**GAME CODE AVANGARIS**

**by Cristea codrin**

**STARTER**

/\*SALUT! Si bun venit la jocul meu!

Jocul este este impartit in mai multe stagii si substagii pentru

a individualiza si remedia mai usor bug-urile si erorile.

Stagiul 2 are un frate stagiul 21, iar stagiul 3 are 2 frati stagiul

31 si stagiul 31, deoarece nu merge sa fac apel la aceleasi stagii

in programe diferite daca le-am declarat in alt program.

Dar in principiu au acelasi cod.

DACA APAR UNELE PROBLEME PE PARCURS APASATI TASTA "SPACE" DE MAI MULTTE ORI!

DACA PROGRAMUL ARE O EROARE DE RENDER, DATI RESTART LA JOC!

DACA PROBELMA NU ESTE REZOLVATA SAU JOCUL MERGE PREA GREU, DATI UN RESTART LA LAPTOP SAU CALCULATOR!

DISTRACTIE PLACUTA! \*/

#include <bits/stdc++.h>

#include <graphics.h>

#include <windows.h>

#include <ctime>

#include <stdio.h>

#include <MMSystem.h>

#include "mainmenu.cpp"

using namespace std;

int main()

{

mainmenu();

return 0 ;

}

**MAIN MENU**

#include <bits/stdc++.h>

#include <graphics.h>

#include <windows.h>

#include <ctime>

#include <stdio.h>

#include <MMSystem.h>

#include "playmenu.cpp"

#include "rules.cpp"

using namespace std;

int mainmenu()

{

///INITIALIZARE PAGINA GRAFICA FULL SCREEEN

DWORD lungime\_ecran =GetSystemMetrics(SM\_CXSCREEN);

DWORD latime\_ecran =GetSystemMetrics(SM\_CYSCREEN);

/\*intializez 2 variabile pentru care le voi rezolutia ecranului\*/

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

///COORDONATELE CENTRULUI PAGINII GRAFICE

int mijx=lungime\_ecran/2;

int mijy=latime\_ecran/2;

/\* se salveaza coordonatele din centru ecranului\*/

///INITIALIZARE TEXT DE BUN VENIT;

char inceptext[256];

/\*sirul "inceptext" va memora textul de bun venit\*/

strcpy(inceptext,"WELCOME TO AVANGARDIS!");

/\*atribuirea sirului "inceptext" textul dorit\*/

int textx=textwidth(inceptext);

int texty=textheight(inceptext);

/\*cu cele 2 variabile vor salva dimensiunile textului

(lungimea si latimea acestuia) din sirul de caractere "inceptext"\*/

settextstyle(BOLD\_FONT,HORIZ\_DIR,7);

/\*cu aceasta comanda voi seta fontul textului, adica va fide tip bold (ingrosat),

desfasurarea (asezarea) va fi pe orizontala,

iar dimensiunea (marimea) va fi de 7 ori mai mare decat ar fi normal \*/

setcolor(YELLOW);

/\*cu aceasta comanda voi seta culoarea pe care o va avea textul, in acest caz galben (YELLOW) \*/

outtextxy(mijx-5/2\*textx,mijy-17\*texty,inceptext);

/\*cu aceasta comanda voi afisa textul incepand de la un punct fix

motivul pentru care incep afisarea textului imediat ce l-am initializat,

este pentru a creea un suport pentru celelate texte

astfel, daca nu afisam textul imediat ce l-am creat,

textele de toate butoanele s-ar fi situat in partea dreapta a butoanelor,deci nu ar fi centrate

\*/

int distbut=latime\_ecran/9;

/\* disbut reprezinta distanta dintre butoane masurata in pixeli \*/

/// BUTONUL PLAY

int lungbp=lungime\_ecran/5;

/\*lungbp reprezinta lungimea pe care o are butonul de play masurata in pixeli \*/

int latbp=latime\_ecran/6;

/\*latbut reprezinta latimea pe care o are butonul de play masurata in pixeli \*/

int bpix=mijx-lungbp/2;

int bpiy=mijy-latbp/2-distbut;

int bpfx=mijx+lungbp/2;

int bpfy=mijy+latbp/2-distbut;

/\* aceste 4 variabile reprezinta coordonatele primului si ultimului varf

aflate pe diagonala principala a poligolului pe care il vor forma \*/

char playtext[256];

/\* declar sirul pentru care voi atribui textul de "PLAY" \*/

strcpy(playtext,"PLAY");

/\* comanda de atribuire a textului \*/

int textpx=textwidth(playtext);

int textpy=textheight(playtext);

/\* 2 variabile prin care memorez dimensiunile textului \*/

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

/\* comanda prin care i se atribuyie textului un anumit stil, font ingrosat,

desfasurare orizontala, dimensinea caracterelor e de 5 ori mai mare decat ar fi in mod normal \*/

setcolor(YELLOW);

/\* comanda de atribuire a unei culori textului \*/

outtextxy(bpix+(bpfx-bpix)/2-textpx/2,bpiy+(bpfy-bpiy)/2-textpy/2,playtext);

/\* comanda pentru afisarea textului, in asa maniera incat sa fie centrata in interiorul butonului

motivul pentru care incep afisarea textului imediat ce l-am initializat,

este pentru a creea un suport de stabilitate pentru textul aflat in cele 2 animatii

astfel, daca nu afisam textul imediat ce l-am creat,

textul butonului s-ar fi situat in partea dreapta a acestuia,deci nu ar fi centrat

\*/

/// BUTONUL RULES

int lungbr=lungime\_ecran/5;

int latbr=latime\_ecran/6;

int brix=mijx-lungbr/2;

int briy=mijy-latbr/2+distbut;

int brfx=mijx+lungbr/2;

int brfy=mijy+latbr/2+distbut;

char ruletext[256];

strcpy(ruletext,"RULES");

int textrx=textwidth(ruletext);

int textry=textheight(ruletext);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(brix+(brfx-brix)/2-textrx/2,briy+(brfy-briy)/2-textry/2,ruletext);

/\* se aplica aceeasi mecanica de la BUTONUL PLAY \*/

/// BUTONUL QUIT

int lungbc=lungime\_ecran/5;

int latbc=latime\_ecran/6;

int bcix=mijx-lungbr/2;

int bciy=mijy-latbr/2+3\*distbut;

int bcfx=mijx+lungbr/2;

int bcfy=mijy+latbr/2+3\*distbut;

char credittext[256];

strcpy(credittext,"QUIT");

int textcx=textwidth(ruletext);

int textcy=textheight(ruletext);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(bcix+(bcfx-bcix)/2-textcx/3\*2,bciy+(bcfy-bciy)/2-textcy/2,credittext);

/\* se aplica aceeasi mecanica de la BUTONUL PLAY \*/

int page=0;

/\* page va memora pagina pecare se va desena \*/

int timeless=2;

/\* timeless va memora viteza de redare a desenelor \*/

int viteza=5;

/\* viteza va memora viteza de deplasare a butoanelor in decursul animatiei \*/

int cabral=0;

/\* variabila cabral va memora optiunea utilizatorului\*/

///SETARI CURSOR

POINT cursor\_poz;

/\* cursor\_poz va memora pozitia cursorului mouse-ului la un anumit moment \*/

///APARITIE TEXT

for(int i=1; i<=mijy-17\*texty; i=i+viteza-2)

{

/\* structura repetitiva for este utilizata pentru a face textul sa se deplaseze

de la partea de sus a ecranului, pana in pozitia stabilita in initializare\*/

setactivepage(page);

/\* seteaza pagina pe care se va desena \*/

setvisualpage(1-page);

/\* seteaza ca pagina pe care o va vizualiza observatorul sa fie cea precedenta \*/

cleardevice();

/\* sterge tot ce este pe pagina activa \*/

readimagefile("bk.jpg",0,0,lungime\_ecran,latime\_ecran);

/\* backgroundul programului va fi o imat=gine cu dimensiunea de 1x1 pixeli

pentru a face citirea si afisarea mai rapida,

astfel animatia va derula mai usor si mai eficient \*/

settextstyle(BOLD\_FONT,HORIZ\_DIR,7);

setcolor(YELLOW);

setbkcolor(RED);

/\*cu aceasta comanda voi seta culoarea backgroundului pe care o va avea textul, in acest caz rosu (RED)

pentru a evita spatiul negru din jurul textlui \*/

outtextxy(mijx-5/2\*textx,i,inceptext);

/\* aceeasi mecanica ca si la initializare, doar ca se afiseaza textul la o anumita pozitie i\*/

page=1-page;

/\* page va avea doar 2 valori, astfel se vor folosi doar 2 pagini, una anterioada care va fi afisata utilizatorului,

iar a doua pagina va fi stearga si redesenata

tranzitia dintre cele 2 pagini va elimita efectul de glich pecare l-ar fi avut animatia \*/

delay(timeless);

/\* aceasta comanda ingheata programul pentru cateva milisecunde,

in functie de ce valoare are timeless \*/

}

///APARITIE PLAY

for(int i=1; i<=bpix; i=i+viteza)

{

/\* structura repetitiva for este utilizata pentru a face butonul sa se deplaseze

din partea stanga a ecranului, pana in pozitia stabilita in initializare\*/

setactivepage(page);

setvisualpage(1-page);

cleardevice();

readimagefile("bk.jpg",0,0,lungime\_ecran,latime\_ecran);

///AFISARE TEXT

settextstyle(BOLD\_FONT,HORIZ\_DIR,7);

setcolor(YELLOW);

/\* pricipiul este acelasi ca si la TEXT SPAWN \*/

setbkcolor(RED);

outtextxy(mijx-5/2\*textx,mijy-17\*texty,inceptext);

/\* se afiseaza textul de bun venit in pozitia optima \*/

///PLAY

setbkcolor(GREEN);

setfillstyle(SOLID\_FILL,GREEN);

bar(i,bpiy,i+lungbp,bpfy);

/\* se stabileste tipul de culoare a polinomului ce va reprezenta butonul de PLAY

se constuieste un dreptunghi de la primul varf pana la ultimul varf de pe diagonala principala \*/

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(i+lungbp/2-textpx/3,bpiy+(bpfy-bpiy)/2-textpy/3,playtext);

/\* aceeasi mecanica ca si la initializare, doar ca se afiseaza textul la o anumita pozitie i\*/

page=1-page;

delay(timeless);

/\* pricipiul este acelasi ca si la APARITIE TEXT \*/

}

///APARITIE RULE

for(int i=lungime\_ecran-lungbc; i>=brix; i=i-viteza)

{

setactivepage(page);

setvisualpage(1-page);

cleardevice();

readimagefile("bk.jpg",0,0,lungime\_ecran,latime\_ecran);

///AFISARE TEXT

settextstyle(BOLD\_FONT,HORIZ\_DIR,7);

setcolor(YELLOW);

setbkcolor(RED);

outtextxy(mijx-5/2\*textx,mijy-17\*texty,inceptext);

///AFISARE BUTON PLAY

setfillstyle(SOLID\_FILL,GREEN);

bar(bpix,bpiy,bpfx,bpfy);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

setbkcolor(GREEN);

outtextxy(bpix+(bpfx-bpix)/2-textpx/3,bpiy+(bpfy-bpiy)/2-textpy/3,playtext);

/\* se afiseaza butonul de PLAY in pozitia initiala\*/

bar(i,briy,i+lungbr,brfy);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

setbkcolor(GREEN);

outtextxy(i+lungbp/2-textrx/2,briy+(brfy-briy)/2-textry/2,ruletext);

page=1-page;

delay(timeless);

}

/\* in rest principiul este acelasi ca si in cazul APARITIE PLAY \*/

/// APARITIE QUIT

for(int i=1; i<=bcix; i=i+viteza)

{

setactivepage(page);

setvisualpage(1-page);

cleardevice();

readimagefile("bk.jpg",0,0,lungime\_ecran,latime\_ecran);

///AFISARE TEXT

settextstyle(BOLD\_FONT,HORIZ\_DIR,7);

setcolor(YELLOW);

setbkcolor(RED);

outtextxy(mijx-5/2\*textx,mijy-17\*texty,inceptext);

/// AFISARE BUTON PLAY SI RULES

setfillstyle(SOLID\_FILL,GREEN);

bar(bpix,bpiy,bpfx,bpfy);

bar(brix,briy,brfx,brfy);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

setbkcolor(GREEN);

outtextxy(bpix+(bpfx-bpix)/2-textpx/3,bpiy+(bpfy-bpiy)/2-textpy/3,playtext);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(brix+(brfx-brix)/2-textrx/2,briy+(brfy-briy)/2-textry/2,ruletext);

