C Project

On

The Tetris Game

Made by

Mintoo Kumar 303/COE/13

Nitesh Dodeja 312/COE/13

Piyush Singhal 319/COE/13

CODE

#include <stdlib.h>

#include <graphics.h>

#include <bios.h>

#include <stdio.h>

#define mDRAW 5

#define mLINE 25

#define mADOWN 7

#define mGEN 8

#define mLEFT 75

#define mRIGHT 77

#define mSPACE 57

#define mDOWN 80

#define mESC 1

#define MAXX 15

#define MAXY 30

#define BACKCOLOR BLACK

#define WINX 200

#define WINY 470

#define GAP 6

#define AREAX (WINX+GAP)

#define AREAY (WINY-GAP)

#define BOXW 15

int left, top, right, bottom;

int oldarea[MAXY+1][MAXX];

int area[MAXY+1][MAXX];

int actW,actH,actX,actY;

int curX,curY,curColor,curW,curH;

int newX,newY,newColor,newW,newH;

int active;

int TIMEINT;

int box[4][4], box2[4][4];

int FORCOLOR, FORCOLOR2;

int MESSAGE;

int score;

char ch;

int BOX[7][4][4]={

{

{1,1,1,1},

{0,0,0,0},

{0,0,0,0},

{0,0,0,0}

},{

{1,1,1,0},

{1,0,0,0},

{0,0,0,0},

{0,0,0,0}

},{

{1,1,1,0},

{0,0,1,0},

{0,0,0,0},

{0,0,0,0}

},{

{1,1,1,0},

{0,1,0,0},

{0,0,0,0},

{0,0,0,0}

},{

{1,1,0,0},

{0,1,1,0},

{0,0,0,0},

{0,0,0,0}

},{

{0,1,1,0},

{1,1,0,0},

{0,0,0,0},

{0,0,0,0}

},{

{1,1,0,0},

{1,1,0,0},

{0,0,0,0},

{0,0,0,0}

}

};

void initMenu();

void menu();

void level();

void highscore();

void howToplay();

void disHighscore();

void init();

void draw();

void nextTOcome();

int genBox();

int getKey();

void lineFull();

int moveLeft();

int moveRight();

int moveDown();

int rotate();

int getW();

int getH();

void clearOldBox();

void putNewBox();

int collisionRotate(int box[][4]);

void getMessage();

void dispatchMessage();

int timeCome();

void fallDown();

int gameOver();

void main()

{

int i,j,boxidx;

int driver=DETECT, mode=0;

randomize();

//registerbgidriver(EGAVGA\_driver);

initgraph(&driver,&mode,"");

xyz:

do

{

menu();

init();

boxidx=random(7);

FORCOLOR2=random(7)+1;

for(j=0;j<4;j++)

for(i=0;i<4;i++)

box2[j][i]=BOX[boxidx][j][i];

do

{

getMessage();

dispatchMessage();

}while(!gameOver());

highscore();

getch();

}while(1);

}

void initMenu()

{

cleardevice();

setfillstyle(SOLID\_FILL,GREEN);

bar(0,0,639,479);

left = getmaxx() / 2 - 100;

top = getmaxy() / 2 - 100;

right = getmaxx() / 2 + 100;

bottom = getmaxy() / 2 + 100;

setfillstyle(SOLID\_FILL,BLACK);

bar(left-5, top-5, right+5, bottom+5);

setfillstyle(SOLID\_FILL,YELLOW);

bar(left,top,right,bottom);

setcolor(0);

}

void menu()

{

char choice;

begin:

initMenu();

settextstyle(GOTHIC\_FONT, HORIZ\_DIR, 4);

outtextxy(getmaxx()/2-45, top+5, "MENU");

settextstyle(DEFAULT\_FONT, HORIZ\_DIR, 1.75);

outtextxy(getmaxx()/2-75, top+60, "1. START");

outtextxy(getmaxx()/2-75, top+70, "2. HIGH SCORE");

outtextxy(getmaxx()/2-75, top+80, "3. EXIT");

error1:

choice=getch();

switch(choice)

