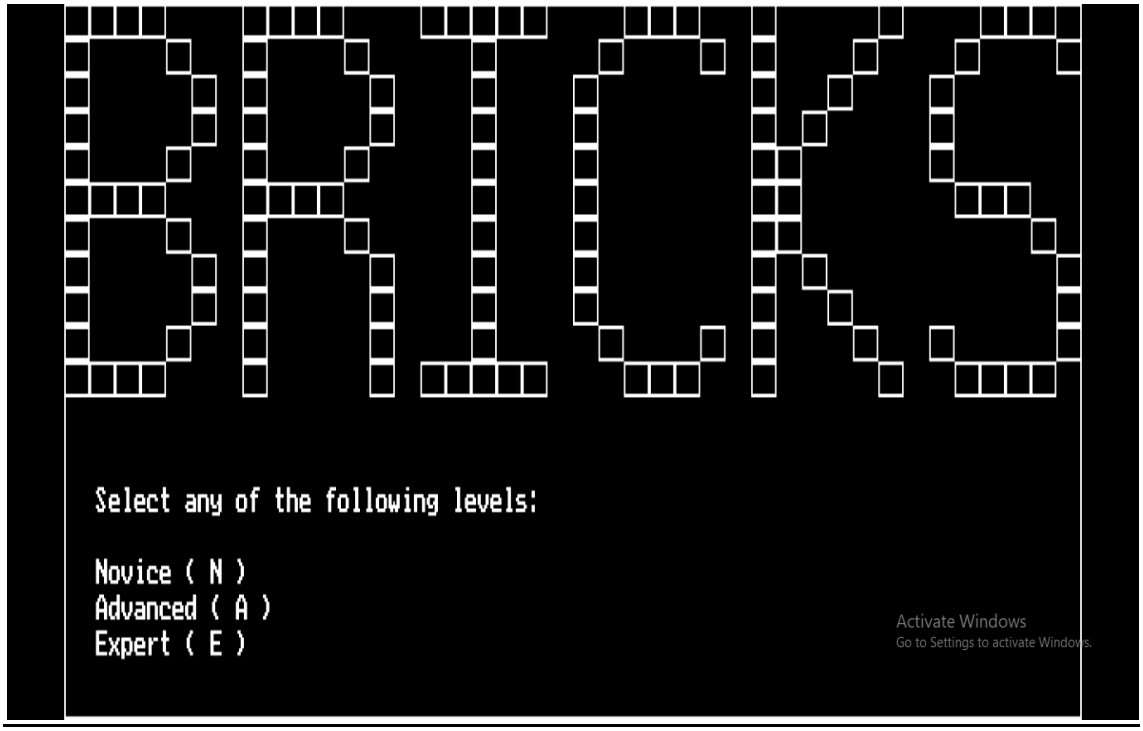
```
void erasebrick(int b, int l) {

/* b - brick number, l - layer */
setcolor(BLACK);
rectangle(b * 32, l * 10, (b * 32) + 31, (l * 10) + 9);
rectangle(b * 32 + 1, l * 10, (b * 32) + 31 - 1, (l * 10) + 9 - 1);
rectangle(b * 32 + 2, l * 10, (b * 32) + 31 - 2, (l * 10) + 9 - 2);
setcolor(WHITE);
}

/* plays different types of music */
void music(int type) {
/* natural frequencies of 7 notes */
float octave[7] = {
130.81,
146.83,
164.81,
174.61,


196,
220,
246.94
};
int n, i;
switch (type) {
case 1:
for (i = 0; i < 7; i++) {
sound(octave[i] * 8);
delay(30);
}
nosound();
break;
case 2:
for (i = 0; i < 15; i++) {
n = random(7);
sound(octave[n] * 4);
delay(100);
}
nosound();
```

```
break;
case 3:
while (!kbhit()) {
n = random(7);
sound(octave[n] * 4);

delay(100);
}
nosound(); /* flush the keyboard buffer */
if (getch() == 0) getch();
break;
case 4:
for (i = 4; i >= 0; i--) {
sound(octave[i] * 4);
delay(15);
}
nosound();
break;

case 5:


sound(octave[6] * 2);
delay(50);
nosound();
}
}
```

