

**1.) Detalii despre realizare**

**2.) Detalii tehnice**

**3.) Cerinte pentru a putea fi jucat**

**4.) Studiu de piata**

**5.) Bibliografie**

**1.) Detalii despre realizare**

**Maze game este o combinatie de jocuri "primitive", fiind dezvoltat pe baza jocurilor bine cunoscute "Snake Game" si "Escape the labirinth" .**

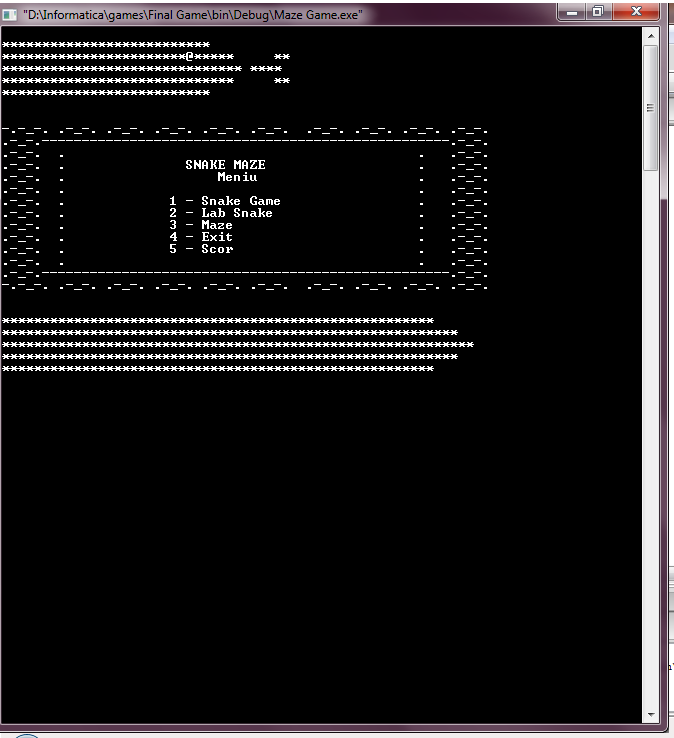
**Este construit din functii elementare ale limbajului C++ : if, for, while ,switch , la care am adaugat 2 functii din librariile <windows.h> si <MMSystem.h> , pentru a adauga muzica ,respectiv a schimba putin aspectul obisnuit , primitiv al jocului. Am realizat jocul in CodeBlocks , folosind exterior acestuia, Audiacity, un program cu ajutorul caruia am editat melodiile care au fost adaugate ca fundal sonor. Singura schimbare pe care am facut-o asupra acesturoa a fost sa le convertesc in formatul .wav .**

**Cpul sarpelui este reprezentat de caracterul ASCII cu codul 15 urmat de caractere cu codul ASCII 7, in cazul in care sarpelui i se mareste coada. Coada se formeaza in momentul in care sarpele mananca frunze, caractere cu codul ASCII 5. Mai jos sunt prezentate capul sapelui, un sarpe cu coada de lungime 4(a mancat 4 frunze) si o frunza.**

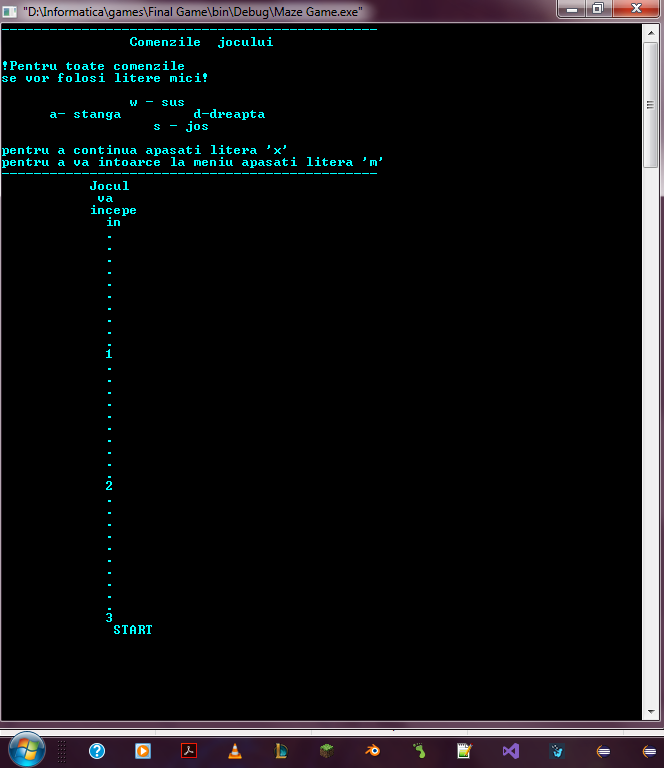
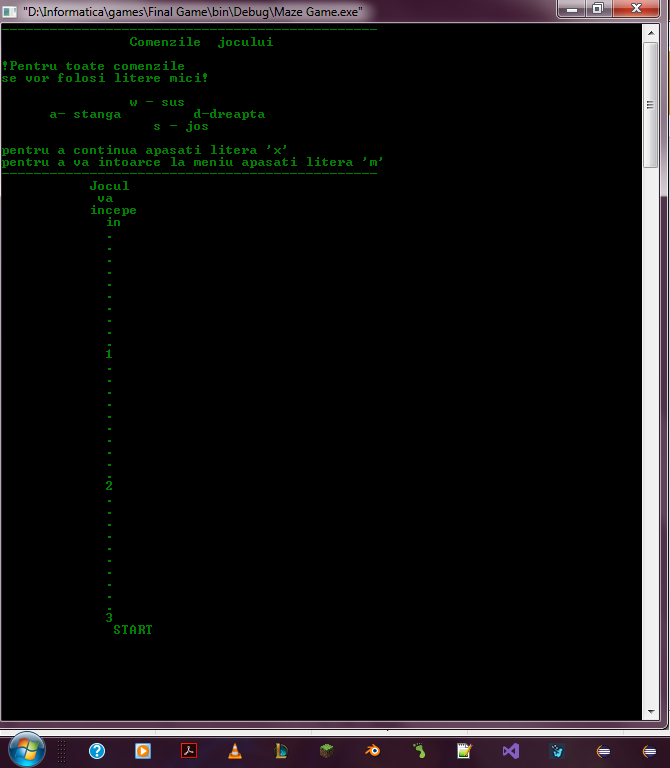
****