/\* se afiseaza butonul de PLAY si butonul de RULES in pozitia initiala\*/

bar(i,bciy,i+lungbc,bcfy);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(i+lungbp/2-textcx/2,bciy+(bcfy-bciy)/2-textcy/2,credittext);

page=1-page;

delay(timeless);

}

/\* in rest principiul este acelasi ca si in cazul APARITIE PLAY \*/

///FREEROOM

while(cabral==0)

{

/\* cat timp utilizatorul nu a selectat vreun buton, se genereaza in continuu se deseneaza o pagina \*/

setactivepage(page);

setvisualpage(1-page);

cleardevice();

/\* acelasi principiu ca si la animatie \*/

GetCursorPos(&cursor\_poz);

/\* aceasta comanda reda pozitia la care se afla cursorul mouse-ului in acel moment \*/

readimagefile("bk.jpg",0,0,lungime\_ecran,latime\_ecran);

///AFISARE TEXT

settextstyle(BOLD\_FONT,HORIZ\_DIR,7);

setcolor(YELLOW);

setbkcolor(RED);

outtextxy(mijx-5/2\*textx,mijy-17\*texty,inceptext);

///AFISAREA BUTONULUI DE PLAY

setfillstyle(SOLID\_FILL,GREEN);

bar(bpix,bpiy,bpfx,bpfy);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

setbkcolor(GREEN);

outtextxy(bpix+(bpfx-bpix)/2-textpx/3,bpiy+(bpfy-bpiy)/2-textpy/3,playtext);

///AFISAREA BUTONULUI DE RULES

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

bar(brix,briy,brfx,brfy);

outtextxy(brix+(brfx-brix)/2-textrx/2,briy+(brfy-briy)/2-textry/2,ruletext);

setfillstyle(SOLID\_FILL,GREEN);

///AFISAREA BUTONULUI DE QUIT

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

bar(bcix,bciy,bcfx,bcfy);

outtextxy(bcix+(bcfx-bcix)/2-textcx/2,bciy+(bcfy-bciy)/2-textcy/2,credittext);

/\*acelasi pricipiu ca si la initializare \*/

if(GetAsyncKeyState(VK\_LBUTTON))

{

/\* daca s-a apasat cumva click, verifica urmatoarele 3 lucruri \*/

if(cursor\_poz.x>=bpix&&cursor\_poz.x<=bpfx&&cursor\_poz.y>=bpiy&&cursor\_poz.y<=bpfy)

{

/\* daca cursorul se afla in interiorul butonului de PLAY,

daca da atunci optiunea utilizatorului (cabral) devine 1 (primul buton) \*/

cabral=1;

}

if(cursor\_poz.x>=brix&&cursor\_poz.x<=brfx&&cursor\_poz.y>=briy&&cursor\_poz.y<=brfy)

{

/\* daca cursorul se afla in interiorul butonului de RULES,

daca da atunci optiunea utilizatorului (cabral) devine 2 (al doilea buton) \*/

cabral=2;

}

if(cursor\_poz.x>=bcix&&cursor\_poz.x<=bcfx&&cursor\_poz.y>=bciy&&cursor\_poz.y<=bcfy)

{

/\* daca cursorul se afla in interiorul butonului de QUIT,

daca da programul se inchide, deci si aplicatia se va inchide \*/

return 0;

}

}

page=1-page;

delay(10);

}

/\* in rest este aceeasi mecanica ca si la animatie \*/

closegraph();

/\* se inchide toate paginile deschise, astfel sa nu fie deschise

prea multe pagini, deci ca sa nu scada performanta jocului\*/

if(cabral==1)

playmenu();

else if(cabral==2)

{

rules();

mainmenu();

}

/\*daca optiunea utilizatorului este 1 atunci se dechide meniul de play

daca optiunea utilizatorului este 2 atunci se va deschide meniul de rules

si se va relua programul din meniul principal\*/

return 0;

}

**PLAY MENU**

#include <bits/stdc++.h>

#include <windows.h>

#include <graphics.h>

#include "stagiu1.cpp"

#include "stagiu2.cpp"

#include "stagiu3.cpp"

using namespace std;

int playmenu()

{

///INITIALIZARE PAGINA GRAFICA FULL SCREEEN

DWORD lungime\_ecran =GetSystemMetrics(SM\_CXSCREEN);

DWORD latime\_ecran =GetSystemMetrics(SM\_CYSCREEN);

/\*intializez 2 variabile pentru care le voi rezolutia ecranului\*/

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

///COORDONATELE CENTRULUI PAGINII GRAFICE

int mijx=lungime\_ecran/2;

int mijy=latime\_ecran/2;

/\* se salveaza coordonatele din centru ecranului\*/

int lbut=lungime\_ecran/6;

/\*lbut reprezinta lungimea pe care o are butonul masurata in pixeli \*/

int ltbut=latime\_ecran/6;

/\*ltbut reprezinta latimea pe care o are butonul masurata in pixeli \*/

int disbut=latime\_ecran/9.1;

/\* disbut reprezinta distanta dintre butoane masurata in pixeli \*/

struct button

{

int xi;

int yi;

int xf;

int yf;

} but;

but.xi=mijx/3;

but.yi=3\*mijy/6;

but.xf=but.xi+lbut;

but.yf=but.yi+ltbut;

/\* folosesc aceasta structura pentru a creea coodonatele primului si ultimului varf,

aflate pe diagonala principala, al primului buton

creez doar primul buton in asa fel incat toate celelalte butoane sa depinda depozitia

si dimensiule primului buton, pentru a fi mai usor de editat

\*/

///BUTON 1

setfillstyle(SOLID\_FILL,GREEN);

bar(but.xi,but.yi,but.xf,but.yf);

/\* se stabileste tipul de culoare a polinomului ce va reprezenta butonul

se constuieste un dreptunghi de la primul varf pana la ultimul varf de pe diagonala principala \*/

char lvl1text[256];

/\*sirul "lvl1text" va memora textul corespunzator \*/

strcpy(lvl1text,"LEVEL 1");

/\*atribuirea sirului "lvl1text" textul dorit\*/

int text1x=textwidth(lvl1text);

int text1y=textheight(lvl1text);

/\*cu cele 2 variabile vor salva dimensiunile textului

(lungimea si latimea acestuia) din sirul de caractere "inceptext"\*/

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

/\* comanda prin care i se atribuyie textului un anumit stil, font ingrosat,

desfasurare orizontala, dimensinea caracterelor e de 5 ori mai mare decat ar fi in mod normal \*/

setcolor(YELLOW);

/\*cu aceasta comanda voi seta culoarea pe care o va avea textul, in acest caz galben (YELLOW) \*/

outtextxy(but.xi+(but.xf-but.xi)/2-5\*text1x/3,but.yi+(but.yf-but.yi)/2-4\*text1y/3,lvl1text);

/\*cu aceasta comanda voi afisa textul incepand de la un punct fix

motivul pentru care incep afisarea textului imediat ce l-am initializat,

este pentru a creea un suport pentru celelate texte

astfel, daca nu afisam textul imediat ce l-am creat,

textele de toate butoanele s-ar fi situat in partea dreapta a butoanelor,deci nu ar fi centrate

\*/

///BUTON 2

setfillstyle(SOLID\_FILL,GREEN);

setcolor(GREEN);

bar(but.xi+lbut+disbut,but.yi,but.xf+disbut+lbut,but.yf);

char lvl2text[256];

strcpy(lvl2text,"LEVEL 2");

int text2x=textwidth(lvl2text);

int text2y=textheight(lvl2text);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(but.xi+lbut+disbut+(but.xf-but.xi)/2-5\*text1x/3,but.yi+(but.yf-but.yi)/2-4\*text1y/3,lvl2text);

/\* se aplica acelasi principiu ca si in cazul BUTON 1, BUTON 2 fiind o derivata a acestuia\*/

///BUTON 3

setfillstyle(SOLID\_FILL,GREEN);

setcolor(GREEN);

bar(but.xi+2\*lbut+2\*disbut,but.yi,but.xf+2\*disbut+2\*lbut,but.yf);

char lvl3text[256];

strcpy(lvl3text,"LEVEL 3");

int text3x=textwidth(lvl3text);

int text3y=textheight(lvl3text);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(but.xi+2\*lbut+2\*disbut+(but.xf-but.xi)/2-5\*text1x/3,but.yi+(but.yf-but.yi)/2-4\*text1y/3,lvl3text);

/\* se aplica acelasi principiu ca si in cazul BUTON 1, BUTON 3 fiind o derivata a acestuia\*/

///BUTON QUIT

setfillstyle(SOLID\_FILL,GREEN);

setcolor(GREEN);

int lbutb=lungime\_ecran/13;

int ltbutb=latime\_ecran/10;

bar(10,10,2\*lbutb,ltbutb);

char backtext[256];

strcpy(backtext,"QUIT");

int textbx=textwidth(backtext);

int textby=textheight(backtext);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(lbutb/2-5\*text1x/3+10,ltbutb/2-4\*text1y/3,backtext);

/\* se aplica acelasi principiu ca si in cazul BUTON 1, BUTON QUIT fiind o derivata a acestuia\*/

POINT mouse;

/\* mouse va memora pozitia cursorului mouse-ului la un anumit moment \*/

int timeless=10;

/\* timeless va memora viteza de redare a desenelor \*/

int viteza=3;

/\* viteza va memora viteza de deplasare a butoanelor in decursul animatiei \*/

int cabral=0;

/\* variabila cabral va memora obtiunea utilizatorului\*/

int page=0;

/\* page va memora pagina pecare se va desena \*/

///ANIMATIIE

for(int i=1; i<=but.yi; i=i+viteza)

{

/\* structura repetitiva for este utilizata pentru a face ca cele trei butoane sa se deplaseze

de la partea de sus a ecranului, pana in pozitia stabilita in initializare\*/

setactivepage(page);

/\* seteaza pagina pe care se va desena \*/

setvisualpage(1-page);

/\* seteaza ca pagina pe care o va vizualiza observatorul sa fie cea precedenta \*/

cleardevice();

/\* sterge tot ce este pe pagina activa \*/

readimagefile("bk.jpg",0,0,lungime\_ecran,latime\_ecran);

/\* backgroundul programului va fi o imat=gine cu dimensiunea de 1x1 pixeli

pentru a face citirea si afisarea mai rapida,

astfel animatia va derula mai usor si mai eficient \*/

///BUTON 1

setfillstyle(SOLID\_FILL,GREEN);

setcolor(GREEN);

bar(but.xi,i,but.xf,i+ltbut);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setbkcolor(GREEN);

/\*cu aceasta comanda voi seta culoarea backgroundului pe care o va avea textul, in acest caz verde (GREEN)

pentru a evita spatiul negru din jurul textlui \*/

setcolor(YELLOW);

outtextxy(but.xi+(but.xf-but.xi)/2-5\*text1x/3,i+ltbut/2-4\*text1y/3,lvl1text);

///BUTON 2

setfillstyle(SOLID\_FILL,GREEN);

setcolor(GREEN);

bar(but.xi+lbut+disbut,i,but.xf+disbut+lbut,i+ltbut);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

setbkcolor(GREEN);

outtextxy(but.xi+lbut+disbut+(but.xf-but.xi)/2-5\*text1x/3,i+ltbut/2-4\*text1y/3,lvl2text);

///BUTON 3

setfillstyle(SOLID\_FILL,GREEN);

setcolor(GREEN);

bar(but.xi+2\*lbut+2\*disbut,i,but.xf+2\*disbut+2\*lbut,i+ltbut);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

setbkcolor(GREEN);

outtextxy(but.xi+2\*lbut+2\*disbut+(but.xf-but.xi)/2-5\*text1x/3,i+ltbut/2-4\*text1y/3,lvl3text);

/\* stilul celor trei butoane este acelasi, difera doar pozitia acestora in functie de distanta

la care se afla de fiecare buton, astfel doar primul buton va fi mutat, restul se vor situa la

acelasi nivel cu primu, dar la coordonate orizontale diferite, acestea ramanand aceleasi

in rest mecanica este aceeasi ca si la intializare \*/

page=1-page;

/\* page va avea doar 2 valori, astfel se vor folosi doar 2 pagini, una anterioada care va fi afisata utilizatorului,

iar a doua pagina va fi stearga si redesenata

tranzitia dintre cele 2 pagini va elimita efectul de glich pecare l-ar fi avut animatia \*/

delay(timeless);

/\* aceasta comanda ingheata programul pentru cateva milisecunde,

in functie de ce valoare are timeless \*/

}

///FREEDOM

while(cabral==0)

{

/\* cat timp utilizatorul nu a selectat vreun buton, se genereaza in continuu se deseneaza o pagina \*/

setactivepage(page);

setvisualpage(1-page);

cleardevice();