❖ **Output of the program :-**

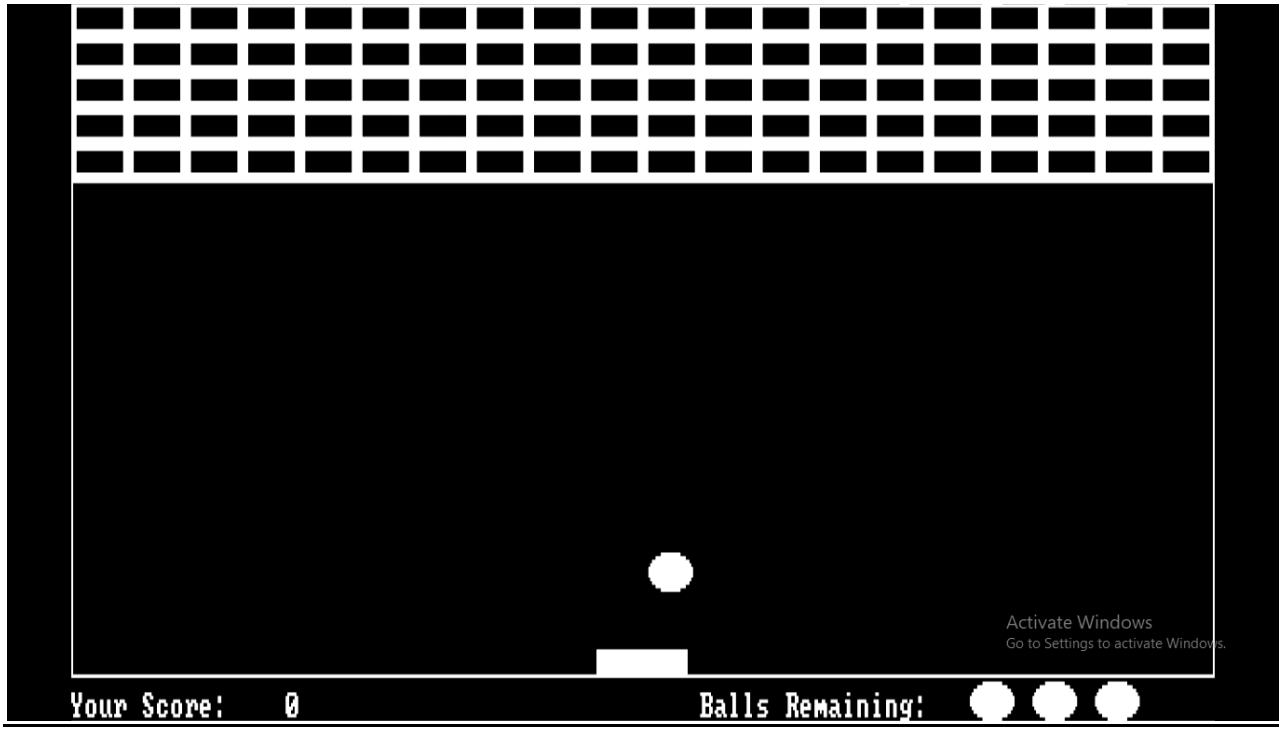


When we press P:-

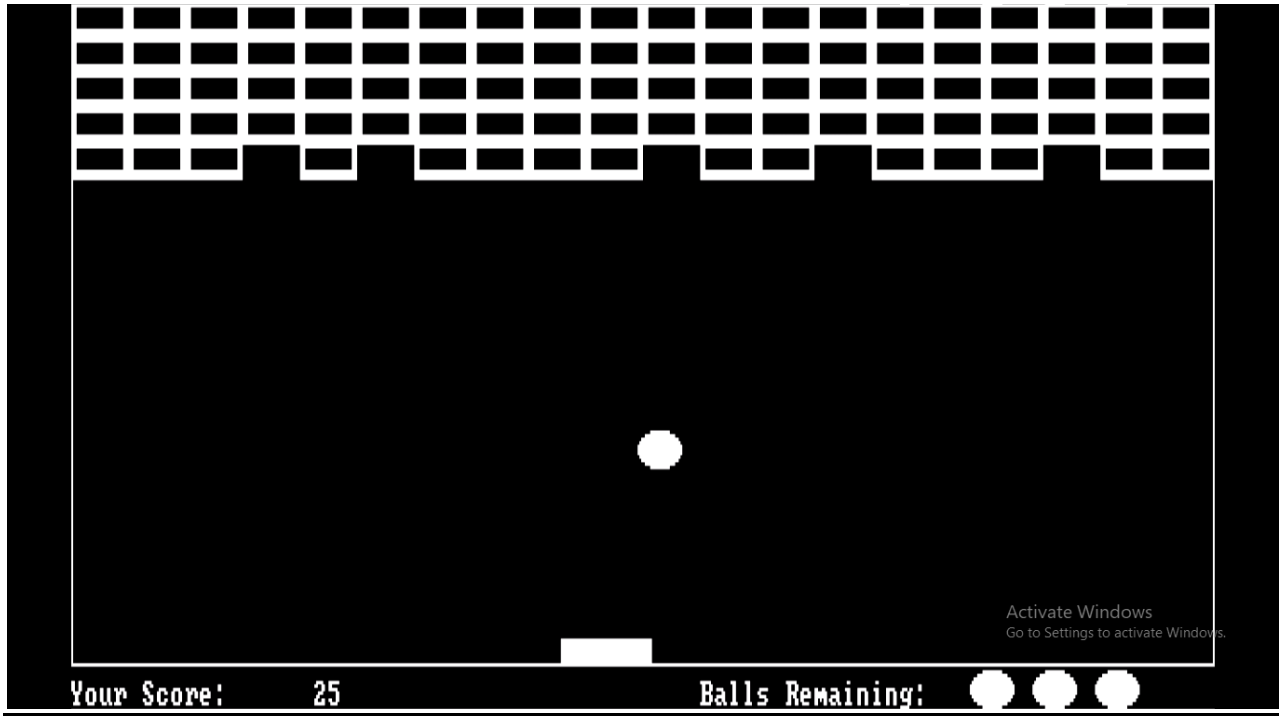