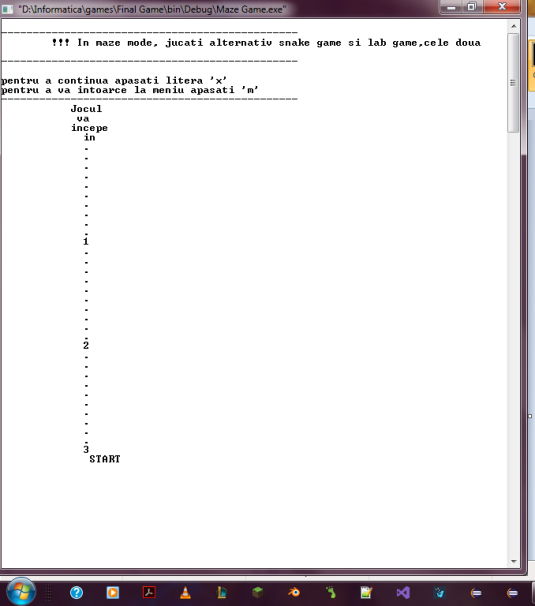
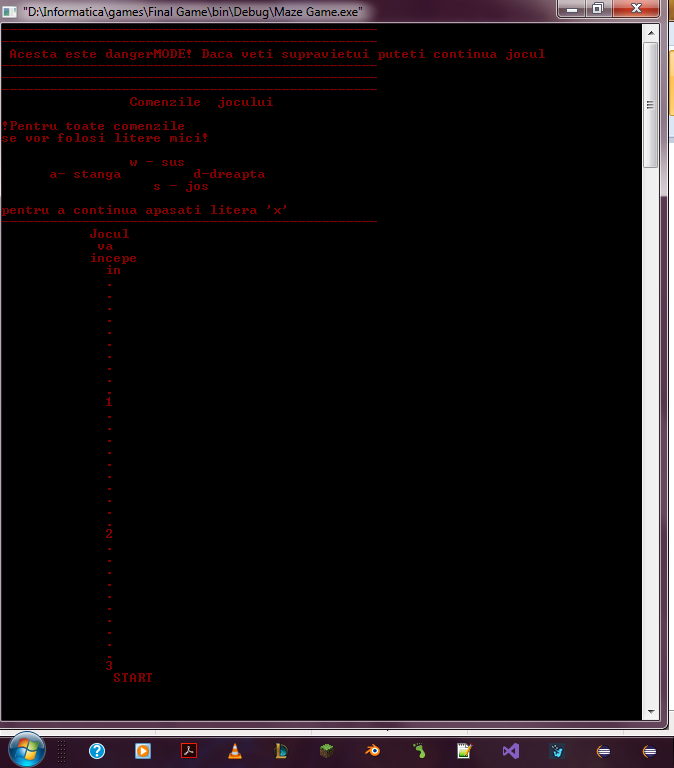
**Alte caractere care au fost folosite in joc au fost sunt "X" , care tine locul unei bombe (snake game), caracterele : "|" , "-" , care reprezeinta conturul hartii.**

**Maze game este construit pe 3 niveluri : Snake Game, Lab Game si Maze Game. Se regasesc si optiunile de EXIT si SCOR. Jocul se deschide direct cu meniul, care este inconjurat de un sarpe fromat din caracterele "\*","@" . In meniu se gasesc cele 5 optiuni. Daca jocatorul apasa alta tasat diferita de 1,2,3,4 sau 5, jocul nu sufera nicio schimbare.**

****

**Fiecare nivel al jocului are o culoare diferita si o melodie diefrita pe fundal. Snake Game :Al Wilson-** "**The Snake" si The piano Brothers - , Lab Game : Donovon - " Lord of The Dance" , iar maze game combina toate cele 3 melodii.De asemenea, snake game are fundalul negru si scisul verde (fundalul negru si scrisul rosu in Danger Mod) , lab game are fundalul negru si scris albastru , iar maze game-ul are la inceput fundalul alb si scris negru , urmand a se schimba in functie de subjoc. Fiecare joc are un panou de instructiuni care se afiseaza inainte sa inceapa jocul , care ii permite jucatorului sa se intoarca la meniul principal sau sa inceapa jocul.**

****

** **

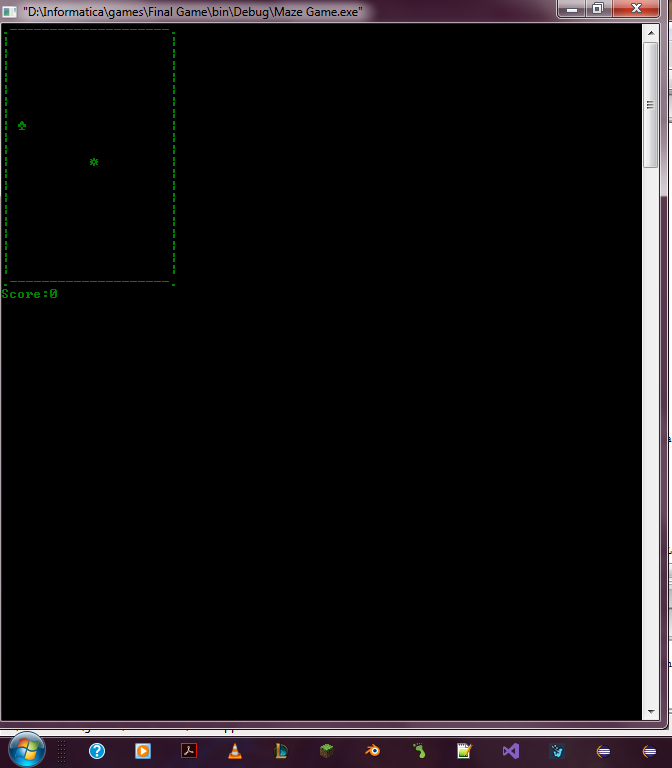
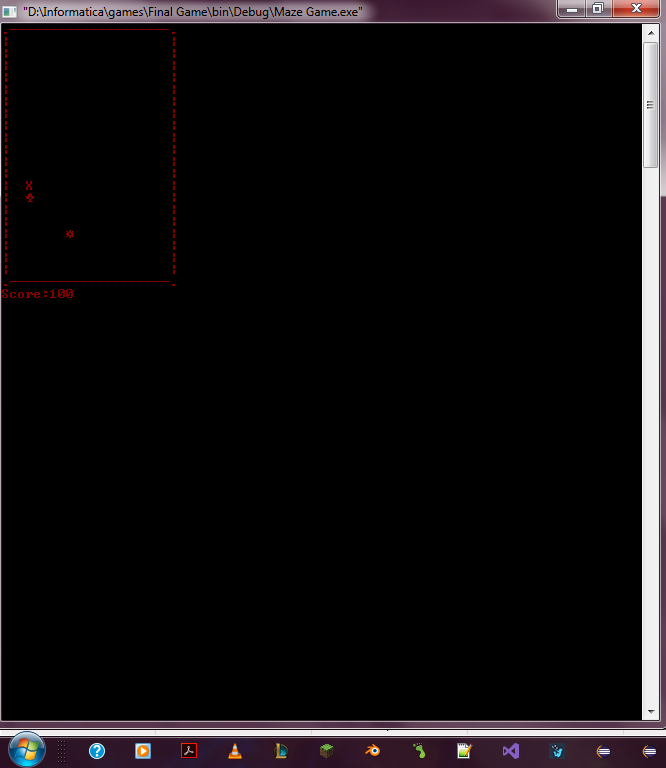
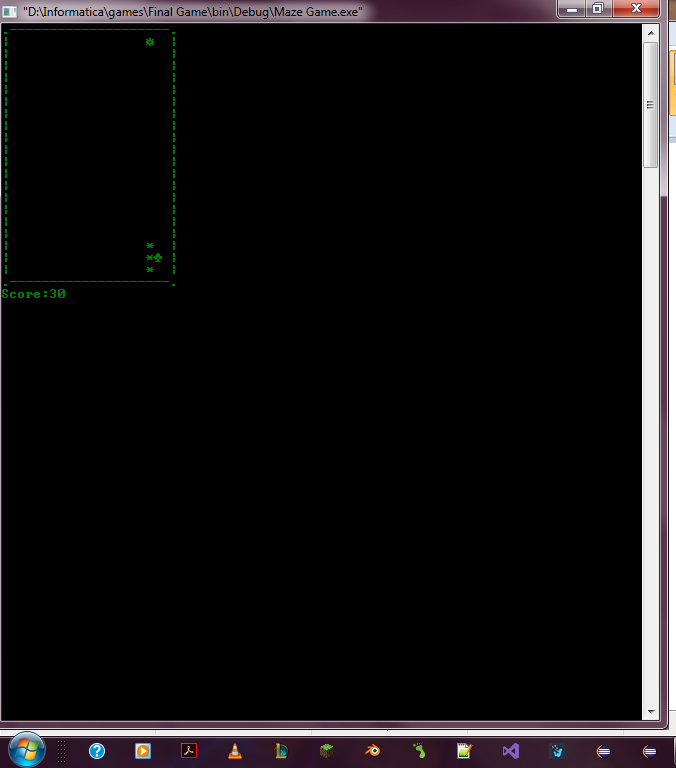
**Primul nivel , Snake Game , poate fi comparat cu un mod "survival" in care jucatorul trebuie sa realizeze un scor cat mai mare. Scorul creste cu 10 atunci cand sarpele mananca o frunza si odata cu scorul creste si coada sarpelui cu un caracter "\*". Atunci cand scorul devine divizibil cu 10 , jucatorul intra intr-un subjoc, DangerMod, in care coada se reseteaza si apar caracterele "X" care au rolul unor bombe . Odata ce un jucator mananca un "X" , jocul se termina si se afiseaza mesajul "GameOver". Jucatorul pierde si in momentul in care isi mananca coada sau apasa o alta tasta diferita de caracterele mici "w","a","s","d". Dupa ce se afiseaza mesajul GameOver,**