{

case '1' : level();

return;

case '2' : disHighscore();

goto begin;

case '3' : closegraph();

exit(0);

default : goto error1;

}

}

void level()

{

initMenu();

settextstyle(3, HORIZ\_DIR, 4);

outtextxy(getmaxx()/2-40, top+5, "LEVEL");

settextstyle(DEFAULT\_FONT, HORIZ\_DIR, 1.75);

outtextxy(getmaxx()/2-75, top+60, "1. BEGINNER");

outtextxy(getmaxx()/2-75, top+70, "2. PROFESSIONAL");

outtextxy(getmaxx()/2-75, top+80, "3. EXPERT");

error:

ch=getch();

switch(ch)

{

case '1' : TIMEINT=7;

break;

case '2' : TIMEINT=3;

break;

case '3' : TIMEINT=2;

break;

default : goto error;

}

}

void disHighscore()

{

FILE \*p;

int scr;

char str[6];

p= fopen("score.dat", "r");

initMenu();

settextstyle(3, HORIZ\_DIR, 4);

outtextxy(getmaxx()/2-80, top+5, "HIGHSCORE");

settextstyle(DEFAULT\_FONT, HORIZ\_DIR, 1.75);

fscanf(p, "%d ", &scr);

sprintf(str, "%d", scr);

outtextxy(getmaxx()/2-75, top+60, "1. BEGINNER:");

outtextxy(getmaxx()/2+25, top+60, str);

fscanf(p, "%d ", &scr);

sprintf(str, "%d", scr);

outtextxy(getmaxx()/2-75, top+70, "2. PROFESSIONAL:");

outtextxy(getmaxx()/2+60, top+70, str);

fscanf(p, "%d ", &scr);

sprintf(str, "%d", scr);

outtextxy(getmaxx()/2-75, top+80, "3. EXPERT:");

outtextxy(getmaxx()/2+10, top+80, str);

outtextxy(getmaxx()/2-100, top+240, "PRESS ANY KEY TO CONTINUE..");

fclose(p);

getch();

}

void highscore()

{

FILE \*p, \*q;

char i;

int scr;

p= fopen("score.dat", "r");

q= fopen("new.dat", "w");

for(i='1'; i<'4'; i++)

{

fscanf(p, "%d ", &scr);

if(i==ch)

if(score>scr)

scr=score;

fprintf(q, "%d ", scr);

}

fclose(p);

fclose(q);

remove("score.dat");

rename("new.dat", "score.dat");

}

void howToplay()

{

area[][]={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}

}

void disScore()

{

char scr[6];

setfillstyle(SOLID\_FILL,BLACK);

bar(30, 295, 160, 345);

bar(30, 235, 160, 285);

setfillstyle(SOLID\_FILL,YELLOW);

bar(35, 240, 155, 280);

bar(35, 300, 155, 340);

setcolor(0);

settextstyle(1, HORIZ\_DIR, 2);

sprintf(scr, "%c", ch);

outtextxy(37, 243, "LEVEL:");

outtextxy(102, 243, scr);

sprintf(scr, "%d", score);

outtextxy(37, 303, "SCORE:");

outtextxy(102, 303, scr);

}

void getMessage()

{

if(MESSAGE) return;

if(timeCome())

{

MESSAGE=mADOWN;

return;

}

if(bioskey(1))

{

MESSAGE=bioskey(0)>>8;

return;

}

}

void dispatchMessage()

{

switch(MESSAGE)

{

case mLEFT: moveLeft();break;

case mRIGHT: moveRight();break;

case mADOWN: moveDown();break;

case mSPACE: rotate();break;

case mDOWN: fallDown(); break;

case mDRAW: draw();break;

case mLINE: lineFull();break;

case mGEN: genBox();break;

case mESC: closegraph(); main();

default: MESSAGE=0;

}

}

void fallDown()

{

while(active)