/\* acelasi principiu ca si la animatie \*/

GetCursorPos(&mouse);

/\* aceasta comanda reda pozitia la care se afla cursorul mouse-ului in acel moment \*/

readimagefile("bk.jpg",0,0,lungime\_ecran,latime\_ecran);

///BUTON 1

setfillstyle(SOLID\_FILL,GREEN);

setcolor(GREEN);

bar(but.xi,but.yi,but.xf,but.yf);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

setbkcolor(GREEN);

outtextxy(but.xi+(but.xf-but.xi)/2-5\*text1x/3,but.yi+(but.yf-but.yi)/2-4\*text1y/3,lvl1text);

///BUTON 2

setfillstyle(SOLID\_FILL,GREEN);

setcolor(GREEN);

bar(but.xi+lbut+disbut,but.yi,but.xf+disbut+lbut,but.yf);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

setbkcolor(GREEN);

outtextxy(but.xi+lbut+disbut+(but.xf-but.xi)/2-5\*text1x/3,but.yi+(but.yf-but.yi)/2-4\*text1y/3,lvl2text);

///BUTON 3

setfillstyle(SOLID\_FILL,GREEN);

setcolor(GREEN);

bar(but.xi+2\*lbut+2\*disbut,but.yi,but.xf+2\*disbut+2\*lbut,but.yf);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

setbkcolor(GREEN);

outtextxy(but.xi+2\*lbut+2\*disbut+(but.xf-but.xi)/2-5\*text1x/3,but.yi+(but.yf-but.yi)/2-4\*text1y/3,lvl3text);

///BUTON BACK

setfillstyle(SOLID\_FILL,GREEN);

setcolor(GREEN);

bar(10,10,2\*lbutb,ltbutb);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(2\*lbutb/2-textbx/2,ltbutb/2-textby/2,backtext);

/\*acelasi pricipiu ca si la initializare \*/

if(GetAsyncKeyState(VK\_LBUTTON))

{

/\* daca s-a apasat cumva click, verifica urmatoarele 4 lucruri \*/

if(mouse.x>=but.xi&&mouse.x<=but.xf&&mouse.y>=but.yi&&mouse.y<=but.yf)

{

/\* daca cursorul se afla in interiorul butonului de LEVEL 1,

daca da atunci optiunea utilizatorului (cabral) devine 1 (primul buton) \*/

cabral=1;

}

if(mouse.x>=but.xi+lbut+disbut&&mouse.x<=but.xf+disbut+lbut&&mouse.y>=but.yi&&mouse.y<=but.yf)

{

/\* daca cursorul se afla in interiorul butonului de LEVEL 2,

daca da atunci optiunea utilizatorului (cabral) devine 2 (al doilea buton) \*/

cabral=2;

}

if(mouse.x>=but.xi+2\*lbut+2\*disbut&&mouse.x<=but.xf+2\*disbut+2\*lbut&&mouse.y>=but.yi&&mouse.y<=but.yf)

{

/\* daca cursorul se afla in interiorul butonului de LEVEL 3,

daca da atunci optiunea utilizatorului (cabral) devine 3 (al treilea buton) \*/

cabral=3;

}

if(mouse.x>=10&&mouse.x<=2\*lbutb&&mouse.y>=10&&mouse.y<=ltbutb)

{

/\* daca cursorul se afla in interiorul butonului de QUIT,

daca da programul se inchide, deci si aplicatia se va inchide \*/

return 0;

}

}

page=1-page;

delay(timeless);

}

/\* in rest este aceeasi mecanica ca si la animatie \*/

closegraph();

/\* se inchide toate paginile deschise, astfel sa nu fie deschise

prea multe pagini, deci ca sa nu scada performanta jocului\*/

if(cabral==1)

stagiu1();

else if(cabral==2)

stagiu2();

else if(cabral==3)

stagiu3();

/\*daca optiunea utilizatorului este 1 atunci se dechide stagiul 1

daca optiunea utilizatorului este 2 atunci se dechide stagiul 2

daca optiunea utilizatorului este 3 atunci se dechide stagiul 3

\*/

return 0;

}

**RULES MENU**

#include <bits/stdc++.h>

#include <graphics.h>

#include <windows.h>

using namespace std;

int rules()

{

int cabral=0;

///INITIALIZARE ECRAN

DWORD lungime\_ecran=GetSystemMetrics(SM\_CXSCREEN);

DWORD latime\_ecran=GetSystemMetrics(SM\_CYSCREEN);

/\*intializez 2 variabile pentru care le voi rezolutia ecranului\*/

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

///COORDONATELE CENTRULUI PAGINII GRAFICE

int mijx=lungime\_ecran/2;

int mijy=latime\_ecran/2;

/\* se salveaza coordonatele din centru ecranului\*/

int lungime=lungime\_ecran/6;

/\*lungime reprezinta lungimea pe care o are butonul masurata in pixeli \*/

int latime=latime\_ecran/7;

/\*latime reprezinta lungimea pe care o are butonul masurata in pixeli \*/

struct buton

{

int xi;

int yi;

} br;

br.xi=mijx/2;

br.yi=mijy/2;

/\* folosesc aceasta structura pentru a creea coodonatele primului varf,

aflat pe diagonala principala, al primului buton, al doilea varf fiind

in functie de lungimea si latimea butonului

creez doar primul buton in asa fel incat toate celelalte butoane sa depinda depozitia

si dimensiule primului buton, pentru a fi mai usor de editat

\*/

/// BUTON ROMANA

char textr[256]= {"ROMANA"};

/\*sirul "textr" va memora textul "ROMANA" \*/

setfillstyle(SOLID\_FILL,YELLOW);

bar(br.xi,br.yi,br.xi+lungime,br.yi+latime);

/\* se stabileste tipul de culoare a polinomului ce va reprezenta butonul

se constuieste un dreptunghi de la primul varf pana la ultimul varf de pe diagonala principala \*/

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

/\* comanda prin care i se atribuyie textului un anumit stil, font ingrosat,

desfasurare orizontala, dimensinea caracterelor e de 5 ori mai mare decat ar fi in mod normal \*/

setcolor(BLUE);

/\*cu aceasta comanda voi seta culoarea pe care o va avea textul, in acest caz albastru (BLUE) \*/

setbkcolor(YELLOW);

/\*cu aceasta comanda voi seta culoarea backgroundului pe care o va avea textul, in acest caz galben (YELLOW)

pentru a evita spatiul negru din jurul textlui \*/

outtextxy(br.xi+lungime/2-textwidth(textr)/2, br.yi+latime/2-textheight(textr)/2,textr);

/\*cu aceasta comanda voi afisa textul incepand de la un punct fix

motivul pentru care incep afisarea textului imediat ce l-am initializat,

este pentru a creea un suport pentru celelate texte

astfel, daca nu afisam textul imediat ce l-am creat,

textele de toate butoanele s-ar fi situat in partea dreapta a butoanelor,deci nu ar fi centrate

\*/

/// BUTON ENGLEZA

char texte[256]= {"ENGLEZA"};

setfillstyle(SOLID\_FILL,YELLOW);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(BLUE);

bar(br.xi,br.yi+3\*latime,br.xi+lungime,br.yi+4\*latime);

setbkcolor(YELLOW);

outtextxy(br.xi+lungime/2-textwidth(texte)/2, br.yi+7\*latime/2-textheight(texte)/2,texte);

/\* se aplica acelasi principiu ca si in cazul BUTTON 1, QUIT BUTTON fiind o derivata a acestuia\*/

POINT mouse;

/\* mouse va memora pozitia cursorului mouse-ului la un anumit moment \*/

int page=0;

/\* page va memora pagina pecare se va desena \*/

int aparitii[5]= {0};

/\* vectorul aparitii va fi folosit drept un contor care va asigura faptul ca imaginile nu se vor afisa de mai multe ori \*/

while(cabral==0)

{/\* cat timp utilizatorul nu a selectat vreun buton, se genereaza in continuu se deseneaza o pagina \*/

setactivepage(page);

/\* seteaza pagina pe care se va desena \*/

setvisualpage(1-page);

/\* seteaza ca pagina pe care o va vizualiza observatorul sa fie cea precedenta \*/

cleardevice();

/\* sterge tot ce este pe pagina activa \*/

GetCursorPos(&mouse);

/\* aceasta comanda reda pozitia la care se afla cursorul mouse-ului in acel moment \*/

readimagefile("bk.jpg",0,0,lungime\_ecran,latime\_ecran);

/\* backgroundul programului va fi o imat=gine cu dimensiunea de 1x1 pixeli

pentru a face citirea si afisarea mai rapida,

astfel animatia va derula mai usor si mai eficient \*/

///ROMANA

setfillstyle(SOLID\_FILL,YELLOW);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(BLUE);

bar(br.xi,br.yi,br.xi+lungime,br.yi+latime);

setbkcolor(YELLOW);

outtextxy(br.xi+lungime/2-textwidth(textr)/2, br.yi+latime/2-textheight(textr)/2,textr);

///ENGLEZA

setfillstyle(SOLID\_FILL,YELLOW);

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(BLUE);

bar(br.xi,br.yi+3\*latime,br.xi+lungime,br.yi+4\*latime);

setbkcolor(YELLOW);

outtextxy(br.xi+lungime/2-textwidth(texte)/2, br.yi+7\*latime/2-textheight(texte)/2,texte);

/\*acelasi pricipiu ca si la initializare \*/

if(GetAsyncKeyState(VK\_LBUTTON))

{/\* daca s-a apasat cumva click, verifica urmatoarele 2 lucruri \*/

if(mouse.x>=br.xi&&mouse.x<=br.xi+lungime&&mouse.y>=br.yi&&mouse.y<=br.yi+latime)

{ /\* daca cursorul se afla in interiorul butonului de ROMANA,

daca da atunci optiunea utilizatorului (cabral) devine 1 (primul buton) \*/

cabral=1;

}

if(mouse.x>=br.xi&&mouse.x<=br.xi+lungime&&mouse.y>=br.yi+3\*latime&&mouse.y<=br.yi+4\*latime)

{/\* daca cursorul se afla in interiorul butonului de ENGLEZA,

daca da atunci optiunea utilizatorului (cabral) devine 2 (al doilea buton) \*/

cabral=2;

}

}

page=1-page;

/\* page va avea doar 2 valori, astfel se vor folosi doar 2 pagini, una anterioada care va fi afisata utilizatorului,

iar a doua pagina va fi stearga si redesenata

tranzitia dintre cele 2 pagini va elimita efectul de glich pecare l-ar fi avut animatia \*/

delay(10);

/\* aceasta comanda ingheata programul pentru 10 milisecunde\*/

}

closegraph();

/\* se inchide toate paginile deschise, astfel sa nu fie deschise

prea multe pagini, deci ca sa nu scada performanta jocului\*/

if(cabral==1)

{/\* daca utilizatorul a ales limba romana\*/

int gata=0;

/\* este un contor care va verifica daca utilizatorul a terminat de citit toate instructiunile\*/

char textc[256]= {"APASA SAGEATA DREAPTA PENTRU A TRECE LA PAGINA URMATOARE"};

/\*sirul de caractere va memora un text care va fi afisat pe tot parcursul animatiei,

cu scopul de a indruma si ajuta utilizatorul \*/

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

while(gata==0)

{/\* cat timp utilizatorulnu a terminat de citit\*/

settextstyle(1,HORIZ\_DIR,2);

setcolor(WHITE);

outtextxy(mijx-textwidth(textc)/2,latime\_ecran-50-textheight(textc)/2,textc);

/\* se afiseaza textul ajutator\*/

if(aparitii[0]==0)

readimagefile("regula 1 ro-gigapixel.jpg",0,0,lungime\_ecran,latime\_ecran);

/\*daca utilizatorul nu a apasat tasta sageata "dreapta", se va afisa prima imagine cu instructiuni\*/

if(GetAsyncKeyState(VK\_RIGHT))

{/\* daca a fost apasata tasta sageata "dreapta", inseamna ca se doreste trecerea la alta pagina\*/

if(aparitii[0]==0)

{

aparitii[0]=1;

getch();

}

/\* prin acest if se asigura faptul ca prima imagine nu va fi asfisata incontinuu\*/

if(aparitii[1]==0)

{/\*daca nu a fost afisata prima poza, se afiseaza\*/

getch();

cleardevice();

/\* se curata pagina precendenta pentru a fi redata o noua imagine\*/

readimagefile("regula 2 ro-gigapixel.jpg",0,0,lungime\_ecran,latime\_ecran);

/\* se afiseaza imaginea pe tot ecranul\*/

outtextxy(mijx-textwidth(textc)/2,latime\_ecran-50-textheight(textc)/2,textc);