****

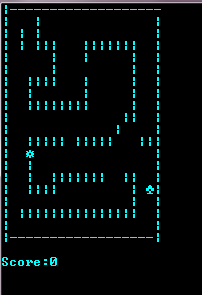
**se afiseaza scorul , care a fost calculat live si mesajul "Nume jucator" . Urmeaza ca jucatorul sa isi scrie numele, care va fi pastrat intr-un fisier de tip text. Mesajul "Nume jucator" apare doar daca jucatorul are un scor > decat 0. De fiecare data cand jucatorul reporneste jocul ,fisierul se reseteaza.**

****

**Jocul are urmatoarul aspect :**

** **

**Cel de-al doilea nivel este asemanator construit. Are fundalul negru si scrisul albastru , pe fundal este Donovon -"The Lord of The Dance". Labirintul este format din semne "-" , "|" .si scopul jucatorului este de face un scor cat mai mare fara a se lovi se peretiii labirintului.**

****

**Al treilea nivel este o combinatie intre primele doua. Jucatorul are doua vieti . Modul incepe cu Snake Game , iar in momentul in care jucatorul moare, urmaeaza Lab Game.**

**2.) Detalii tehnice :**

**Imaginile si ideea jocului este aceea de "suprascriere" in afisare , adica se afiseaza de foarte multe ori o schema, stergandu-se cea veche.**

**Reprezentarea in memorie se face in felul urmator : Saroele, coordonatele cozii , directia sunt reprezentate prin vectori de tip int, harta printr-o matrice de tip char, scorul printr-o matrice de tip int, frunzele,X prin variabile char . Pe langa acestea exista un grup de varinte folosite pentru a retine : coordonatele frunzei , coordonatele bombelor, pentru a genera numere random , pentru a retine coordonate vechi , pentru a retine daca jocatorul a pierdut sau inca joaca(gameOver de tip bool) , flaguri pentru bucle repetitive , variabile pentru sunetul "BEEP" etc. .**

**#include <iostream>**

**#include <conio.h>**

**#include <math.h>**

**#include <windows.h>**

**#include <MMSystem.h>**

**//Biblioteci pentru sunet, culori si functia Sleep()**

**#include <fstream>**

**//Biblioteca pentru a scrie si a citi in fisier**

**using namespace std;**

**char date[] = "scor.txt";**

**ifstream in(date);**

**ofstream out("scor.out");**

**enum eDirecton { STOP = 0, LEFT, RIGHT, UP, DOWN};**

**eDirecton dir;**

**//Directiile pe care se poate deplasa sarpele**

**bool gameOver=true,flag=true,dan,dd=true;**

**const int width = 20;**

**const int height = 20;**

**int ms[19],x, y,xx,yy, fruitX, fruitY, score,tailX[100], tailY[100],nTail,time;**

**char matrice[10][101],cap=15, corp=42 ,fruct=5, sound=7;**

**int nr,fdangerx,fdangery,a=0,timedanger=10,danger=0;**

**char draw1[20][20], caracter='-';**

**bool b =true, mazee = false;**

**void playTheSnake()**

**{**

**PlaySound(TEXT("The Snake - Al Wilson - Copy.wav"),NULL, SND\_FILENAME|SND\_LOOP|SND\_ASYNC);**

**}**

**void playPianoB()**

**{**

**PlaySound(TEXT("Michael Meets Mozar.wav"), NULL, SND\_FILENAME|SND\_LOOP|SND\_ASYNC);**

**}**

**void playDUN()**

**{**

**PlaySound(TEXT("DUN.wav"), NULL, SND\_SYNC);**

**}**

**// Functii de adaugare a muzicii**

**void scriere\_scor()**

**{**

**system ("cls");**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"-----------CLASAMENT SNAKE GAME----------------"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"--- 1---"<<matrice[0]<<" - "<<ms[0]<<"\n";**

**cout<<"--- 2---"<<matrice[1]<<" - "<<ms[1]<<"\n";**

**cout<<"--- 3---"<<matrice[2]<<" - "<<ms[2]<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"-----------CLASAMENT LAB GAME -----------------"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"--- 1---"<<matrice[3]<<" - "<<ms[3]<<"\n";**

**cout<<"--- 2---"<<matrice[4]<<" - "<<ms[4]<<"\n";**

**cout<<"--- 3---"<<matrice[5]<<" - "<<ms[5]<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"-----------CLASAMENT MAZE GAME ----------------"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"--- 1---"<<matrice[6]<<" - "<<ms[6]<<"\n";**

**cout<<"--- 2---"<<matrice[7]<<" - "<<ms[7]<<"\n";**

**cout<<"--- 3---"<<matrice[8]<<" - "<<ms[8]<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"---Apasati 'm' pentru a va intoarce la meniu---"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**}**