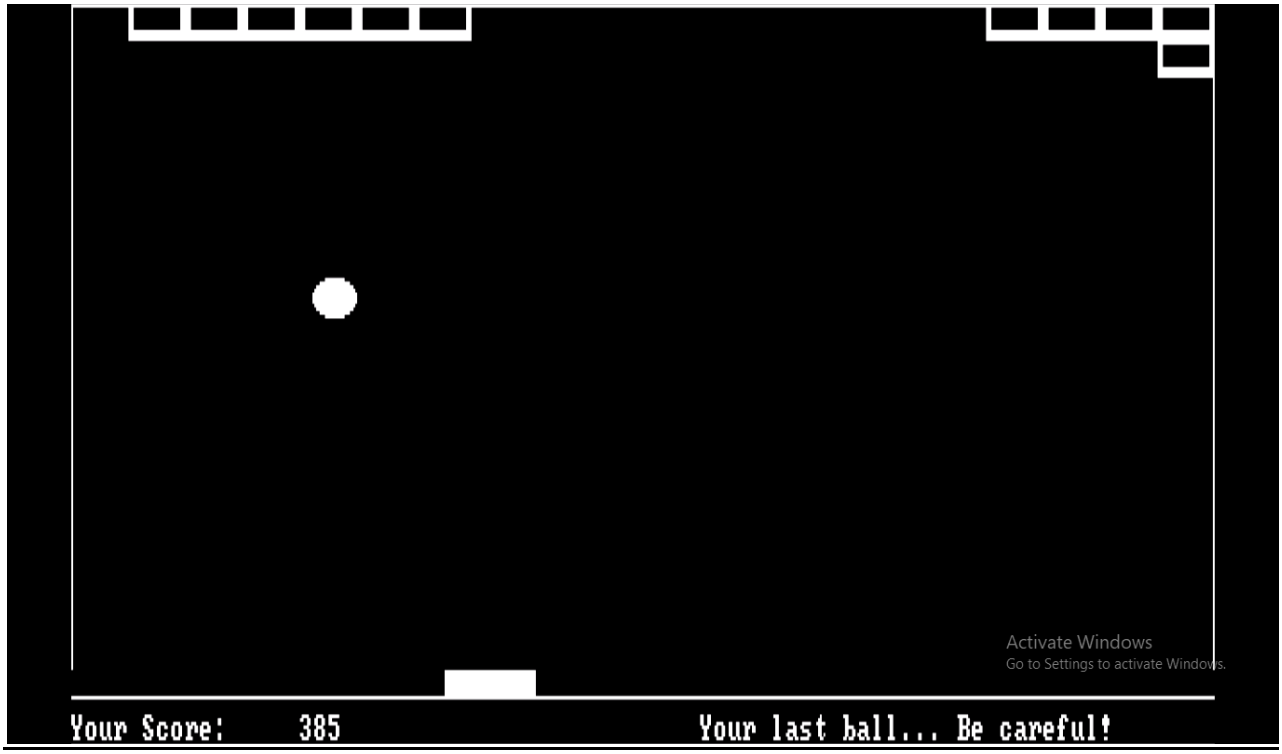
- **When we press N while selecting options for levels then the game appears to normal in a slow mode :-**