{

moveDown(); draw();

}

MESSAGE=mLINE;

}

int timeCome()

{

static long tm, old;

tm=biostime(0,tm);

if(tm-old<TIMEINT) return 0;

else

{

old=tm; return 1;

}

}

void init()

{

int i,j,x1,y1,x2,y2;

int driver=DETECT, mode=0;

score=0;

randomize();

//registerbgidriver(EGAVGA\_driver);

initgraph(&driver,&mode,"");

cleardevice();

setfillstyle(SOLID\_FILL, GREEN);

bar(0,0,639,479);

x1=AREAX;

y1=AREAY-BOXW\*MAXY;

x2=AREAX+MAXX\*BOXW;

y2=AREAY;

rectangle(--x1,--y1,++x2,++y2);

setfillstyle(SOLID\_FILL,BLACK);

bar(++x1,++y1,--x2,--y2);

y1=AREAY-MAXY\*BOXW; y2=AREAY;

setcolor(DARKGRAY);

for(i=0;i<MAXX;i++)

{

x1=AREAX+i\*BOXW;

line(x1,y1,x1,y2);

}

x1=AREAX; x2=x1+MAXX\*BOXW;

for(j=0;j<MAXY;j++)

{

y1=AREAY-j\*BOXW;

line(x1,y1,x2,y1);

}

for(j=0;j<MAXY;j++)

for(i=0;i<MAXX;i++)

area[j][i]=oldarea[j][i]=0;

actX=0; actY=0; actW=MAXX-1; actH=MAXY-1;

draw();

disScore();

MESSAGE=mGEN;

}

void nextTOcome()

{ int i,j,x1=505, y1=275;

setfillstyle(SOLID\_FILL,BLACK);

bar(465, 235, 605, 345);

setfillstyle(SOLID\_FILL,YELLOW);

bar(470, 240, 600, 340);

for(j=0;j<2;j++)

for(i=0;i<4;i++)

if(box2[j][i]==1)

{

setfillstyle(SOLID\_FILL,FORCOLOR2);

bar(x1+i\*BOXW, y1+(2-j)\*BOXW, x1+(i+1)\*BOXW, y1+(1-j)\*BOXW);

setcolor(DARKGRAY);

rectangle(x1+i\*BOXW, y1+(2-j)\*BOXW, x1+(i+1)\*BOXW, y1+(1-j)\*BOXW);

}

}

int genBox()

{

int i,j,boxidx;

for(j=0;j<4;j++)

for(i=0;i<4;i++)

box[j][i]=box2[j][i];

FORCOLOR=FORCOLOR2;

boxidx=random(7); FORCOLOR2=random(7)+1;

for(j=0;j<4;j++)

for(i=0;i<4;i++)

box2[j][i]=BOX[boxidx][j][i];

nextTOcome();

curW=getW(); curH=getH();

curX=(MAXX+curW)/2;

if(curX+curW>=MAXX)curX=MAXX-1-curW;

curY=MAXY-1-curH;

newX=curX; newY=curY; actX=curX;actY=curY;

actW=newW=curW; actH=newH=curH;

active=1;

if(collision(box)) return 0;

putNewBox();

draw(); MESSAGE=0;

return 1;

}

void lineFull()

{

int row,col, rowEnd,full,i,j;

rowEnd=newY+newH;

if(rowEnd>=MAXY-1)

rowEnd=MAXY-2;

for(row=newY; row<=rowEnd;)

{

full=1;

for(col=0;col<MAXX;col++)

if(!area[row][col])

{full=0; break;}

if(!full)

{++row; continue;}

score+=((int)ch-48)\*100;

for(j=row; j<MAXY-1;j++)

for(i=0;i<MAXX;i++)

area[j][i]=area[j+1][i];

actX=0;actY=row; actW=MAXX-1; actH=MAXY-1-row;

draw();

disScore();

rowEnd--;

}

MESSAGE=mGEN;

}

void draw()