**// Scriere clasament**

**void matrice\_scor(int z)**

**{**

**char nume[101];**

**cout<<"Nume jucator : "<<endl;**

**cin.getline(nume,99);**

**cout<<endl;**

**if(z==1)**

**{**

**strcpy(matrice[2], matrice[1]);**

**strcpy(matrice[1], matrice[0]);**

**ms[2]=ms[1];**

**ms[1]=ms[0];**

**ms[0] = score;**

**strcpy(matrice[0], nume);**

**for(int ii=0; ii<9; ii++)**

**out<<matrice[ii]<<endl<<ms[ii]<<endl;**

**}**

**else**

**{**

**if(z==2)**

**{**

**strcpy(matrice[5], matrice[4]);**

**strcpy(matrice[4], matrice[3]);**

**ms[5]=ms[4];**

**ms[4]=ms[3];**

**ms[3] = score;**

**strcpy(matrice[3], nume);**

**for(int ii=0; ii<9; ii++)**

**out<<matrice[ii]<<endl<<ms[ii]<<endl;**

**}**

**else if(z==3)**

**{**

**strcpy(matrice[8], matrice[7]);**

**strcpy(matrice[7], matrice[6]);**

**ms[8]=ms[7];**

**ms[7]=ms[6];**

**ms[6] = score;**

**strcpy(matrice[6], nume);**

**for(int ii=0; ii<9; ii++)**

**out<<matrice[ii]<<endl<<ms[ii]<<endl;**

**}**

**}**

**}**

**// Citire, initializare matrice scor**

**void Setup()**

**{**

**//gameOver = false;**

**dir = STOP;**

**nTail=0;**

**x = width / 2;**

**y = height / 2;**

**for(int i=0; i<nTail; i++)**

**{**

**tailX[i]=0;**

**tailY[i]=0;**

**}**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**}**

**void countdown()**

**{**

**cout<<" Jocul"<<"\n";**

**cout<<" va"<<"\n";**

**cout<<" incepe"<<"\n";**

**cout<<" in"<<"\n";**

**Sleep(10);**

**for(int j=1; j<=3; j++)**

**{**

**for(int i=1; i<=10; i++)**

**{**

**cout<<" ."<<"\n";**

**Sleep(10);**

**}**

**Sleep(25);**

**cout<<" "<<j<<"\n";**

**Sleep(10); //sleep(10);**

**}**

**Sleep(50);**

**cout<<" START ";**

**Sleep(500);**

**}**

**// Scrie o numaratoare inversa pentru a pregati jcuatorul de inceperea jocului**

**void Draw()**

**{**

**system("cls"); //system("clear");**

**for (int i = 0; i < width+2; i++)**

**{**

**if(i==0)**

**cout<<".";**

**else**

**{**

**if(i==width+1)**

**cout<<".";**

**else**

**cout<<'-';**

**}**

**}**

**cout << "\n";**

**for (int i = 0; i < height; i++)**

**{**

**for (int j = 0; j < width; j++)**

**{**

**if (j == 0)**

**cout << "|";**

**if (i == y && j == x)**

**cout <<cap;**

**else if (i == fruitY && j == fruitX)**

**cout << fruct;**

**else**

**{**

**bool print = false;**

**for (int k = 0; k < nTail; k++)**

**{**

**if (tailX[k] == j && tailY[k] == i)**

**{**

**cout <<corp;**

**print = true;**

**}**

**}**

**if (!print)**

**cout << " ";**

**}**

**if (j == width - 1)**

**cout << "|";**

**}**

**cout << endl;**

**}**

**for (int i = 0; i <= width+1; i++)**

**{**

**if(i==0)**

**cout<<".";**

**else**

**{**

**if(i==width+1)**

**cout<<".";**

**else cout << "-";**

**}**

**}**

**cout << "\n";**

**cout << "Score:" << score << endl;**

**}**

**// pentru afisarea matricei harta, sarpelui si a frunzei**

**void Input()**

**{**

**if (\_kbhit())**

**{**

**switch (\_getch())**

**{**

**case 'a':**

**dir = LEFT;**

**break;**

**case 'd':**

**dir = RIGHT;**

**break;**

**case 'w':**

**dir = UP;**

**break;**

**case 's':**

**dir = DOWN;**

**break;**

**default :**

**gameOver = true;**

**break;**

**}**

**}**

**}// pentru initializarea directiei**

**void Logic()**

**{**

**int prevX = tailX[0];**

**int prevY = tailY[0];**

**int prev2X, prev2Y;**

**tailX[0] = x;**

**tailY[0] = y;**

**for (int i = 1; i < nTail; i++)**

**{**

**prev2X = tailX[i];**

**prev2Y = tailY[i];**

**tailX[i] = prevX;**

**tailY[i] = prevY;**

**prevX = prev2X;**

**prevY = prev2Y;**

**}**

**switch (dir)**

**{**

**case LEFT:**

**x--;**

**break;**

**case RIGHT:**

**x++;**

**break;**

**case UP:**

**y--;**

**break;**

**case DOWN:**

**y++;**

**break;**

**default:**

**break;**

**}**

**if (x >= width) x = 0;**

**else if (x < 0) x = width - 1;**

**if (y >= height) y = 0;**

**else if (y < 0) y = height - 1;**

**for (int i = 0; i < nTail; i++)**

**if (tailX[i] == x && tailY[i] == y)**

**gameOver = true;**

**if (x == fruitX && y == fruitY)**

**{**

**score += 10;**

**danger++;**

**nTail++;**

**cout<<sound;**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**}**

**}**

**// pentru modificarea coordonatelor si adaugarea cozii**

**void MazeLogic()**

**{**

**int prevX = tailX[0];**

**int prevY = tailY[0];**

**int prev2X, prev2Y;**

**tailX[0] = x;**

**tailY[0] = y;**

**for (int i = 1; i < nTail; i++)**

**{**

**prev2X = tailX[i];**

**prev2Y = tailY[i];**

**tailX[i] = prevX;**

**tailY[i] = prevY;**

**prevX = prev2X;**

**prevY = prev2Y;**

**}**

**switch (dir)**

**{**

**case LEFT:**

**x--;**

**break;**

**case RIGHT:**

**x++;**

**break;**

**case UP:**

**y--;**

**break;**

**case DOWN:**

**y++;**

**break;**

**default:**

**break;**

**}**

**if (x >= width) x = 0;**

**else if (x < 0) x = width - 1;**

**if (y >= height) y = 0;**

**else if (y < 0) y = height - 1;**

**for (int i = 0; i < nTail; i++)**

**if (tailX[i] == x && tailY[i] == y)**

**gameOver = true;**

**if (x == fruitX && y == fruitY)**

**{**

**score += 10;**

**danger++;**

**nTail++;**

**if(score!