/\* se afiseaza textul ajutator\*/

aparitii[1]=1;

/\* aparitii de poza respectiva se face 1 pentru a nu semaiafisa o data\*/

getch();

/\* se foloseste aceasta comanda pentru a da posibilitatea utilizatorului

de a citi cat timp are nevoie\*/

}

else if(aparitii[2]==0)

{

getch();

cleardevice();

readimagefile("regula 3 ro-gigapixel.jpg",0,0,lungime\_ecran,latime\_ecran);

outtextxy(mijx-textwidth(textc)/2,latime\_ecran-50-textheight(textc)/2,textc);

aparitii[2]=1;

getch();

}

else if(aparitii[3]==0)

{

getch();

cleardevice();

readimagefile("regula 4 ro-gigapixel.jpg",0,0,lungime\_ecran,latime\_ecran);

outtextxy(mijx-textwidth(textc)/2,latime\_ecran-50-textheight(textc)/2,textc);

aparitii[3]=1;

getch();

}

else if(aparitii[4]==0)

{

getch();

cleardevice();

readimagefile("regula 5 ro-gigapixel.jpg",0,0,lungime\_ecran,latime\_ecran);

outtextxy(mijx-textwidth(textc)/2,latime\_ecran-50-textheight(textc)/2,textc);

aparitii[4]=1;

gata=1;

getch();

}

/\* acelasi principiu este la toate if-urile\*/

}

}

}

else if (cabral==2)

{/\*daca utilizatorul a selectat ENGLEZA

se aplica acelasi mecanism ca in cazul sectiunii de ROMANA\*/

int gata=0;

char textc[256]= {"PRESS THE RIGHT ARROW TO GO TO THE NEXT PAGE"};

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

while(gata==0)

{

settextstyle(1,HORIZ\_DIR,2);

setcolor(WHITE);

outtextxy(mijx-textwidth(textc)/2,latime\_ecran-25-textheight(textc)/2,textc);

if(aparitii[0]==0)

readimagefile("regula 1 eng-gigapixel.jpg",0,0,lungime\_ecran,latime\_ecran);

if(GetAsyncKeyState(VK\_RIGHT))

{

if(aparitii[0]==0)

{

aparitii[0]=1;

getch();

}

if(aparitii[1]==0)

{

getch();

cleardevice();

readimagefile("regula 2 eng-gigapixel.jpg",0,0,lungime\_ecran,latime\_ecran);

outtextxy(mijx-textwidth(textc)/2,latime\_ecran-25-textheight(textc)/2,textc);

aparitii[1]=1;

getch();

}

else if(aparitii[2]==0)

{

getch();

cleardevice();

readimagefile("regula 3 eng-gigapixel.jpg",0,0,lungime\_ecran,latime\_ecran);

outtextxy(mijx-textwidth(textc)/2,latime\_ecran-25-textheight(textc)/2,textc);

aparitii[2]=1;

getch();

}

else if(aparitii[3]==0)

{

getch();

cleardevice();

readimagefile("regula 4 eng-gigapixel.jpg",0,0,lungime\_ecran,latime\_ecran);

outtextxy(mijx-textwidth(textc)/2,latime\_ecran-25-textheight(textc)/2,textc);

aparitii[3]=1;

getch();

}

else if(aparitii[4]==0)

{

getch();

cleardevice();

readimagefile("regula 5 eng-gigapixel.jpg",0,0,lungime\_ecran,latime\_ecran);

outtextxy(mijx-textwidth(textc)/2,latime\_ecran-25-textheight(textc)/2,textc);

aparitii[4]=1;

gata=1;

getch();

}

}

}

}

/\*motivul pentru care am ales sa afisez poze in loc de a scrie regulile efectiv

a fost acela de a evita aparitia altor erori si bug-uri,

sau de a scadea performanta si stilul programului\*/

closegraph();

return 0;

}

**STAGE 1**

#include <bits/stdc++.h>

#include <graphics.h>

#include <windows.h>

#include <ctime>

#include <stdio.h>

#include <MMSystem.h>

#include "stagiu21.cpp"

using namespace std;

int stagiu1()

{

///INITIALIZARE PAGINA GRAFICA FULL SCREEEN

DWORD lungime\_ecran =GetSystemMetrics(SM\_CXSCREEN);

DWORD latime\_ecran =GetSystemMetrics(SM\_CYSCREEN);

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

///COORDONATELE CENTRULUI PAGINII GRAFICE

int mijx=lungime\_ecran/2;

int mijy=latime\_ecran/2;

///LEVEL 1

///POZITIA PUNCTULUI DE SUS

struct PozitiePlayerPunct1

{

int x;

int y;

} pozplayer1;

pozplayer1.x=mijx;

pozplayer1.y=latime\_ecran-90;

///POZITIA PUNCTULUI DE JOS STANGA

struct PozitiePlayerPunct2

{

int x;

int y;

} pozplayer2;

pozplayer2.x=pozplayer1.x-20;

pozplayer2.y=pozplayer1.y+30;

///POZITIA PUNCTULUI DE JOS DREAPTA

struct PozitiePlayerPunct3

{

int x;

int y;

} pozplayer3;

pozplayer3.x=pozplayer1.x+20;

pozplayer3.y=pozplayer1.y+30;

char kill[50];

/\* un text ce memoreaza numarul de inamici distrusi \*/

char life[50];

char shiel[50];

/\* un text care memoreaza intervalul de timp in care se incarca scutul sau tunul\*/

int page=0;

/\* page va memora pagina pecare se va desena \*/

int viteza=10;

/\* este viteza cucare se va deplasa avatarul pe ecran\*/

unsigned long long current\_time;

/\* aceasta variabila va memora timpul curent din animatie,

fiind folosita la ghidarea comenzilor jucatorului\*/

unsigned long long raw\_time=time(NULL);

/\* aceasta variabila va memora timpul curent de dinainte de animatie,

cu scopul de a ajuta la organizarea coloanelor de gloante\*/

unsigned long long time\_shield=0;

/\* este timpul in care actioneaza scutul\*/

unsigned long long timer\_shield=0;

/\* este timpul de incarcare a scutului\*/

unsigned long long timer\_gun=0;

/\*este timpul necesar de incarcare a tunului\*/

unsigned long long time\_shoot=0;

/\* este timpul de actiune al tunului\*/

unsigned long long auxi,auxi\_b=2;

/\* doua variabile care vor ajuta la crearea fazelor tunului \*/

unsigned long long last\_wave;

/\* o variabila folosita la initializarea inamicilor si a gloantelor\*/

unsigned long long time\_add;

/\* timpul de afisare al unui anunt\*/

unsigned long long time\_add1;

/\* timpul de afisare al unui anunt\*/

int shootx,shooty,shootfinx;

/\* coordonatele de inceput ale razeitunului\*/

int raza\_enemy=25;

/\* dimensiunea inamicului\*/

int viteza\_enemy[21]= {};

int viteza\_enem=10;

/\*viteza de deplasare a inamicilor\*/

int dir\_enemy[21]= {};

/\*directia pentru fiecare inimic, toate incep cu 0, 1 spre dreapta, 0 spre stanga\*/

int pozenemyx[21];

/\* pozitia pe orizontala a inamicului\*/

int pozenemyy[21]= {};

/\* pozitia pe verticala a inamicului\*/

int life\_enemy[21]= {1};

/\* viata fiecarui inamic, 1- este in viata, deci poate fi desenat, 0- este mort, deci nu poate fi desenat\*/

int raza\_shield=50;

/\*raza scutului jucatorului\*/

long long life\_bull[101]= {};

/\* viata fiecarui glont\*/

int nr\_enemy=10;

/\* numarul maxim de inamici de pe ecran\*/

int bullet;

/\* o variabila ce reprezinta glontul\*/

int speed\_bul=10;

/\* viteza de deplasare a glontului\*/

int fire;

int bull\_st[101]= {};

/\* un vector ce contine un registru al gloantelor, 1 este in desfasurare, 0 a ajuns in partea de jos a ecranului\*/

int bull\_stp[101]= {1};

/\* un vector ce contine informatii cu privire la gloantele care au lovit jucatorul\*/

int contor\_p=0;

/\* un contor ce verifica daca un glont a lovit avatarul jucatorului\*/

int total\_bull=0;

/\*numarul total de gloante dintr-un val\*/

int contor\_bul=0;

/\*variabila prin care se ia fiecare glont in parte\*/

int cnt;

/\* este un contor\*/

int hit=0;

/\* o variabila ce arata daca avataatul este lovit de gloante sau nu\*/

int life\_player=3;

/\* vietile jucatorului\*/

int gun\_fire=0;

/\* variabila ne arata daca tunul este activ(1) sau nu(0)\*/

int shield=0;

/\* variabila ne arata daca scutul este activ(1) sau nu(0) \*/

int nr\_kill=0;

/\* reprezinta numarul de inamici pe care jucatorul i-a distrus in acel moment\*/

int limit\_kill=60;

/\* numarul maxim de inamici care pot fi distrusi\*/

int contor\_kill=0;

/\* numara cati inamici au fost distrusi intr-un val\*/

int enemyr[50]= {};

/\* un vector ce memoreaza numarul de valuri alre inamicilor\*/

int r\_limit=0;

/\* numarul de inamici ce trebuie distrusi pentru a declansa un val\*/

int car=0;

/\* aceasta variabila salveaza rezultatul jocului, astfel, daca este 1

atunci inseamna ca jucatorul a terminat cu succes stadiul, daca este 2

inseamna ca jucatorul a pierdut stadiul\*/

struct fire\_poof

{

int x;

int y;

} bullets[101];

/\* coordonatele de inceput ale gloantelor\*/

int nr\_bul=50;

/\* numarul maxim de gloante dintr-un val\*/

last\_wave=raw\_time;

/// APARITII BULLET

total\_bull=rand()%60;

/\* numarul total de gloante dintru val va fi generat la intamplare\*/

for(int bullet=1; bullet<=total\_bull; bullet++)

{

bullets[bullet].x=rand()%lungime\_ecran;

bullets[bullet].y=50;

life\_bull[bullet]=raw\_time+rand()%7;

bull\_st[bullet]=1;

bull\_stp[bullet]=1;

/\* se ia fiecare glont in parte si se alege la intamplare

coordonata de orizontala, apoi se stabileste un interval de timp

de la inceperea valului la care vor aparea gloantele (life\_bull)

totodata li se atribuieste 2 statuturi, primul este acela ca pot aparea pe

ecran si al doilea reprezinta faptul ca nu a lovit pana acm jucatorul

\*/

}

total\_bull=rand()%60;

/\* dupa se genereaza numarul total de gloante pentru urmatorul val\*/

///APARITII ENEMY

for(int enemy=1; enemy<=10; enemy++)

{

pozenemyx[enemy]=rand()%lungime\_ecran+1;

pozenemyy[enemy]=50+rand()%100+1;

viteza\_enemy[enemy]=10;

life\_enemy[enemy]=1;

if(pozenemyx[enemy]>=mijx&&pozenemyx[enemy]<=lungime\_ecran)

dir\_enemy[enemy]=1;

else if(pozenemyx[enemy]<mijx&&pozenemyx[enemy]>=raza\_enemy)

dir\_enemy[enemy]=0;

for(int i=raza\_enemy; i>=15; i--)

circle(pozenemyx[enemy],pozenemyy[enemy],raza\_enemy);

/\* se ia fiecare inamic si i se genereaza o coordonata orizontala la intamplare

si o coordonata verticala intr-un anumit interval, tot la intamplare

adica se organizeaza gloantele intr-un fel de matrice, in care numarul de linii

este deta de timpul la cate apar acestea, iar numarul de colone reprezinta fix

lungimea ecranului

dupa se verifica pozitia relativa acestora fata de mijlocul ecranului,

daca se afla in partea stanga a acelui punct, inamicul se va deplasa

catre stanga si viceversa

apoi se deseneaza inamicul efectiv, adica un cerc de o anumita grosime

\*/

}

/\* cat timp jucatorul are vieti si cat timp inca sunt inamici pe ecran\*/

while(nr\_kill<limit\_kill&&life\_player>0)

{

setactivepage(page);

/\* seteaza pagina pe care se va desena \*/

setvisualpage(1-page);

/\* seteaza ca pagina pe care o va vizualiza observatorul sa fie cea precedenta \*/

cleardevice();

/\* sterge tot ce este pe pagina activa \*/

setfillstyle(SOLID\_FILL,LIGHTBLUE);

floodfill(1,1,LIGHTBLUE);