{

int row,col,x1,y1,x2,y2;

for(row=actY;row<=actY+actH;row++)

for(col=actX;col<=actX+actW;col++)

if(area[row][col]!=oldarea[row][col])

{

if(area[row][col]==0)

setfillstyle(SOLID\_FILL,BACKCOLOR);

else

setfillstyle(SOLID\_FILL,FORCOLOR);

x1=AREAX+col\*BOXW; x2=x1+BOXW;

y1=AREAY-(row+1)\*BOXW; y2=y1+BOXW;

bar(++x1,++y1,--x2,--y2);

oldarea[row][col]=area[row][col];

}

MESSAGE=0;

}

int moveLeft()

{

newX=curX-1; clearOldBox();

if(collision(box))

{

newX=curX;

putNewBox();

MESSAGE=0;

return 0;

}

putNewBox();

actW=curW+1; actX=curX=newX;

MESSAGE=mDRAW;

return 1;

}

int moveRight()

{

newX=curX+1; clearOldBox();

if(collision(box))

{

newX=curX;

putNewBox();

MESSAGE=0;

return 0;

}

putNewBox();

actW=curW+1; actX=curX; curX=newX;

MESSAGE=mDRAW;

return 1;

}

int moveDown()

{

int i,j;

newY=curY-1;

clearOldBox();

if(collision(box))

{

newY=curY;

putNewBox();

active=0;

MESSAGE=mLINE;

return 0;

}

putNewBox();

actH=curH+1; actY=newY; curY=newY;

MESSAGE=mDRAW;

return 1;

}

int rotate()

{

int newBox[4][4];

int i,j;

clearOldBox();

for(j=0;j<4;j++)

for(i=0;i<4;i++)

newBox[j][i]=0;

for(j=0;j<4;j++)

for(i=0;i<4;i++)

newBox[curW-i][j]=box[j][i];

newW=curH; newH=curW;

if(collisionRotate(newBox))

{

newW=curW; newH=curH; newX=curX; newY=curY;

putNewBox();

MESSAGE=0;

return 0;

}

for(j=0;j<4;j++)

for(i=0;i<4;i++)

box[j][i]=newBox[j][i];

putNewBox();

actH=newH>curH? newH:curH;

actW=curX+actH-newX;

actX=newX; actY=newY; curX=newX;

curY=newY; curW=newW; curH=newH;

MESSAGE=mDRAW;

return 1;

}

int getW()

{

int i,j;

for(i=3;i>0;i--)

for(j=0;j<4;j++)

if(box[j][i]) return i;

return 0;

}

int getH()

{

int i,j;

for(j=3;j>0;j--)

for(i=0;i<4;i++)

if(box[j][i]) return j;

return 0;

}

void clearOldBox()

{

int i,j;

for(j=0;j<=curH; j++)

for(i=0;i<=curW; i++)

if(box[j][i])

area[curY+j][curX+i]=0;

}

void putNewBox()

{

int i,j;

for(j=0;j<=newH;j++)

for(i=0;i<=newW;i++)

if(box[j][i])

area[newY+j][newX+i]=FORCOLOR;

}

int collision(int cbox[][4])

{

int i,j;

if(newX<0) return 1;

if(newX+newW>=MAXX) return 1;

if(newY<0) return 1;

for(j=0;j<=newH;j++)

for(i=0;i<=newW;i++)

if(area[newY+j][newX+i]&&cbox[j][i]) return 1;

return 0;

}

int collisionRotate(int cbox[][4])

{

int i,j;

if(newX+newW>=MAXX) newX=MAXX-1-newW;

if(newY+newH>=MAXY) newY=MAXY-1-newH;

if(collision(cbox)) return 1;

for(i=0;i<=newW;i++)

for(j=0;j<=newH;j++)

if(area[newY+j][newX+i])

{

newX-=newW-i+1; goto L;

}

L: return collision(cbox);

}

int gameOver()

{

if(!active &&(curY+curH>MAXY-3))

return 1;

else return 0;

}