Game Starts

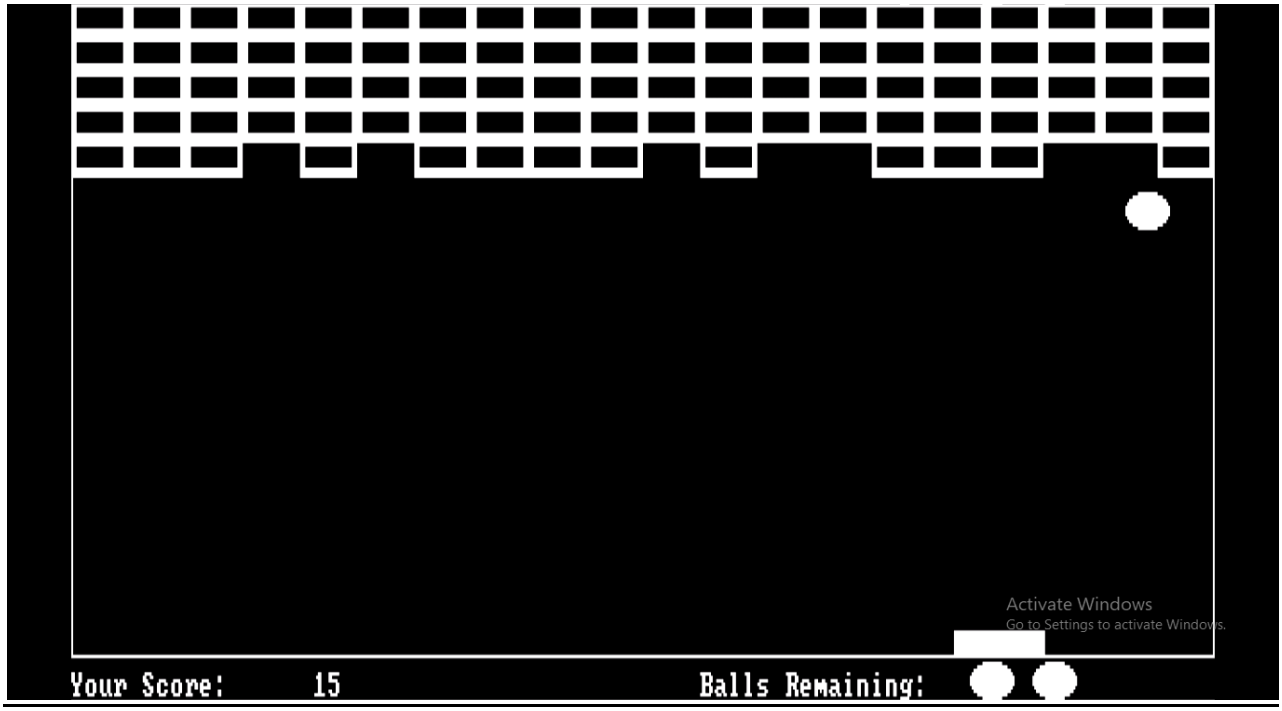


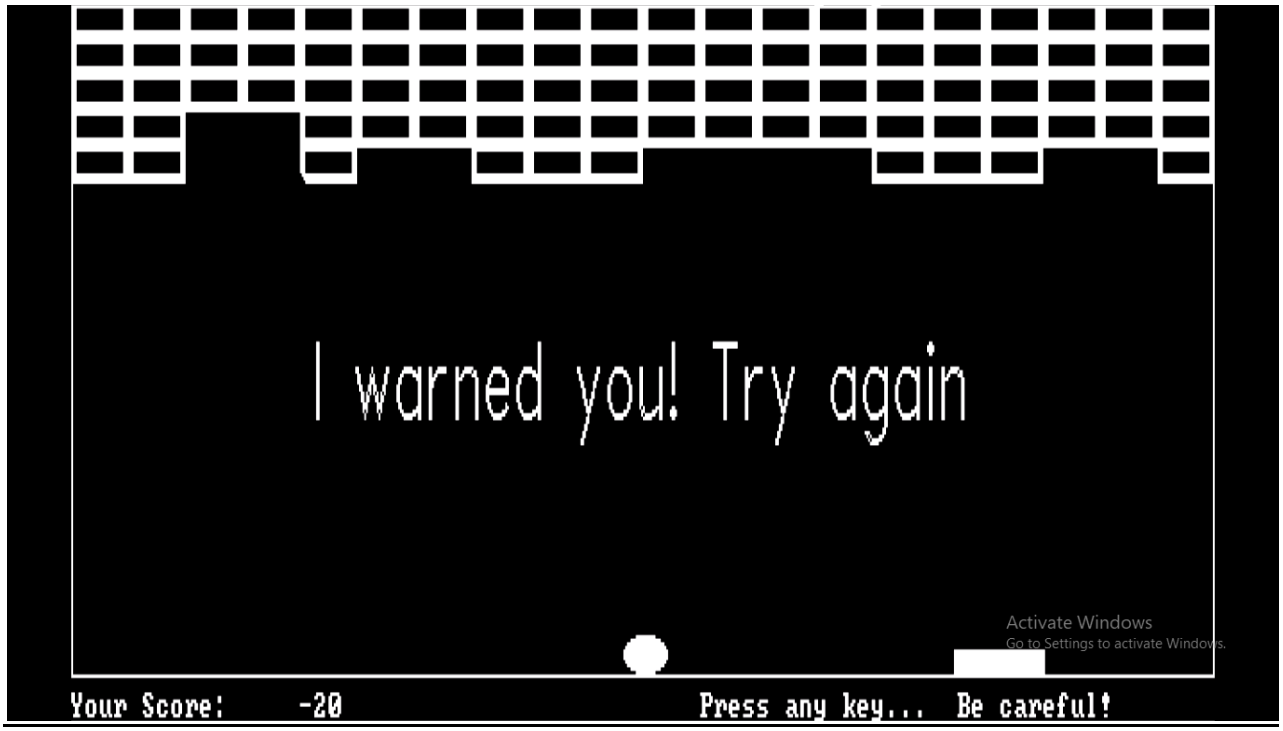
Game continues





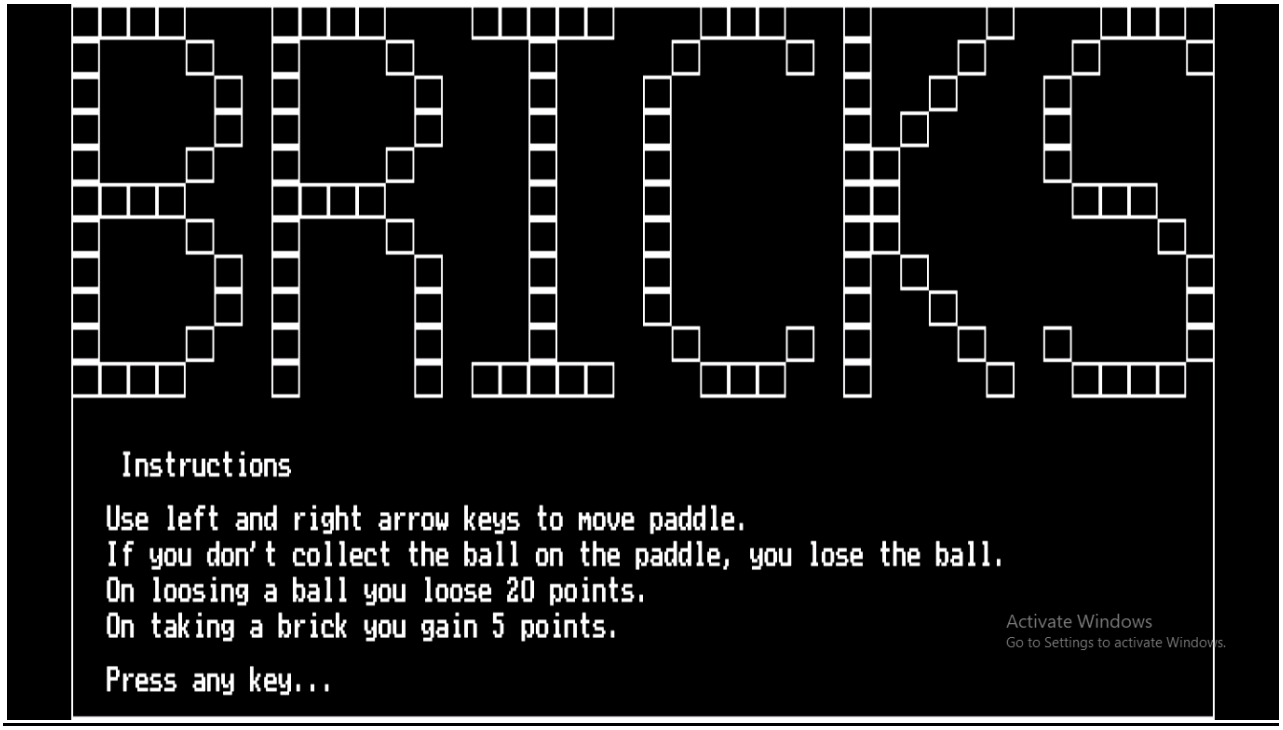
- **When we select Advanced A while selecting options for levels then the game appears to be same just the speed increases a little.**