/\* se deseneaza efectiv fundalul, fara citirea unei imagini\*/

int pozplayer[8]= {pozplayer1.x,pozplayer1.y,pozplayer2.x,pozplayer2.y,pozplayer3.x,pozplayer3.y,pozplayer1.x,pozplayer1.y};

setcolor(YELLOW);

drawpoly(4,pozplayer);

setfillstyle(SOLID\_FILL,YELLOW);

fillpoly(4,pozplayer);

/\* se deseneaza un polinom cu 3 varfuri dupa care este colorat, acest polinom reprezentand jucatorul\*/

current\_time=time(NULL);

/\* se salveaza timpul din joc\*/

///COMANDA MERS DREAPTA

if(GetAsyncKeyState(VK\_RIGHT)&&pozplayer3.x+viteza<=lungime\_ecran)

{

pozplayer1.x+=viteza;

pozplayer2.x+=viteza;

pozplayer3.x+=viteza;

}

/\*daca jucatorul a apasat tasta sageata dreapta si jucatorul nu a ajuns la extremitate

a din dreapta a ecranului, avatarul se va deplasa catre dreapta\*/

///COMANDA MERS STANGA

if(GetAsyncKeyState(VK\_LEFT)&&pozplayer2.x+viteza>=1)

{

pozplayer1.x-=viteza;

pozplayer2.x-=viteza;

pozplayer3.x-=viteza;

}

/\*daca jucatorul a apasat tasta sageata stanga si jucatorul nu a ajuns la extremitate

a din stanga a ecranului, avatarul se va deplasa catre stanga\*/

int ok=0;

///COMANDA SHIELD

if(GetAsyncKeyState(VK\_DOWN))

{/\* daca esta apasata tasta sageata jos\*/

if(timer\_shield<=current\_time&&gun\_fire==0)

{

shield=1;

for(int i=0; i<=5; i++)

arc(pozplayer1.x,pozplayer1.y,0,180,raza\_shield+i);

time\_shield=current\_time+3;

/\* se verifica daca timpul necesar de incarcare a scutului este mai mic decat timpul curent al jocului

in caz afirmativ se stabileste valoarea 1 a variabilei shield, care va arata programului

ca scutul a fost actival, si se va genera scutul

timpul de actiune al scutului este de 3 secunde \*/

}

else if(timer\_shield>current\_time)

{

setcolor(YELLOW);

outtextxy(mijx,mijy,"THE SHIELD IS LOADING...");

/\* daca timpul de incarcare este mai mare decat timpul curent, atunci se va afisa un mesaj corespunzator catre jucator\*/

}

else if(gun\_fire==1)

{

setcolor(YELLOW);

outtextxy(mijx,mijy,"CANNOT USE THE SHIELD WHILE THE CANNON IS ACTIVE !");

/\* daca tunul este activ, pe ecran va aparea un mesaj in care i se va

explica jucatorului ca nu poate folosi doua comenzi in acelasi timp\*/

}

}

if(time\_shield>=current\_time)

{

setcolor(WHITE);

for(int i=0; i<=5; i++)

arc(pozplayer1.x,pozplayer1.y,0,180,raza\_shield+i);

if(time\_shield==current\_time)

{

timer\_shield=current\_time+5;

shield=0;

}

/\* daca timpul de actiune este mai mare decat timpul curent al jocului, atunci se deseneaza scutul

daca cele doua variabile sunt egale, atunci scutul se va dezactiva\*/

}

if(timer\_shield>current\_time)

{

setcolor(WHITE);

sprintf(shiel,"THE SHIELD IS READY IN : %d",timer\_shield-current\_time);

outtextxy(10,mijy+50,shiel);

/\* in cazul in care timpul de incarcare este mai mare decat timpul curent,

atunci se va afisa pe ecran timpul ramas pana la incarcarea completa \*/

}

else if(time\_shield<=current\_time&&shield==0)

{

setcolor(WHITE);

outtextxy(10,mijy+50,"THE SHIELD IS READY!");

/\*in cazul in care scutul este incarcat si inca nu s-a activat,

atunci se va afisa un mesaj in care i se atentioneaza jucatorului ca scutul este gata\*/

}

///COMANDA TUN

if(GetAsyncKeyState(VK\_SPACE))

/\*daca s-a apasat tasta space\*/

if(timer\_gun<=current\_time&&time\_shoot<current\_time&&shield==0)

{

time\_shoot=current\_time+5;

auxi=5;

shootx=pozplayer1.x-3;

shooty=pozplayer1.y;

shootfinx=pozplayer1.x+3;

gun\_fire=1;

/\* daca timpul de incarcare este mai mic decat timpul curent si daca tunul

nu actioneaza si daca scutulnu este activ, atunci se activeaza tunul

o variabila auxi=5, asta inseamna ca in faza a 5a tunul va crea pagube inamicilor aflati in raza

raza tunului va incepe din vf de sus al avatarului se se va extinde pe toata circumferinta superioara a acestuia\*/

}

else if(timer\_gun>current\_time)

{

setcolor(YELLOW);

outtextxy(mijx,mijy,"THE CANNON IS LOADING...");

/\* daca timpul de incarcare este mai mare decat timpul curent, atunci se va afisa un mesaj corespunzator catre jucator\*/

}

else if(shield==1)

{

setcolor(YELLOW);

outtextxy(mijx,mijy,"CANNOT USE THE CANNON WHILE THE SHIELD IS ACTIVE !");

/\* daca scutul este activ, pe ecran va aparea un mesaj in care i se va

explica jucatorului ca nu poate folosi doua comenzi in acelasi timp\*/

}

if(time\_shoot>=current\_time)

if(time\_shoot-current\_time!=auxi)

{

bar(shootx-=10,shooty,shootfinx+=10,1);

auxi=time\_shoot-current\_time;

/\*daca tunul este activat si daca s-a terminat una dintre fazele tunului,

se creeaza raza conform explicatiilor de mai sus

auxi va memora diferenta intre timpul de actiune si timpul curent, pentru a delimita cele auxi faze \*/

}

else

{

setfillstyle(SOLID\_FILL,auxi);

bar(shootx,shooty,shootfinx,1);

if(time\_shoot==current\_time)

{

timer\_gun=current\_time+10;

gun\_fire=0;

}

/\* se creeaza faza tunului, daca cele 2 variabile sunt egale, tunul se dezactiveaza \*/

}

if(timer\_gun>current\_time)

{

setcolor(WHITE);

sprintf(shiel,"THE CANNON IS READY IN : %d",timer\_gun-current\_time);

outtextxy(10,mijy+75,shiel);

/\* in cazul in care timpul de incarcare este mai mare decat timpul curent,

atunci se va afisa pe ecran timpul ramas pana la incarcarea completa \*/

}

else if(timer\_gun<=current\_time&&gun\_fire==0)

{

setcolor(WHITE);

outtextxy(10,mijy+75,"THE CANNON IS READY!");

/\*in cazul in care tunul este incarcat si inca nu s-a activat,

atunci se va afisa un mesaj in care i se atentioneaza jucatorului ca tunul este gata\*/

}

/// CONTORALE ENEMY

for(int enemy=1; enemy<=10; enemy++)

{

if(gun\_fire==1&&pozenemyx[enemy]>=(shootfinx-70)-raza\_enemy&&pozenemyx[enemy]<=shootfinx+raza\_enemy&&auxi==1)

life\_enemy[enemy]=0;

/\*daca inamicul se afla in raza tunului si se afla in faza a cincea, atunci inamicul este distrus\*/

if(life\_enemy[enemy]==1)

{ /\*daca inamicul este in viata\*/

if(pozenemyx[enemy]>=lungime\_ecran-raza\_enemy-viteza\_enem)

{

dir\_enemy[enemy]=0;

pozenemyx[enemy]-=viteza\_enem;

}

/\* daca inamicul este la capatul din dreapta al ecranului atunci se schimba directia\*/

else if(pozenemyx[enemy]<=raza\_enemy+viteza\_enem)

{

dir\_enemy[enemy]=1;

pozenemyx[enemy]+=viteza\_enem;

}

/\* daca inamicul este la capatul din stanga al ecranului atunci se schimba directia\*/

else if(dir\_enemy[enemy]==1)

pozenemyx[enemy]+=viteza\_enem;

/\*inamicul se deplaseaza catre dreapta\*/

else if(dir\_enemy[enemy]==0)

pozenemyx[enemy]-=viteza\_enem;

/\*inamicul se deplaseaza catre stanga\*/

setcolor(MAGENTA);

for(int i=raza\_enemy; i>=15; i--)

circle(pozenemyx[enemy],pozenemyy[enemy],i);

/\* se deseneaza inamicul\*/

}

}

///ENEMY REINFORCEMENTS

contor\_kill=0;

for(cnt=1; cnt<=10; cnt++)

if(life\_enemy[cnt]==0)

contor\_kill++;

/\* se verifica daca unul dintre cei 10 inamici estedoborat\*/

if(contor\_kill+r\_limit>nr\_kill)

nr\_kill=r\_limit+contor\_kill;

{

setcolor(GREEN);

sprintf(kill,"THE NUMBER OF KILLS IS : %d",nr\_kill);

}

/\* daca numarul de inamici doborati adunat cu cel existent este mai mare

decat numarul afisat de ecran, se modifica numarul de inamici distrusi\*/

outtextxy(25,mijy,kill);

if(nr\_kill%5==0&&enemyr[nr\_kill/5]==0)

{

time\_add=current\_time+2;

enemyr[nr\_kill/5]=1;

r\_limit=nr\_kill;

for(cnt=1; cnt<=10; cnt++)

if(life\_enemy[cnt]==0)

{

pozenemyx[cnt]=rand()%lungime\_ecran+1;

pozenemyy[cnt]=50+rand()%100+1;

viteza\_enemy[cnt]=10;

life\_enemy[cnt]=1;

if(pozenemyx[cnt]>=mijx&&pozenemyx[cnt]<=lungime\_ecran)

dir\_enemy[cnt]=1;

else if(pozenemyx[cnt]<mijx&&pozenemyx[cnt]>=raza\_enemy)

dir\_enemy[cnt]=0;

for(int i=raza\_enemy; i>=15; i--)

circle(pozenemyx[cnt],pozenemyy[cnt],raza\_enemy);

}

/\* daca un anumit numar de inamici au fost distrusi se genereaza alti inamici\*/

}

if(nr\_kill>=10&&time\_add>=current\_time)

{

setcolor(RED);

outtextxy(mijx,700,"WARNING! THE ENEMY GATHERED MORE REINFORCEMENTS!");

/\* jucatorul este anuntat ca inamicii au primit intariri\*/

}

else if(time\_add>=current\_time&&nr\_kill<10&&nr\_kill>=5)

{

setcolor(RED);

outtextxy(mijx,700,"WARNING! THE ENEMY GATHERED REINFORCEMENTS!");

/\* jucatorul este anuntat ca inamicii au primit intariri\*/

}

/// FIRE CONTROL

for(bullet=1; bullet<=total\_bull; bullet++)

{

if(life\_bull[bullet]<=current\_time&&bull\_st[bullet]==1)

{

bullets[bullet].y+=speed\_bul;

setfillstyle(SOLID\_FILL,RED);

bar(bullets[bullet].x,bullets[bullet].y,bullets[bullet].x+10,bullets[bullet].y+25);

}

/\* se ia fiecare glont si se verifica daca a pornit si daca nu a ajuns la baza ecranului, in caz afirmativ se deseneaza\*/

if(bullets[bullet].y>=latime\_ecran)

{

bull\_st[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_st[cnt]==1)

contor\_bul++;

if(contor\_bul==0)

total\_bull=0;

/\* daca un glont a ajuns la baza ecranului, se sterge de pe ecran \*/

}

if(shield==1&&bullets[bullet].x>=pozplayer1.x-raza\_shield&&bullets[bullet].x<=pozplayer1.x+raza\_shield&&bullets[bullet].y>=pozplayer1.y-raza\_shield)

{

bull\_st[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_st[cnt]==1)

contor\_bul++;

if(contor\_bul==0)

total\_bull=0;

/\*daca scutul este activ si un glont a atins jucatorul, lovitura este anulat si glontul este sters\*/

}

else if(shield==0&&pozplayer1.x>=bullets[bullet].x&&pozplayer1.x<=bullets[bullet].x+10&&pozplayer1.y>=bullets[bullet].y&&pozplayer1.y<=bullets[bullet].y+25&&bull\_st[bullet]==1&&bull\_stp[bullet]==1)

{

hit=1;

bull\_st[bullet]=0;

bull\_stp[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_st[cnt]==1)

contor\_bul++;

if(contor\_bul==0)

total\_bull=0;

/\* daca glontul loveste varful de sus al avatarului, se salveaza glontul care l-a lovit si se sterge glontul\*/