=0 && score%50==0)**

**{**

**flag=false;**

**return;**

**}**

**cout<<sound;**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**}**

**}**

**void matriceDraw1()**

**{**

**int i,j;**

**draw1[1][1] = cap;**

**for(i=0; i<20; i++)**

**{**

**draw1[0][i]=draw1[19][i]='-';**

**draw1[i][19]=draw1[i][0]='|';**

**}**

**for(i=1; i<19; i++)**

**for(j=1; j<19; j++)**

**draw1[i][j]=' ';**

**draw1[3][5]=caracter;**

**draw1[3][11]=caracter;**

**draw1[3][12]=caracter;**

**draw1[3][13]=caracter;**

**draw1[3][14]=caracter;**

**draw1[3][15]=caracter;**

**draw1[6][4]=caracter;**

**draw1[6][5]=caracter;**

**draw1[8][4]=caracter;**

**draw1[8][5]=caracter;**

**draw1[8][6]=caracter;**

**draw1[8][7]=caracter;**

**draw1[8][8]=caracter;**

**draw1[8][9]=caracter;**

**draw1[9][15]=caracter;**

**draw1[11][4]=caracter;**

**draw1[11][5]=caracter;**

**draw1[11][6]=caracter;**

**draw1[11][7]=caracter;**

**draw1[11][9]=caracter;**

**draw1[11][10]=caracter;**

**draw1[11][11]=caracter;**

**draw1[11][12]=caracter;**

**draw1[14][7]=caracter;**

**draw1[14][8]=caracter;**

**draw1[14][9]=caracter;**

**draw1[14][10]=caracter;**

**draw1[14][11]=caracter;**

**draw1[14][12]=caracter;**

**draw1[14][15]=caracter;**

**draw1[15][4]=caracter;**

**draw1[15][5]=caracter;**

**draw1[17][3]=caracter;**

**draw1[17][4]=caracter;**

**draw1[17][5]=caracter;**

**draw1[17][6]=caracter;**

**draw1[17][7]=caracter;**

**draw1[17][8]=caracter;**

**draw1[17][9]=caracter;**

**draw1[17][10]=caracter;**

**draw1[17][11]=caracter;**

**draw1[17][12]=caracter;**

**draw1[17][13]=caracter;**

**draw1[17][14]=caracter;**

**draw1[17][15]=caracter;**

**caracter='|';**

**draw1[1][4]=caracter;**

**draw1[2][2]=caracter;**

**draw1[2][4]=caracter;**

**draw1[3][2]=caracter;**

**draw1[3][4]=caracter;**

**draw1[3][6]=caracter;**

**draw1[3][10]=caracter;**

**draw1[3][16]=caracter;**

**draw1[4][6]=caracter;**

**draw1[4][10]=caracter;**

**draw1[4][16]=caracter;**

**draw1[5][6]=caracter;**

**draw1[5][16]=caracter;**

**draw1[6][3]=caracter;**

**draw1[6][6]=caracter;**

**draw1[6][10]=caracter;**

**draw1[6][16]=caracter;**

**draw1[7][3]=caracter;**

**draw1[7][10]=caracter;**

**draw1[7][16]=caracter;**

**draw1[8][3]=caracter;**

**draw1[8][10]=caracter;**

**draw1[8][16]=caracter;**

**draw1[9][16]=caracter;**

**draw1[10][14]=caracter;**

**draw1[11][3]=caracter;**

**draw1[11][13]=caracter;**

**draw1[11][17]=caracter;**

**draw1[11][18]=caracter;**

**draw1[12][3]=caracter;**

**draw1[13][3]=caracter;**

**draw1[14][3]=caracter;**

**draw1[14][6]=caracter;**

**draw1[14][16]=caracter;**

**draw1[15][3]=caracter;**

**draw1[15][6]=caracter;**

**draw1[15][16]=caracter;**

**draw1[16][16]=caracter;**

**draw1[17][2]=caracter;**

**draw1[17][16]=caracter;**

**fruitY = rand() %width ;**

**fruitX = rand() % height;**

**while(draw1[fruitX][fruitY]=='-'|| draw1[fruitX][fruitY]=='|' )**

**{**

**fruitY= rand() % width;**

**fruitX = rand() % height;**

**}**

**draw1[fruitX][fruitY]=5;**

**}**

**void LabSetup()**

**{**

**dir = STOP;**

**switch (dir)**

**{**

**case LEFT:**

**draw1[x][y]=' ';**

**x--;**

**if(draw1[x][y]=='-'||draw1[x][y]=='|')**

**{**

**gameOver=true;**

**return;**

**}**

**draw1[x][y]=15;**

**break;**

**case RIGHT:**

**draw1[x][y]=' ';**

**x++;**

**if(draw1[x][y]=='-'||draw1[x][y]=='|')**

**{**

**gameOver=true;**

**return;**

**}**

**draw1[x][y]=15;**

**break;**

**case UP:**

**draw1[x][y]=' ';**

**y++;**

**if(draw1[x][y]=='-'||draw1[x][y]=='|')**

**{**

**gameOver=true;**

**return;**

**}**

**draw1[x][y]=15;**

**break;**

**case DOWN:**

**draw1[x][y]=' ';**

**y--;**

**if(draw1[x][y]=='-'||draw1[x][y]=='|')**

**{**

**gameOver=true;**

**return;**

**}**

**draw1[x][y]=15;**

**break;**

**default:**

**break;**

**}**

**if (x == fruitX && y == fruitY)**

**{**

**score += 10;**

**cout<<sound;**

**int ss=fruitX, tt=fruitY;**

**while(draw1[fruitX][fruitY]=='-'|| draw1[fruitX][fruitY]=='|' )**

**{**

**fruitY = rand() % width;**

**fruitX = rand() % height;**

**}**

**draw1[ss][tt]=0;**

**draw1[fruitX][fruitY]=5;**

**}**

**/\*if( draw1[x][y] =='-' || draw1[x][y]=='|')**

**{**

**gameOver=true;**

**return;**

**}\*/**

**}**

**void Draw1()**

**{**

**system("cls"); //system("clear");**

**int i,j;**

**for(i=0; i<20; i++)**

**{**

**if(x==0 || x == 19)**

**x = 1;**

**if(y==0 || y==19)**

**y =1;**

**draw1[y][x] = cap;**

**for(j=0; j<20; j++)**

**{**

**cout<<draw1[i][j];**

**}**

**cout<<"\n";**

**}**

**cout << "\n";**

**cout << "Score:" << score << endl;**

**}**

**void LabInput()**

**{**

**if (\_kbhit())**

**{**

**switch (\_getch())**

**{**

**case 'a':**

**dir = LEFT;**

**break;**

**case 'd':**

**dir = RIGHT;**

**break;**

**case 'w':**

**dir = UP;**

**break;**

**case 's':**

**dir = DOWN;**

**break;**

**default :**

**gameOver = true;**

**break;**

**}**

**}**

**}**

**void LabLogic()**

**{**

**switch (dir)**

**{**

**case LEFT:**

**draw1[y][x]=' ';**

**x--;**

**if(draw1[y][x]=='-'||draw1[y][x]=='|')**

**{**

**gameOver=true;**

**return;**

**}**

**else if (y == fruitX && x == fruitY)**

**{**

**score += 10;**

**cout<<sound;**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**while(draw1[fruitX][fruitY]=='-'|| draw1[fruitX][fruitY]=='|' )**

**{**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**}**

**draw1[fruitX][fruitY]=5;**

**}**

**draw1[y][x]=5;**

**break;**

**case RIGHT:**

**draw1[y][x]=' ';**

**x++;**

**if(draw1[y][x]=='-'||draw1[y][x]=='|')**

**{**

**gameOver=true;**

**return;**

**}**

**else if (y == fruitX && x == fruitY)**

**{**

**score += 10;**

**cout<<sound;**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**while(draw1[fruitX][fruitY]=='-'|| draw1[fruitX][fruitY]=='|' )**