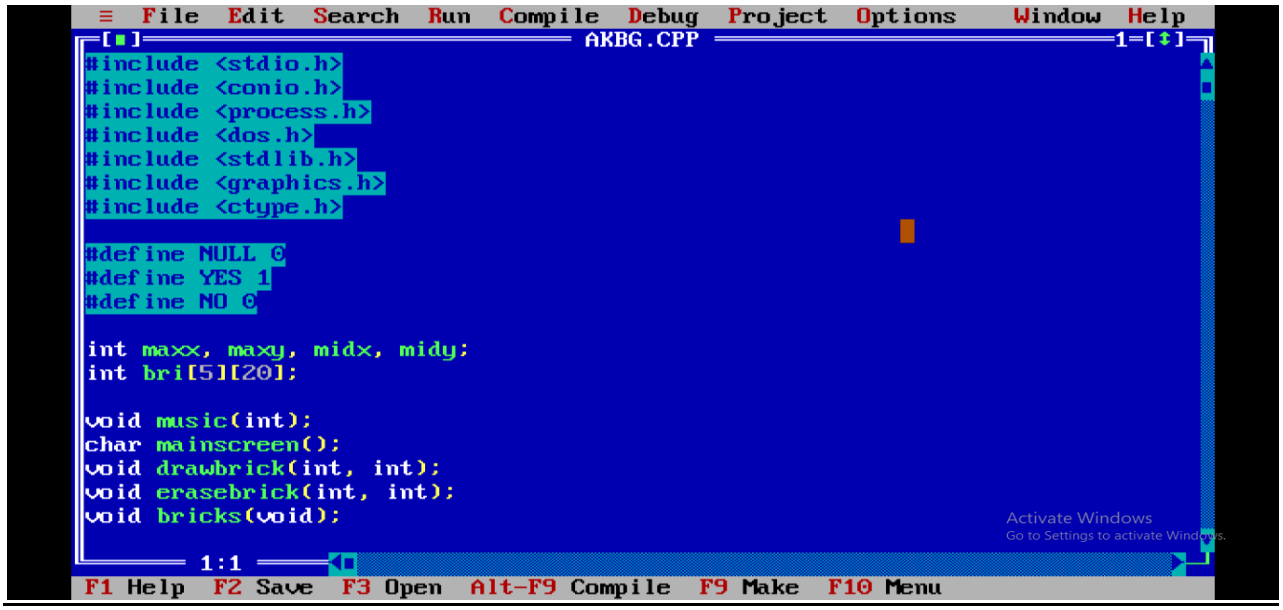


- **When we select Expert E while selecting options for levels then the game appears to be same just the speed increases a little bit more than the previous level i.e. speed of game is more than the advanced level.**

- **When we press I while selecting the options when we are told to select any one of the following we get :-**



- **When we press E while selecting the options when we are told to select any of the following options we return on our c compiler :-**



Title of Micro project: Bricks game in C.

1.0 Brief Description: In this project, we have Developed the Bricks game in C by creating objects and design by initializing the graphics system and then calculating the x and y co ordinates along with the center of the screen and calculating the various co-ordinates of the old and new positions of the ball and the bricks and also created 2 different sounds for the game along with the variations of levels which make the game totally interesting and enjoyable to play.

2.0 Aim of micro Project: The aim of the project is to create a Bricks game in C.

3.0Course Outcome Integrated : To learn and understand the basic knowledge of Operating System and its functions .

4.0Actual Procedure Followed: 1. Created and gathered information and sorted it in order.
2.Understood the concepts and planned the dates to do it .
3.Executed the plan properly whole gaining knowledge and discipline .

5.0 Actual Resources Used (mention the actual resources used.)

Sr. NO	Name of resource / material	Specification	Quantity	Remarks
1	Computer System	Windows 7 ultimate.	1	
2	Mobile	Samsung,lg,oppo.	1	
3	Web	Chrome browser to acquire basic knowledge about Operating System .	1	

6.0 Outputs of the Micro Projects : Thus we have studied and under stood all the basic concepts and functions in Operating System .Outputs of projects have attached at respective places .

7.0 Skill Developed/ Learning out of this Micro project