}

else if(shield==0&&pozplayer2.x>=bullets[bullet].x&&pozplayer2.x<=bullets[bullet].x+10&&pozplayer2.y>=bullets[bullet].y&&pozplayer2.y<=bullets[bullet].y+25&&bull\_st[bullet]==1&&bull\_stp[bullet]==1)

{

hit=1;

bull\_st[bullet]=0;

bull\_stp[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_st[cnt]==1)

contor\_bul++;

if(contor\_bul==0)

total\_bull=0;

/\* daca glontul loveste varful de din drepta al avatarului, se salveaza glontul care l-a lovit si se sterge glontul\*/

}

else if(shield==0&&pozplayer3.x>=bullets[bullet].x&&pozplayer3.x<=bullets[bullet].x+10&&pozplayer3.y>=bullets[bullet].y&&pozplayer3.y<=bullets[bullet].y+25&&bull\_st[bullet]==1&&bull\_stp[bullet]==1)

{

hit=1;

bull\_st[bullet]=0;

bull\_stp[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_st[cnt]==1)

contor\_bul++;

if(contor\_bul==0)

total\_bull=0;

/\* daca glontul loveste varful de stanga al avatarului, se salveaza glontul care l-a lovit si se sterge glontul\*/

}

}

///BULLETS RESET

if(total\_bull==0)

{

total\_bull=rand()%60;

for(bullet=1; bullet<=total\_bull; bullet++)

{

bullets[bullet].x=rand()%lungime\_ecran;

bullets[bullet].y=50;

life\_bull[bullet]=current\_time+rand()%7;

bull\_st[bullet]=1;

bull\_stp[bullet]=1;

}

/\*se genereaza noi gloante, daca toate gloantele au ajuns jos\*/

}

///PLAYER'S LIFES

for(bullet=1; bullet<=total\_bull; bullet++)

if ( bull\_stp[bullet]==0)

{

contor\_p=1;

bull\_stp[bullet]=1;

}

/\* se verifica daca un glont a lovit jucatorul\*/

if(contor\_p==1)

{

life\_player--;

contor\_p=0;

}

/\* daca un glont a lovit jucatorul, viata jucatorului scade

eu numar gloantele care au lovit avatarul, nu numarul de lovituri ale acestora,

deoarece viata jucatorului s-ar reduce la 0 dupa primul glont\*/

setcolor(YELLOW);

for(cnt=1; cnt<=life\_player; cnt++)

for(int i=1; i<=20; i++)

circle(20+cnt\*50,latime\_ecran-20,i);

/\* se afiseaza viata jucatorului\*/

delay(5);

/\* aceasta comanda ingheata programul pentru 5 milisecunde\*/

page=1-page;

/\* page va avea doar 2 valori, astfel se vor folosi doar 2 pagini, una anterioada care va fi afisata utilizatorului,

iar a doua pagina va fi stearga si redesenata

tranzitia dintre cele 2 pagini va elimita efectul de glich pecare l-ar fi avut animatia \*/

}

if (life\_player<=0)

{/\* daca jucatorul a ramas fara vieti atunci se afiseaza un mesaj prin care i se instiinteaza acest lucru, si urmatoarele instructiuni\*/

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

char textulet[256]="PRESS SPACE TO RESTART THE STAGE";

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(mijx-textwidth("GAME OVER!")/2,mijy-textheight("GAME OVER!")/2,"GAME OVER!");

settextstyle(BOLD\_FONT,HORIZ\_DIR,2);

setcolor(WHITE);

outtextxy(mijx-textwidth(textulet)/2,3\*mijy/2-textheight(textulet)/2,textulet);

delay(2000);

getch();

car=2;

}

else

{ /\* daca jucatorul a terminat stagiul si mai are vieti atunci se afiseaza un mesaj prin care i se instiinteaza acest lucru,

si urmatoarele instructiuni\*/

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

char textulet[256]="PRESS SPACE TO CONTINUE TO THE NEXT STAGE";

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(mijx-textwidth("VICTORY!")/2,mijy-textheight("VICTORY!")/2,"VICTORY!");

settextstyle(BOLD\_FONT,HORIZ\_DIR,2);

setcolor(WHITE);

outtextxy(mijx-textwidth(textulet)/2,3\*mijy/2-textheight(textulet)/2,textulet);

delay(2000);

getch();

car=1;

}

closegraph();

/\* se inchide toate paginile deschise, astfel sa nu fie deschise

prea multe pagini, deci ca sa nu scada performanta jocului\*/

if(car==2)

stagiu1();

else if(car==1)

stagiu21();

/\*daca car este egal cu 2, adica daca jucatorul a pierdut se reia stagiul

daca car este egal cu 1, adica daca jucatorul a castigat se trece la stagiul urmator\*/

return 0;

}

**STAGE 2 / STAGE 21**

#include <bits/stdc++.h>

#include <graphics.h>

#include <windows.h>

#include <time.h>

#include <stdio.h>

#include <MMSystem.h>

#include "stagiu31.cpp"

using namespace std;

int stagiu2()

{

///INITIALIZARE PAGINA GRAFICA FULL SCREEEN

DWORD lungime\_ecran =GetSystemMetrics(SM\_CXSCREEN);

DWORD latime\_ecran =GetSystemMetrics(SM\_CYSCREEN);

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

///COORDONATELE CENTRULUI PAGINII GRAFICE

int mijx=lungime\_ecran/2;

int mijy=latime\_ecran/2;

///LEVEL 2

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

int viteza=10;

///POZITIA PUNCTULUI DE SUS

struct PozitiePlayerPunct1

{

int x;

int y;

} pozplayer1;

pozplayer1.x=mijx;

pozplayer1.y=latime\_ecran-90;

///POZITIA PUNCTULUI DE JOS STANGA

struct PozitiePlayerPunct2

{

int x;

int y;

} pozplayer2;

pozplayer2.x=pozplayer1.x-20;

pozplayer2.y=pozplayer1.y+30;

///POZITIA PUNCTULUI DE JOS DREAPTA

struct PozitiePlayerPunct3

{

int x;

int y;

} pozplayer3;

pozplayer3.x=pozplayer1.x+20;

pozplayer3.y=pozplayer1.y+30;

char kill[50];

char life[50];

char shiel[50];

int page=0;

unsigned long long current\_time;

unsigned long long raw\_time=time(NULL);

unsigned long long time\_shield=0;

unsigned long long timer\_shield=0;

unsigned long long timer\_gun=0;

unsigned long long time\_shoot=0;

unsigned long long auxi,auxi\_b=2;

unsigned long long last\_wave;

int shootx,shooty,shootfinx;

int raza\_enemy=25;

int viteza\_enemy[21]= {};

int viteza\_enem=10;

int dir\_enemy[21]= {};

int pozenemyx[21];

int pozenemyy[21]= {};

int life\_enemy[21]= {1};

int raza\_shield=50;

long long life\_bull[101]= {};

int nr\_enemy=10;

int bullet;

int speed\_bul=10;

int fire;

int bull\_st[101]= {};

int bull\_stp[101]= {1};

int contor\_p=0;

int total\_bull=0;

int contor\_bul=0;

int cnt;

int hit=0;

int life\_player=5;

int gun\_fire=0;

int shield=0;

int nr\_kill=0;

int limit\_kill=60;

int contor\_kill=0;

int enemyr[50]= {};

unsigned long long time\_add;

unsigned long long time\_add1;

struct fire\_poof

{

int x;

int y;

} bullets[101];

int car=0;

int nr\_bul=50;

/\* toate aceste variabile au fost explicate in stagiul 1\*/

int r\_limit=180;

/\* timpul total de joc\*/

int raza\_bomb=25;

/\* dimensiunea bombelor\*/

int bull\_ste[101]= {};

/\* un vector care salveaza un numar care arata daca avatarul a fost sau nu prin in explozie\*/

int raza\_exp=60;

/\* dimensiunea exploziei\*/

unsigned long long time\_left;

/\* timpul de joc\*/

long long life\_ex[101]= {};

/\* un vector care salveaza statusul exploziilor, 1 dacaeste in cur de detonare, 0 daca a fost detonata deja \*/

char time1[256];

/\* un text care afiseaza timpul ramas\*/

///BOMB SPAWN

total\_bull=rand()%30;

for(int bullet=1; bullet<=total\_bull; bullet++)

{

bullets[bullet].x=rand()%lungime\_ecran;

bullets[bullet].y=50;

life\_bull[bullet]=raw\_time+rand()%7;

bull\_st[bullet]=1;

bull\_stp[bullet]=1;

bull\_ste[bullet]=0;

}

/\* este acelasi principiu ca si la aparitia gloantelor\*/

current\_time=time(NULL);

total\_bull=rand()%30;

time\_left=raw\_time+r\_limit;

/\* cat timp timpul nu s-a scurs si cat timp jucatorul are vieti\*/

while(time\_left>=current\_time&&life\_player>0)

{

setactivepage(page);

setvisualpage(1-page);

cleardevice();

setfillstyle(SOLID\_FILL,LIGHTGREEN);

floodfill(1,1,LIGHTGREEN);

int pozplayer[8]= {pozplayer1.x,pozplayer1.y,pozplayer2.x,pozplayer2.y,pozplayer3.x,pozplayer3.y,pozplayer1.x,pozplayer1.y};

setcolor(YELLOW);

drawpoly(4,pozplayer);

setfillstyle(SOLID\_FILL,YELLOW);

fillpoly(4,pozplayer);

current\_time=time(NULL);

///COMANDA MERS DREAPTA

if(GetAsyncKeyState(VK\_RIGHT)&&pozplayer3.x+viteza<=lungime\_ecran)

{

pozplayer1.x+=viteza;

pozplayer2.x+=viteza;

pozplayer3.x+=viteza;

}

///COMANDA MERS STANGA

if(GetAsyncKeyState(VK\_LEFT)&&pozplayer2.x+viteza>=1)

{

pozplayer1.x-=viteza;

pozplayer2.x-=viteza;

pozplayer3.x-=viteza;

}

///COMANDA SHIELD

if(GetAsyncKeyState(VK\_DOWN))

{

if(timer\_shield<=current\_time&&gun\_fire==0)

{

shield=1;

for(int i=0; i<=5; i++)

arc(pozplayer1.x,pozplayer1.y,0,180,raza\_shield+i);

time\_shield=current\_time+3;

}

else if(timer\_shield>current\_time)

{

setcolor(YELLOW);

//PlaySound(TEXT("DENIED"),NULL,SND\_SYNC);

settextstyle(8,HORIZ\_DIR,1);

outtextxy(mijx,mijy,"THE SHIELD IS LOADING...");

}

else if(gun\_fire==1)

{

setcolor(YELLOW);

// PlaySound(TEXT("DENIED"),NULL,SND\_SYNC);

outtextxy(mijx,mijy,"CANNOT USE THE SHIELD WHILE THE CANNON IS ACTIVE !");

settextstyle(8,HORIZ\_DIR,1);

}

}

if(time\_shield>=current\_time)

{

setcolor(WHITE);

for(int i=0; i<=5; i++)

arc(pozplayer1.x,pozplayer1.y,0,180,raza\_shield+i);

if(time\_shield==current\_time)

{

timer\_shield=current\_time+5;

shield=0;

}

}

if(timer\_shield>current\_time)

{

setcolor(WHITE);

sprintf(shiel,"THE SHIELD IS READY IN : %d",timer\_shield-current\_time);

settextstyle(8,HORIZ\_DIR,1);

outtextxy(10,mijy+50,shiel);

}

else if(time\_shield<=current\_time&&shield==0)

{

setcolor(WHITE);

{

outtextxy(10,mijy+50,"THE SHIELD IS READY!");

settextstyle(8,HORIZ\_DIR,1);

}

}

/\* este esact acelasi principiu ca si la stagiul 1\*/

/// FIRE CONTROL

for(bullet=1; bullet<=total\_bull; bullet++)

{

if(life\_bull[bullet]<=current\_time&&bull\_st[bullet]==1)

{

bullets[bullet].y+=speed\_bul;

setfillstyle(SOLID\_FILL,RED);

setcolor(WHITE);

for(int i=0; i<=3; i++)

circle(bullets[bullet].x,bullets[bullet].y,raza\_bomb-i);

}

if(bullets[bullet].y+raza\_bomb>=latime\_ecran&&bull\_ste[bullet]==0)

{

bull\_st[bullet]=0;

bull\_ste[bullet]=1;

life\_ex[bullet]=current\_time+1;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_st[cnt]==1)

contor\_bul++;

if(contor\_bul==0)

total\_bull=0;

}

/\* este acelasi principiu ca si la stagiul 1 doar ca dupa ce bombele au fost generate, li se atribuie un stadiu de explozie\*/