**{**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**}**

**draw1[fruitX][fruitY]=5;**

**}**

**draw1[y][x]=5;**

**break;**

**case UP:**

**draw1[y][x]=' ';**

**y--;**

**if(draw1[y][x]=='-'||draw1[y][x]=='|')**

**{**

**gameOver=true;**

**return;**

**}**

**else if (y == fruitX && x == fruitY)**

**{**

**score += 10;**

**cout<<sound;**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**while(draw1[fruitX][fruitY]=='-'|| draw1[fruitX][fruitY]=='|' )**

**{**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**}**

**draw1[fruitX][fruitY]=5;**

**}**

**draw1[y][x]=5;**

**break;**

**case DOWN:**

**draw1[y][x]=' ';**

**y++;**

**if(draw1[y][x]=='-'||draw1[y][x]=='|')**

**{**

**gameOver=true;**

**return;**

**}**

**else if (y == fruitX && x == fruitY)**

**{**

**score += 10;**

**cout<<sound;**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**while(draw1[fruitX][fruitY]=='-'|| draw1[fruitX][fruitY]=='|' )**

**{**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**}**

**draw1[fruitX][fruitY]=5;**

**}**

**draw1[y][x]=5;**

**break;**

**default:**

**break;**

**}**

**/\***

**if (x == fruitX && y == fruitY)**

**{**

**score += 10;**

**cout<<sound;**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**while(draw1[fruitX][fruitY]=='-'|| draw1[fruitX][fruitY]=='|' )**

**{**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**}**

**draw1[fruitX][fruitY]=5;**

**}**

**/\*if( draw1[x][y] =='-' || draw1[x][y]=='|')**

**{**

**gameOver=true;**

**return;**

**}\*/**

**}**

**void meniu()**

**{**

**PlaySound(TEXT("Donovan.wav"),NULL, SND\_FILENAME|SND\_LOOP|SND\_ASYNC);**

**system("Color 0F");**

**cout<<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" <<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*@\*\*\*\*\* \*\*" <<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*" <<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*" <<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" <<"\n";**

**cout<<"\n";**

**cout<<"\n";**

**cout<<"-.-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-."<<"\n";**

**cout<<".-\_-.---------------------------------------------------.-\_-."<<"\n";**

**cout<<".-\_-. . . .-\_-."<<"\n";**

**cout<<".-\_-. . SNAKE MAZE . .-\_-."<<"\n";**

**cout<<".-\_-. . Meniu . .-\_-."<<"\n";**

**cout<<".-\_-. . . .-\_-."<<"\n";**

**cout<<".-\_-. . 1 - Snake Game . .-\_-."<<"\n";**

**//cout<<".-\_-. . 2 - Jump Snake . .-\_-."<<"\n";**

**cout<<".-\_-. . 2 - Lab Snake . .-\_-."<<"\n";**

**cout<<".-\_-. . 3 - Maze . .-\_-."<<"\n";**

**cout<<".-\_-. . 4 - Exit . .-\_-."<<"\n";**

**cout<<".-\_-. . 5 - Scor . .-\_-."<<"\n";**

**cout<<".-\_-. . . .-\_-."<<"\n";**

**cout<<".-\_-.---------------------------------------------------.-\_-."<<"\n";**

**cout<<"-.-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-."<<"\n";**

**cout<<"\n";**

**cout<<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<"\n";**

**cout<<"\n";**

**return;**

**}**

**//afisare meniu**

**void gameover()**

**{**

**cout<<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" <<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*@\*\*\*\*\* \*\*" <<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*" <<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*" <<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" <<"\n";**