Throughout this project we developed some important skills like leadership quality, scheduling of the project and risk management. One of the most important skills we developed is the communication and coordination between our team members. We also learned planning skills, time management and adaptability.

- **Teacher Evaluation Sheet**

- **Name of students :** Ashish Dhananjay Kadam .

- **Enrollment No**: 1807110129

- **Name of programme :** Computer Engineering.

- **Semester**: Vth

- **Course Title**: Operating System

- **Code:** 22516

- **Title of Micro Project:** Bricks game in c

- **Course Outcomes Achieved:**

Course #	Course Outcome Statement
a.	Install operating system and configure it.
b.	Use operating system tools to perform various functions.
c.	Execute process command commands for performing process management operations.
d.	Apply scheduling algorithms to calculate turnaround time and average waiting time.
e.	Calculate efficiency of different memory management techniques.
f.	Apply file management techniques.

Evaluation as per suggested Rubric for Assessment of Micro Project

Sr. No	Characteristic to be assessed	Poor (Marks1-3)	Average (Marks 4 - 5)	Good (Marks 6 - 8)	Excellent (Marks 9 - 10)
1	Relevance to the course				
2	Literature Survey / Information collection				
3	Project Proposal				
4	Completion of the Target as per Project Proposal				
5	Analysis of data and representation				
6	Quality of Prototype/ Model				
7	Report preparation				
8	Presentation				
9	Defense				

Micro Project Evaluation Sheet

Process Assessment		Product Assessment		Total Marks
Part A - Project Proposal (2 Marks)	Project Methodology (2 Marks)	Par B - Project Report/ working Model (2 Marks)	Individual Presentation/ Viva (4 Marks)	10

Note: Every course teacher is expected to assign marks for group evaluation in first 3 columns and individual evaluation 4th column

Comment/ suggestion about team work/leadership/ interpersonal communication (If any)

……
……
……

……

Any other comment:

……
……
……

Name and Designation of the Faculty Member: Ms. Z.S.Sajjade

Signature: ……