///EXPLOSION STATE

if(life\_ex[bullet]>=current\_time&&bull\_ste[bullet]==1)

{

/\* da bombele au explodat\*/

if(bullets[bullet].x-raza\_exp<=pozplayer1.x&&bullets[bullet].x+raza\_exp>=pozplayer1.x&&bullets[bullet].y-raza\_exp<=pozplayer1.y&&bullets[bullet].y+raza\_exp>=pozplayer1.y)

{

bull\_stp[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_ste[cnt]==1&&life\_ex[cnt]<current\_time)

contor\_bul++;

if(contor\_bul==total\_bull)

total\_bull=0;

life\_ex[bullet]=current\_time-1;

}

else if(bullets[bullet].x-raza\_exp<=pozplayer2.x&&bullets[bullet].x+raza\_exp>=pozplayer2.x&&bullets[bullet].y-raza\_exp<=pozplayer2.y&&bullets[bullet].y+raza\_exp>=pozplayer2.y)

{

bull\_stp[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_ste[cnt]==1&&life\_ex[cnt]<current\_time)

contor\_bul++;

if(contor\_bul==total\_bull)

total\_bull=0;

life\_ex[bullet]=current\_time-1;

}

else if(bullets[bullet].x-raza\_exp<=pozplayer3.x&&bullets[bullet].x+raza\_exp>=pozplayer3.x&&bullets[bullet].y-raza\_exp<=pozplayer3.y&&bullets[bullet].y+raza\_exp>=pozplayer3.y)

{

bull\_stp[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_ste[cnt]==1&&life\_ex[cnt]<current\_time)

contor\_bul++;

if(contor\_bul==total\_bull)

total\_bull=0;

life\_ex[bullet]=current\_time-1;

}

/\* in aceasta secventa de cod se verifica daca in raza

explozitiei se afla unul dintre punctele avatarului, in caz

afirmativ se salveaza boba care a provocat acea explozie \*/

if(bull\_stp[bullet]==1)

{

setcolor(RED);

for(cnt=raza\_exp; cnt>=raza\_exp-5; cnt--)

circle(bullets[bullet].x,bullets[bullet].y,cnt);

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_ste[cnt]==1&&life\_ex[cnt]<current\_time)

contor\_bul++;

if(contor\_bul==total\_bull)

total\_bull=0;

}

/\* daca timup de desfasurare a exploziei nu s-a sfarsit, se deseneaza explozia\*/

}

if(shield==1&&bullets[bullet].x-raza\_bomb>=pozplayer1.x-raza\_shield&&bullets[bullet].x-raza\_bomb<=pozplayer1.x+raza\_shield&&bullets[bullet].y-raza\_bomb>=pozplayer1.y-raza\_shield)

{

bull\_st[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_ste[cnt]==1&&life\_ex[cnt]<current\_time)

contor\_bul++;

if(contor\_bul==total\_bull)

total\_bull=0;

}

else if(shield==0&&pozplayer1.x>=bullets[bullet].x-raza\_bomb&&pozplayer1.x<=bullets[bullet].x+raza\_bomb&&pozplayer1.y>=bullets[bullet].y-raza\_bomb&&pozplayer1.y<=bullets[bullet].y+raza\_bomb&&bull\_st[bullet]==1&&bull\_stp[bullet]==1)

{

hit=1;

bull\_st[bullet]=0;

bull\_stp[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_ste[cnt]==1&&life\_ex[cnt]<current\_time)

contor\_bul++;

if(contor\_bul==total\_bull)

total\_bull=0;

}

else if(shield==0&&pozplayer2.x>=bullets[bullet].x-raza\_bomb&&pozplayer2.x<=bullets[bullet].x+raza\_bomb&&pozplayer2.y>=bullets[bullet].y-raza\_bomb&&pozplayer2.y<=bullets[bullet].y+raza\_bomb&&bull\_st[bullet]==1&&bull\_stp[bullet]==1)

{

hit=1;

bull\_st[bullet]=0;

bull\_stp[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_ste[cnt]==1&&life\_ex[cnt]<current\_time)

contor\_bul++;

if(contor\_bul==total\_bull)

total\_bull=0;

}

else if(shield==0&&pozplayer3.x>=bullets[bullet].x-raza\_bomb&&pozplayer3.x<=bullets[bullet].x+raza\_bomb&&pozplayer3.y>=bullets[bullet].y-raza\_bomb&&pozplayer3.y<=bullets[bullet].y+raza\_bomb&&bull\_st[bullet]==1&&bull\_stp[bullet]==1)

{

hit=1;

bull\_st[bullet]=0;

bull\_stp[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_ste[cnt]==1&&life\_ex[cnt]<current\_time)

contor\_bul++;

if(contor\_bul==total\_bull)

total\_bull=0;

}

/\* este acelasi principiu ca si in cazul gloantelor de la stagiul 1, in cazul in care avatarul este lovit de o bomba \*/

}

///PLAYER'S LIFES

for(bullet=1; bullet<=total\_bull; bullet++)

if ( bull\_stp[bullet]==0)

{

contor\_p=1;

bull\_stp[bullet]=1;

}

if(contor\_p==1)

{

life\_player--;

contor\_p=0;

}

setcolor(YELLOW);

for(cnt=1; cnt<=life\_player; cnt++)

for(int i=1; i<=20; i++)

circle(20+cnt\*50,latime\_ecran-20,i);

///BULLETS RESET

if(total\_bull==0)

{

total\_bull=rand()%30;

for(bullet=1; bullet<=total\_bull; bullet++)

{

bullets[bullet].x=rand()%lungime\_ecran;

bullets[bullet].y=50;

life\_bull[bullet]=current\_time+rand()%7;

bull\_st[bullet]=1;

bull\_stp[bullet]=1;

bull\_ste[bullet]=0;

}

}

/\* este acelasi principiu ca si in stagiul 1\*/

setcolor(YELLOW);

sprintf(time1,"TIME LEFT : %d SECONDS ",time\_left-current\_time);

settextstyle(8,HORIZ\_DIR,2);

outtextxy(30,400,time1);

/\* se afiseaza timpul ramas\*/

page=1-page;

delay(10);

}

int ok=0;

if (life\_player<=0)

{

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

char textulet[256]="PRESS SPACE TO RESTART THE STAGE";

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(mijx-textwidth("GAME OVER!")/2,mijy-textheight("GAME OVER!")/2,"GAME OVER!");

settextstyle(BOLD\_FONT,HORIZ\_DIR,2);

setcolor(WHITE);

outtextxy(mijx-textwidth(textulet)/2,3\*mijy/2-textheight(textulet)/2,textulet);

delay(2000);

getch();

car=2;

}

else

{

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

char textulet[256]="PRESS SPACE TO CONTINUE TO THE NEXT STAGE";

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(mijx-textwidth("VICTORY!")/2,mijy-textheight("VICTORY!")/2,"VICTORY!");

settextstyle(BOLD\_FONT,HORIZ\_DIR,2);

setcolor(WHITE);

outtextxy(mijx-textwidth(textulet)/2,3\*mijy/2-textheight(textulet)/2,textulet);

delay(2000);

getch();

car=1;

}

/\* este aceeasi mecanica ca si in stagiul 1\*/

closegraph();

if(car==2)

stagiu2(); sau stagiu21();

else if(car==1)

stagiu31(); sau stagiu32();

/\*daca car este egal cu 2, adica daca jucatorul a pierdut se reia stagiul

daca car este egal cu 1, adica daca jucatorul a castigat se trece la stagiul urmator\*/

return 0;

}

**STAGE 3 / STAGE 31 / STAGE 32**

#include <bits/stdc++.h>

#include <graphics.h>

#include <windows.h>

#include <time.h>

#include <stdio.h>

#include <MMSystem.h>

using namespace std;

int stagiu3()

{

int car=0;

///INITIALIZARE PAGINA GRAFICA FULL SCREEEN

DWORD lungime\_ecran =GetSystemMetrics(SM\_CXSCREEN);

DWORD latime\_ecran =GetSystemMetrics(SM\_CYSCREEN);

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

///COORDONATELE CENTRULUI PAGINII GRAFICE

int mijx=lungime\_ecran/2;

int mijy=latime\_ecran/2;

///LEVEL 3

int viteza=10;

///POZITIA PUNCTULUI DE SUS

struct PozitiePlayerPunct1

{

int x;

int y;

} pozplayer1;

pozplayer1.x=mijx;

pozplayer1.y=latime\_ecran-90;

///POZITIA PUNCTULUI DE JOS STANGA

struct PozitiePlayerPunct2

{

int x;

int y;

} pozplayer2;

pozplayer2.x=pozplayer1.x-20;

pozplayer2.y=pozplayer1.y+30;

///POZITIA PUNCTULUI DE JOS DREAPTA

struct PozitiePlayerPunct3

{

int x;

int y;

} pozplayer3;

pozplayer3.x=pozplayer1.x+20;

pozplayer3.y=pozplayer1.y+30;

char kill[50];

char life[50];

char shiel[50];

char time1[256];

int page=0;

unsigned long long current\_time;

unsigned long long raw\_time=time(NULL);

unsigned long long time\_left;

unsigned long long time\_shield=0;

unsigned long long timer\_shield=0;

unsigned long long timer\_gun=0;

unsigned long long time\_shoot=0;

unsigned long long auxi,auxi\_b=2;

unsigned long long last\_wave;

int shootx,shooty,shootfinx;

int viteza\_enemy[21]= {};

int viteza\_enem=10;

int dir\_enemy[21]= {};

int pozenemyx[21];

int pozenemyy[21]= {};

int life\_enemy[21]= {1};

int raza\_shield=50;

long long life\_bull[101]= {};

int nr\_enemy=10;

int bullet;

int speed\_bul=10;

int fire;

int bull\_st[101]= {};

int bull\_stp[101]= {1};

int bull\_ste[101]= {};

int contor\_p=0;

int total\_bull=0;

int contor\_bul=0;

int cnt;

int hit=0;

int life\_player=7;

int gun\_fire=0;

int shield=0;

int nr\_kill=0;

int limit\_kill=60;

int contor\_kill=0;

int enemyr[50]= {};

int r\_limit=180;

unsigned long long time\_add=0;

unsigned long long time\_add1;

struct fire\_poof

{

int x;

int y;

} bullets[101];

int raza\_bomb=25;

int raza\_exp=60;

long long life\_ex[101]= {};

int nr\_bul=50;

///BULLETS SPAWN

total\_bull=rand()%60;

for(int bullet=1; bullet<=total\_bull; bullet++)

{

bullets[bullet].x=rand()%lungime\_ecran;

bullets[bullet].y=50;

life\_bull[bullet]=raw\_time+rand()%7;

bull\_st[bullet]=1;

bull\_stp[bullet]=1;

}

total\_bull=rand()%60;

current\_time=time(NULL);

total\_bull=rand()%60;

///BOSS SPAWN

struct bossf

{

int x;

int y;

} boss;

boss.x=mijx+1;

boss.y=200;

/\* coordonatele centrului boss-ului\*/

int r\_boss=125;

/\* raza boss-ului\*/

int sh\_r=5;

/\* numarul de straturi ale scutului\*/

int boss\_shieldr=150;

/\* raza scutului boss-ului\*/

int r\_eye=30;

/\* raza ochiului\*/

int dis\_eye=70;

/\* distantadintre ochi\*/

int eyex=mijx-30;

int eyey=150;

/\* coordonatele centrului, primului ochi\*/

int r\_sad=50;

/\* raza gurii\*/

int sady=250;

/\* coordonata de inceput a gurii\*/

int dir\_boss=1;

/\* directia de deplasare a boss-ului\*/

int nr\_shots=0;

/\* numarul de lovituri aplicate boss-ului\*/

int dif\_shots=0;

/\* vechea valoare a nr\_shots in cazul unei noi lovituri\*/

int life\_boss=lungime\_ecran;

/\* viata boss-ului\*/

int damage=100;

/\* pagubele create de o singura lovitura de tun\*/

for(int i=0; i<=7; i++)

circle(boss.x,boss.y,r\_boss-i);

/\* se deseneaza boss-ul\*/

while(life\_boss>0&&life\_player>=0)

{

setactivepage(page);

setvisualpage(1-page);

cleardevice();

setfillstyle(SOLID\_FILL,LIGHTGRAY);

floodfill(1,1,LIGHTGRAY);

int pozplayer[8]= {pozplayer1.x,pozplayer1.y,pozplayer2.x,pozplayer2.y,pozplayer3.x,pozplayer3.y,pozplayer1.x,pozplayer1.y};

setcolor(YELLOW);

drawpoly(4,pozplayer);

setfillstyle(SOLID\_FILL,YELLOW);

fillpoly(4,pozplayer);

current\_time=time(NULL);