**cout<<"\n";**

**cout<<"-.-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-."<<"\n";**

**cout<<".-\_-.--------------------------------------------------- .-\_-."<<"\n";**

**cout<<".-\_-. . . .-\_-."<<"\n";**

**cout<<".-\_-. . . .-\_-."<<"\n";**

**cout<<".-\_-. . GAME . .-\_-."<<"\n";**

**cout<<".-\_-. . . .-\_-."<<"\n";**

**cout<<".-\_-. . OVER . .-\_-."<<"\n";**

**cout<<".-\_-. . . .-\_-."<<"\n";**

**cout<<".-\_-. . . .-\_-."<<"\n";**

**cout<<".-\_-. . . .-\_-."<<"\n";**

**cout<<".-\_-. . . .-\_-."<<"\n";**

**cout<<".-\_-.---------------------------------------------------- .-\_-."<<"\n";**

**cout<<"-.-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-."<<"\n";**

**cout<<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<"\n";**

**cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<"\n";**

**cout<<"\n";**

**}**

**// afisare mesaj game over**

**void snake\_game\_ruls()**

**{**

**PlaySound(TEXT("The Snake - Al Wilson - Copy.wav"),NULL, SND\_FILENAME|SND\_LOOP|SND\_ASYNC);**

**playTheSnake();**

**system ("cls");**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<" Comenzile jocului "<<"\n";**

**cout<<"\n";**

**cout<<"!Pentru toate comenzile"<<"\n";**

**cout<<"se vor folosi litere mici!"<<"\n";**

**cout<<"\n";**

**cout<<" w - sus"<<"\n";**

**cout<<" a- stanga d-dreapta"<<"\n";**

**cout<<" s - jos"<<"\n";**

**cout<<"\n";**

**cout<<"pentru a continua apasati litera 'x'"<<"\n";**

**cout<<"pentru a va intoarce la meniu apasati litera 'm'"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**}**

**// afisare instructiuni**

**void maze\_snake\_game\_ruls()**

**{**

**PlaySound(TEXT("The Snake - Al Wilson - Copy.wav"),NULL, SND\_FILENAME|SND\_LOOP|SND\_ASYNC);**

**system ("cls");**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<" Comenzile jocului "<<"\n";**

**cout<<"\n";**

**cout<<"!Pentru toate comenzile"<<"\n";**

**cout<<"se vor folosi litere mici!"<<"\n";**

**cout<<"\n";**

**cout<<" w - sus"<<"\n";**

**cout<<" a- stanga d-dreapta"<<"\n";**

**cout<<" s - jos"<<"\n";**

**cout<<"\n";**

**cout<<"pentru a continua apasati litera 'x'"<<"\n";**

**//cout<<"pentru a schimba cu jocul snake game apasati 'm'"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**}**

**void scor()**

**{**

**cout<<"\n";**

**cout<<"-.-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-."<<"\n";**

**cout<<".-\_-.---------------------------.-\_-."<<"\n";**

**cout<<".-\_-. . . .-\_-."<<"\n";**

**cout<<".-\_-. . SCOR . .-\_-."<<"\n";**

**cout<<".-\_-. . . .-\_-."<<"\n";**

**cout<<".-\_-. . "<<score<<" . .-\_-."<<"\n";**

**cout<<".-\_-.-------------------------- .-\_-."<<"\n";**

**cout<<"-.-\_-. .-\_-. .-\_-. .-\_-. .-\_-. .-\_-."<<"\n";**

**cout<<"\n";**

**}**

**// afisare scor**

**void DangerSetup()**

**{**

**a=0;**

**nTail=0;**

**for(int i=0; i<nTail; i++)**

**{**

**tailX[i]=tailY[i]=0;**

**}**

**nTail=0;**

**danger=0;**

**dir = STOP;**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**while (fruitX == x && fruitY==y)**

**{**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**}**

**xx= rand() % width;**

**yy= rand() % height;**

**while((xx==fruitX && yy==fruitY) ||(yy==fruitX && xx==fruitY))**

**{**

**xx= rand() % width;**

**yy= rand() % height;**

**}**

**for(int t=0; t <nTail; t++)**

**if(xx == tailX[t] && yy == tailY[t])**

**{**

**xx= rand() % width;**

**yy= rand() % height;**

**}**

**}**

**void DangerDraw()**

**{**

**system("cls"); //system("clear");**

**for (int i = 0; i < width+2; i++)**

**{**

**if(i==0)**

**cout<<".";**

**else**

**{**

**if(i==width+1)**

**cout<<".";**

**else**

**cout<<'-';**

**}**

**}**

**cout << "\n";**

**for (int i = 0; i < height; i++)**

**{**

**for (int j = 0; j < width; j++)**

**{**

**if (j == 0)**

**cout << "|";**

**if (i == y && j == x)**

**cout <<cap;**

**else if (i == fruitY && j == fruitX)**

**cout << fruct;**

**else**

**{**

**if(i ==xx && j == yy)**

**cout<<"X";**

**else**

**{**

**bool print = false;**

**for (int k = 0; k < nTail; k++)**

**{**

**if (tailX[k] == j && tailY[k] == i)**

**{**

**cout <<corp;**

**print = true;**

**}**

**}**

**if (!print)**

**cout << " ";**

**}**

**}**

**if (j == width - 1)**

**cout << "|";**

**}**

**cout << endl;**

**}**

**for (int i = 0; i <= width+1; i++)**

**{**

**if(i==0)**

**cout<<".";**

**else**

**{**

**if(i==width+1)**

**cout<<".";**

**else cout << "-";**

**}**

**}**

**cout << "\n";**

**cout << "Score:" << score << endl;**

**}**

**void DangerInput()**

**{**

**if (\_kbhit())**

**{**

**switch (\_getch())**

**{**

**case 'a':**

**dir = LEFT;**

**break;**

**case 'd':**

**dir = RIGHT;**

**break;**

**case 'w':**

**dir = UP;**

**break;**

**case 's':**

**dir = DOWN;**

**break;**

**default :**

**gameOver = true;**

**break;**

**}**

**}**

**}**

**void DangerLogic()**

**{**

**int prevX = tailX[0];**

**int prevY = tailY[0];**

**int prev2X, prev2Y;**

**tailX[0] = x;**

**tailY[0] = y;**

**for (int i = 1; i < nTail; i++)**

**{**

**prev2X = tailX[i];**

**prev2Y = tailY[i];**

**tailX[i] = prevX;**

**tailY[i] = prevY;**

**prevX = prev2X;**

**prevY = prev2Y;**

**}**

**switch (dir)**

**{**

**case LEFT:**

**x--;**

**break;**

**case RIGHT:**

**x++;**

**break;**

**case UP:**

**y--;**

**break;**

**case DOWN:**

**y++;**

**break;**

**default:**

**break;**

**}**

**if (x >= width) x = 0;**

**else if (x < 0) x = width - 1;**

**if (y >= height) y = 0;**

**else if (y < 0) y = height - 1;**

**for (int i = 0; i < nTail; i++)**

**if (tailX[i] == x && tailY[i] == y)**

**{**

**gameOver = true;**

**a=100;**

**}**

**if (x == fruitX && y == fruitY)**

**{**

**score += 10;**

**a++;**

**nTail++;**

**cout<<sound;**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**while (fruitX == x && fruitY==y)**

**{**

**fruitX = rand() % width;**

**fruitY = rand() % height;**

**}**

**xx= rand() % width;**

**yy= rand() % height;**

**while ((xx==fruitX && yy==fruitY) || (yy==fruitX && xx==fruitY))**

**{**

**xx= rand() % width;**

**yy= rand() % height;**

**}**

**}**

**if((x == xx && y == yy)||(y == xx && x == yy))**

**{**

**a=100;**

**gameOver=true;**

**return;**

**}**

**}**

**void danger\_snake\_ruls()**

**{**

**PlaySound(TEXT("Michael Meets Mozar.wav"), NULL, SND\_FILENAME|SND\_LOOP|SND\_ASYNC);**

**system ("cls");**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<" Acesta este dangerMODE! Daca veti supravietui puteti continua jocul "<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<" Comenzile jocului "<<"\n";**