///COMANDA MERS DREAPTA

if(GetAsyncKeyState(VK\_RIGHT)&&pozplayer3.x+viteza<=lungime\_ecran)

{

pozplayer1.x+=viteza;

pozplayer2.x+=viteza;

pozplayer3.x+=viteza;

}

///COMANDA MERS STANGA

if(GetAsyncKeyState(VK\_LEFT)&&pozplayer2.x+viteza>=1)

{

pozplayer1.x-=viteza;

pozplayer2.x-=viteza;

pozplayer3.x-=viteza;

}

int ok=0;

///COMANDA SHIELD

if(GetAsyncKeyState(VK\_DOWN))

{

if(timer\_shield<=current\_time&&gun\_fire==0)

{

shield=1;

for(int i=0; i<=5; i++)

arc(pozplayer1.x,pozplayer1.y,0,180,raza\_shield+i);

time\_shield=current\_time+3;

}

else if(timer\_shield>current\_time)

{

setcolor(YELLOW);

//PlaySound(TEXT("DENIED"),NULL,SND\_SYNC);

outtextxy(mijx,mijy,"THE SHIELD IS LOADING...");

}

else if(gun\_fire==1)

{

setcolor(YELLOW);

// PlaySound(TEXT("DENIED"),NULL,SND\_SYNC);

outtextxy(mijx,mijy,"CANNOT USE THE SHIELD WHILE THE CANNON IS ACTIVE !");

}

}

if(time\_shield>=current\_time)

{

setcolor(WHITE);

for(int i=0; i<=5; i++)

arc(pozplayer1.x,pozplayer1.y,0,180,raza\_shield+i);

if(time\_shield==current\_time)

{

timer\_shield=current\_time+5;

shield=0;

}

}

if(timer\_shield>current\_time)

{

setcolor(WHITE);

sprintf(shiel,"THE SHIELD IS READY IN : %d",timer\_shield-current\_time);

outtextxy(10,mijy+50,shiel);

}

else if(time\_shield<=current\_time&&shield==0)

{

setcolor(WHITE);

outtextxy(10,mijy+50,"THE SHIELD IS READY!");

}

///COMANDA TUN

if(GetAsyncKeyState(VK\_SPACE))

if(timer\_gun<=current\_time&&time\_shoot<current\_time&&shield==0)

{

time\_shoot=current\_time+5;

auxi=5;

shootx=pozplayer1.x-3;

shooty=pozplayer1.y;

shootfinx=pozplayer1.x+3;

gun\_fire=1;

nr\_shots++;

}

else if(timer\_gun>current\_time)

{

setcolor(YELLOW);

outtextxy(mijx,mijy,"THE CANNON IS LOADING...");

}

else if(shield==1)

{

setcolor(YELLOW);

outtextxy(mijx,mijy,"CANNOT USE THE CANNON WHILE THE SHIELD IS ACTIVE !");

}

if(time\_shoot>=current\_time)

if(time\_shoot-current\_time!=auxi)

{

bar(shootx-=10,shooty,shootfinx+=10,1);

auxi=time\_shoot-current\_time;

}

else

{

setfillstyle(SOLID\_FILL,auxi);

bar(shootx,shooty,shootfinx,1);

if(time\_shoot==current\_time)

{

timer\_gun=current\_time+10;

gun\_fire=0;

}

}

if(timer\_gun>current\_time)

{

setcolor(WHITE);

sprintf(shiel,"THE CANNON IS READY IN : %d",timer\_gun-current\_time);

outtextxy(10,mijy+75,shiel);

}

else if(timer\_gun<=current\_time&&gun\_fire==0)

{

setcolor(WHITE);

outtextxy(10,mijy+75,"THE CANNON IS READY!");

}

///BOSS CONTROLE

if(boss.x>=lungime\_ecran-r\_boss-viteza\_enem)

{

dir\_boss=0;

boss.x-=viteza\_enem;

eyex-=viteza\_enem;

}

/\* daca boss-ul se afla in extrimitatea din dreapta a ecranului atunci se schimba directia\*/

else if(boss.x<=r\_boss+viteza\_enem)

{

dir\_boss=1;

boss.x+=viteza\_enem;

eyex+=viteza\_enem;

}

/\* daca boss-ul se afla in extrimitatea din dreapta a ecranului atunci se schimba directia\*/

else if(dir\_boss==1)

{

boss.x+=viteza\_enem;

eyex+=viteza\_enem;

}

else if(dir\_boss==0)

{

boss.x-=viteza\_enem;

eyex-=viteza\_enem;

}

/\* deplasarea boss-ului catre dreapta, respectiv stanga\*/

///BOSS SPAWN

setcolor(YELLOW);

for(int i=0; i<=10; i++)

circle(boss.x,boss.y,r\_boss-i);

setcolor(MAGENTA);

for(int i=1; i<=7; i++)

circle(eyex,eyey,r\_eye-i);

for(int i=1; i<=7; i++)

circle(eyex+dis\_eye,eyey,r\_eye-i);

setcolor(BLUE);

for(int i=1; i<=5; i++)

arc(boss.x,sady-i,0,180,r\_sad);

/\* se deseneaza boss-ul (ochii, gura si restul corpului)\*/

///BOSS SHIELD

if(sh\_r)

{/\*daca nu s-au dus toate straturile din scutul boss-ului\*/

int cont=1;

setcolor(WHITE);

for(int i=0; i<sh\_r; i++)

circle(boss.x,boss.y,boss\_shieldr-i);

/\*afiseaza cate straturi au mai ramas\*/

if(gun\_fire==1&&boss.x>=(shootfinx-70)-r\_boss&&boss.x<=shootfinx+r\_boss&&auxi==1)

cont=0;

if(dif\_shots!=nr\_shots&&cont==0)

{

sh\_r--;

dif\_shots=nr\_shots;

}

if(sh\_r==0)

time\_add=current\_time+3;

}

/\* daca tunul este activ si boss-ul se afla in aceasta raza

atunci se verifica daca nu se face o confuzie dintre lovituri,

adica o lovitura sa nu ii scada 2 staturi deodata, in caz afirmativ,

se deuce un strat, iar cand numarul de straturi ajunge la 0, acestea dispar, altfel

scutul boss-ului ar scadea la 0 dupa prima lovitura\*/

if(time\_add>=current\_time)

{

setcolor(YELLOW);

outtextxy(300,mijy,"THE BOSS SHIELD IS DOWN !");

}

/\* daca scutul este distrus, se afiseaza un mesaj corespunzator pe ecran\*/

///LIFE BOSS

if(gun\_fire==1&&boss.x>=(shootfinx-70)-r\_boss&&boss.x<=shootfinx+r\_boss&&auxi==1&&sh\_r<=0)

{

if(dif\_shots!=nr\_shots)

{

life\_boss-=damage;

dif\_shots=nr\_shots;

}

}

/\* daca tunul este activ si boss-ul se afla in aceasta raza

atunci se verifica daca nu se face o confuzie dintre lovituri,

adica o lovitura sa nu aibe echivalentul a doua lovituri, altfel

viata boss-ului ar scadea la 0 dupa prima lovitura\*/

if(sh\_r<=0)

{

setfillstyle(SOLID\_FILL,RED);

bar(0,25,life\_boss,35);

}

/\*daca scutul este distrus, se afiseaza viata boss-ului\*/

/// FIRE CONTROL

for(bullet=1; bullet<=total\_bull; bullet++)

{

if(life\_bull[bullet]<=current\_time&&bull\_st[bullet]==1)

{

bullets[bullet].y+=speed\_bul;

setfillstyle(SOLID\_FILL,RED);

bar(bullets[bullet].x,bullets[bullet].y,bullets[bullet].x+10,bullets[bullet].y+25);

}

if(bullets[bullet].y>=latime\_ecran)

{

bull\_st[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_st[cnt]==1)

contor\_bul++;

if(contor\_bul==0)

total\_bull=0;

}

if(shield==1&&bullets[bullet].x>=pozplayer1.x-raza\_shield&&bullets[bullet].x<=pozplayer1.x+raza\_shield&&bullets[bullet].y>=pozplayer1.y-raza\_shield)

{

bull\_st[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_st[cnt]==1)

contor\_bul++;

if(contor\_bul==0)

total\_bull=0;

}

else if(shield==0&&pozplayer1.x>=bullets[bullet].x&&pozplayer1.x<=bullets[bullet].x+10&&pozplayer1.y>=bullets[bullet].y&&pozplayer1.y<=bullets[bullet].y+25&&bull\_st[bullet]==1&&bull\_stp[bullet]==1)

{

hit=1;

bull\_st[bullet]=0;

bull\_stp[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_st[cnt]==1)

contor\_bul++;

if(contor\_bul==0)

total\_bull=0;

}

else if(shield==0&&pozplayer2.x>=bullets[bullet].x&&pozplayer2.x<=bullets[bullet].x+10&&pozplayer2.y>=bullets[bullet].y&&pozplayer2.y<=bullets[bullet].y+25&&bull\_st[bullet]==1&&bull\_stp[bullet]==1)

{

hit=1;

bull\_st[bullet]=0;

bull\_stp[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_st[cnt]==1)

contor\_bul++;

if(contor\_bul==0)

total\_bull=0;

}

else if(shield==0&&pozplayer3.x>=bullets[bullet].x&&pozplayer3.x<=bullets[bullet].x+10&&pozplayer3.y>=bullets[bullet].y&&pozplayer3.y<=bullets[bullet].y+25&&bull\_st[bullet]==1&&bull\_stp[bullet]==1)

{

hit=1;

bull\_st[bullet]=0;

bull\_stp[bullet]=0;

contor\_bul=0;

for(cnt=1; cnt<=total\_bull; cnt++)

if(bull\_st[cnt]==1)

contor\_bul++;

if(contor\_bul==0)

total\_bull=0;

}

}

///BULLETS RESET

if(total\_bull==0)

{

total\_bull=rand()%60;

for(bullet=1; bullet<=total\_bull; bullet++)

{

bullets[bullet].x=rand()%lungime\_ecran;

bullets[bullet].y=50;

life\_bull[bullet]=current\_time+rand()%7;

bull\_st[bullet]=1;

bull\_stp[bullet]=1;

}

}

///PLAYER'S LIFES

for(bullet=1; bullet<=total\_bull; bullet++)

if ( bull\_stp[bullet]==0)

{

contor\_p=1;

bull\_stp[bullet]=1;

}

if(contor\_p==1)

{

life\_player--;

contor\_p=0;

}

setcolor(YELLOW);

for(cnt=1; cnt<=life\_player; cnt++)

for(int i=1; i<=20; i++)

circle(20+cnt\*50,latime\_ecran-20,i);

page=1-page;

delay(8);

}

if (life\_player<=0)

{

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

char textulet[256]="PRESS SPACE TO RESTART THE STAGE";

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(mijx-textwidth("GAME OVER!")/2,mijy-textheight("GAME OVER!")/2,"GAME OVER!");

settextstyle(BOLD\_FONT,HORIZ\_DIR,2);

setcolor(WHITE);

outtextxy(mijx-textwidth(textulet)/2,3\*mijy/2-textheight(textulet)/2,textulet);

delay(2000);

getch();

car=2;

}

else

{

initwindow(lungime\_ecran,latime\_ecran,"",-3,-3);

char textulet[256]="PRESS SPACE TO CONTINUE TO EXIT";

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

outtextxy(mijx-textwidth("VICTORY!")/2,mijy-textheight("VICTORY!")/2,"VICTORY!");

settextstyle(BOLD\_FONT,HORIZ\_DIR,2);

setcolor(WHITE);

outtextxy(mijx-textwidth(textulet)/2,3\*mijy/2-textheight(textulet)/2,textulet);

getch();

settextstyle(BOLD\_FONT,HORIZ\_DIR,5);

setcolor(YELLOW);

strcpy(textulet,"THANK YOU FOR PLAYING THE GAME!");

outtextxy(mijx-textwidth(textulet)/2,mijy-textheight(textulet)/2,textulet);

settextstyle(BOLD\_FONT,HORIZ\_DIR,2);

strcpy(textulet,"PRESS SPACE TO CONTINUE TO EXIT");

outtextxy(mijx-textwidth(textulet)/2,3\*mijy/2-textheight(textulet)/2,textulet);

delay(2000);

getch();

car=1;

}

closegraph();

if(car==2)

stagiu3(); sau stagiu31(); sau stagiu32()

/\*daca car este egal cu 2, adica daca jucatorul a pierdut se reia stagiul\*/

/\* in rest toata mecanica si comenzile sunt aceleasi ca si la stagiul 1 si 2\*/

return 0;

}