**cout<<"\n";**

**cout<<"!Pentru toate comenzile"<<"\n";**

**cout<<"se vor folosi litere mici!"<<"\n";**

**cout<<"\n";**

**cout<<" w - sus"<<"\n";**

**cout<<" a- stanga d-dreapta"<<"\n";**

**cout<<" s - jos"<<"\n";**

**cout<<"\n";**

**cout<<"pentru a continua apasati litera 'x'"<<"\n";**

**//cout<<"pentru a va intoarce la meniu apasati litera 'm'"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**}**

**void maze\_danger\_snake()**

**{**

**system("Color 04");**

**switch (\_getch())**

**{**

**case 'x':**

**{**

**countdown();**

**DangerSetup();**

**while(a<5)**

**{**

**DangerDraw();**

**DangerInput();**

**DangerLogic();**

**Sleep(50);**

**}**

**break;**

**}**

**}**

**return;**

**}**

**void danger\_snake()**

**{**

**system("Color 04");**

**switch (\_getch())**

**{**

**case 'x':**

**{**

**countdown();**

**DangerSetup();**

**while(a<5)**

**{**

**DangerDraw();**

**DangerInput();**

**DangerLogic();**

**Sleep(50);**

**}**

**break;**

**}**

**}**

**return;**

**}**

**void snake\_game()**

**{**

**//PlaySound(TEXT("The Snake - Al Wilson - Copy.wav"),NULL, SND\_FILENAME|SND\_LOOP|SND\_ASYNC);**

**system("Color 02");**

**time=50;**

**switch (\_getch())**

**{**

**case 'x':**

**{**

**countdown();**

**Setup();**

**while (!gameOver)**

**{**

**if(danger == timedanger)**

**{**

**danger\_snake\_ruls();**

**danger\_snake();**

**snake\_game\_ruls();**

**}**

**system("Color 02");**

**Draw();**

**Input();**

**Logic();**

**Sleep(time);**

**if(time>5)**

**time-=1/100;**

**}**

**}**

**case 'm':**

**{**

**b = false;**

**return;**

**}**

**}**

**}**

**void jump\_game\_ruls()**

**{**

**}**

**void jump\_game()**

**{**

**}**

**void maze\_lab\_game\_ruls()**

**{**

**system("Color 0B");**

**system ("cls");**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<" Comenzile jocului "<<"\n";**

**cout<<"\n";**

**cout<<"!Pentru toate comenzile"<<"\n";**

**cout<<"se vor folosi litere mici!"<<"\n";**

**cout<<"\n";**

**cout<<" w - sus"<<"\n";**

**cout<<" a- stanga d-dreapta"<<"\n";**

**cout<<" s - jos"<<"\n";**

**cout<<"\n";**

**cout<<"pentru a continua apasati litera 'x'"<<"\n";**

**//cout<<"pentru a schimba cu jocul snake game apasati 'm'"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**}**

**void lab\_game\_ruls()**

**{**

**system("Color 0B");**

**system ("cls");**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<" Comenzile jocului "<<"\n";**

**cout<<"\n";**

**cout<<"!Pentru toate comenzile"<<"\n";**

**cout<<"se vor folosi litere mici!"<<"\n";**

**cout<<"\n";**

**cout<<" w - sus"<<"\n";**

**cout<<" a- stanga d-dreapta"<<"\n";**

**cout<<" s - jos"<<"\n";**

**cout<<"\n";**

**cout<<"pentru a continua apasati litera 'x'"<<"\n";**

**cout<<"pentru a va intoarce la meniu apasati litera 'm'"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**}**

**void lab\_game()**

**{**

**matriceDraw1();**

**time=150;**

**switch (\_getch())**

**{**

**case 'x':**

**{**

**countdown();**

**LabSetup();**

**while (!gameOver)**

**{**

**Draw1();**

**LabInput();**

**LabLogic();**

**Sleep(time);**

**if(time>5)**

**time-=1/100;**

**}**

**if(gameOver)**

**{**

**if(score > ms[6] && !mazee)**

**matrice\_scor(2);**

**return;**

**}**

**}**

**case 'm':**

**{**

**b = false;**

**return;**

**}**

**}**

**}**

**void maze\_lab\_game()**

**{**

**matriceDraw1();**

**time=150;**

**switch (\_getch())**

**{**

**case 'x':**

**{**

**countdown();**

**LabSetup();**

**while (score%60 !=0 && !gameOver)**

**{**

**Draw1();**

**LabInput();**

**LabLogic();**

**Sleep(time);**

**if(time>5)**

**time-=1/100;**

**}**

**}**

**}**

**}**

**void maze\_ruls()**

**{**

**system("Color F0");**

**system ("cls");**

**cout<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<" !!! In maze mode, jucati alternativ snake game si lab game,cele doua "<<"\n";**

**cout<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**cout<<"\n";**

**/\*cout<<"-----------------------------------------------"<<"\n";**

**cout<<" Comenzile jocului "<<"\n";**

**cout<<"\n";**

**cout<<"!Pentru toate comenzile"<<"\n";**

**cout<<"se vor folosi litere mici!"<<"\n";**

**cout<<"\n";**

**cout<<" w - sus"<<"\n";**

**cout<<" a- stanga d-dreapta"<<"\n";**

**cout<<" s - jos"<<"\n";**

**cout<<"\n";\*/**

**cout<<"pentru a continua apasati litera 'x'"<<"\n";**

**cout<<"pentru a va intoarce la meniu apasati 'm'"<<"\n";**

**cout<<"-----------------------------------------------"<<"\n";**

**}**

**void maze\_snake\_game()**

**{**

**system("Color 02");**

**time=50;**

**switch (\_getch())**

**{**

**case 'x':**

**{**

**countdown();**

**Setup();**

**while (!gameOver)**

**{**

**if(danger == timedanger)**

**{**

**danger\_snake\_ruls();**

**danger\_snake();**

**snake\_game\_ruls();**

**}**

**system("Color 02");**

**Draw();**

**Input();**

**Logic();**

**Sleep(time);**

**if(time>5)**

**time-=1/100;**

**}**

**}**

**case 'm':**

**{**

**b = false;**

**gameOver = true;**

**return;**

**}**

**}**

**}**

**void maze()**

**{**

**mazee = true;**

**switch (\_getch())**

**{**

**case 'x':**

**{**

**countdown();**

**Setup();**

**while (!gameOver)**

**{**

**mazee = true;**

**flag=true;**

**maze\_snake\_game\_ruls();**

**time = 50;**

**snake\_game();**

**if(gameOver==true)**

**{**

**gameOver = false;**

**flag=true;**

**maze\_lab\_game\_ruls();**

**lab\_game();**

**}**

**mazee = false;**

**}**

**}**

**if(score > ms[3])**

**matrice\_scor(3);**

**case 'm':**

**{**

**b = false;**

**return;**

**}**

**}**

**}**

**int main()**

**{**

**/\* //verificare matricea Map**

**int i,j;**

**map();**

**for(i=1; i<=n; i++)**

**{**

**for(j=1; j<=n; j++)**

**cout<<Map[i][j]<<" ";**

**cout<<"\n";**

**} \*/**

**for(int ii=0; ii<9; ii++)**

**{**

**in.get(matrice[ii],50);**

**in>>ms[ii];**

**in.get();**

**}**

**bool gameOver2 = true;**

**system("Color 0F");**

**while(gameOver2)**

**{**

**system("cls"); //system("clear");**

**meniu();**

**score=0;**

**b = true;**

**switch (\_getch())**

**{**

**case '1':**

**{**

**if(b)**

**{**

**gameOver=false;**

**snake\_game\_ruls();**

**snake\_game();**

**if(gameOver)**

**{**

**playDUN();**

**gameover();**

**Sleep(1000);**

**scor();**

**if(score > ms[0])**

**matrice\_scor(1);**

**Sleep(900);**

**}**

**break;**

**}**

**}**

**/\*case '2':**

**{**

**gameOver=false;**

**jump\_game\_ruls();**

**jump\_game();**

**break;**

**}\*/**

**case '2':**

**{**

**if(b)**

**{**

**gameOver=false;**

**lab\_game\_ruls();**

**lab\_game();**

**if(gameOver)**

**{**

**playDUN();**

**gameover();**

**Sleep(1000);**

**scor();**

**if(score > ms[3])**

**matrice\_scor(2);**

**Sleep(900);**

**}**

**break;**

**}**

**}**

**case '3':**

**{**

**if(b)**

**{**

**gameOver=false;**

**maze\_ruls();**

**maze();**

**if(gameOver)**

**{**

**playDUN();**

**gameover();**

**Sleep(1000);**

**scor();**

**if(score > ms[6])**

**matrice\_scor(3);**

**Sleep(900);**

**}**

**break;**

**}**

**}**

**case '4':**

**{**

**b = false;**

**gameOver2=false;**

**break;**

**}**

**case '5':**

**{**

**scriere\_scor();**

**switch (\_getch())**

**{**

**case 'm':**

**b = false;**

**break;**

**}**

**}**

**}**

**if(b && gameOver== false)**

**{**

**playDUN();**

**gameover();**

**Sleep(1000);**

**scor();**

**Sleep(900);**

**}**

**//for(int ii=0;ii<9;ii++)**

**// out<<matrice[ii]<<ms[ii]<<endl;**

**}**

**return 0;**

**}**

**3.) Cerinte pentru a putea fi jucat**

**Functioneaza pe orice calculator cu Windows XP,7 sau 7 professional.**

**Nu necesita instalarea altui program alternativ ca CodeBlocks sau Audiacty.**

**Nu necesita acces la internet.**

**4.) Studiul pietii**

**Jocul este dedicat tuturor celor care doresc sa il incerce**

**In urma unui sondaj , jocul a avut cel mai mare impact asupra categorii de varsta 8-12 ani**

**Jocul poate fi promovat pe retelele de socializare prin distribuire.**

**Urmeaza a fi incarcat pe site-uri ca Steam, Filelist ,Seedfile, astefel incat toata lumea sa aiba acces la el.**

**5.) Bibliografie**

**-Sleep()**

**-system("clr")**

**-Random**

**-playNume\_Melodie();**

**de pe www.cplusplus.com**