**Snake**

**-Jocul**

**Cuprins**

**3…………………………………….Ce este Snake?**

**4………………………………………De ce Snake?**

**5……………………………….…Cerinte de sistem  
6…………………..………Prezentarea proiectului**

**20……………………Metoda folosita,inbunatatiri**

**Ce este Snake?**

**Jocul Snake a aparul la sfarsitul anilor 1970, dar si-a castigat popularitatea dupa 1998, cand a fost pus pe Nokia 3310, gratie unui inginer care lucra la Nokia, Taneli Armanto. Numeroase versiuni au fost create de-a lungul timpului, iar in anii trecuti au aparut inclusiv versiuni 3D de Snake.**

**Primele telefoane Nokia pe care a fost jocul sunt 6110 si 8810. Ambele aveau ecran monocrom, la fel ca Nokia 3310, telefonul care a facut celebru sarpete care inghite patratele.**

**Snake ramane un joc de referinta in istoria telefoanelor mobile, asa ca nu e de mirare faptul ca exista aplicatii Snake chiar si pentru smartphone-urile care domina astazi piata.**

**De ce Snake?**

**Deoarece este un joc relative simplu de implementat,am decis sa inceram sa recreez cunoscutul joc in aplicatia oferita de Microsoft,mai exact Microsoft Visual Studio 2013.**

**Programand in limbajul C#,motivati de dorinta de a creea un joc pentru prima data,am reusit dupa cateva ore de implementat si alte cateva ore de design sa creem o versiune asemanatoare originalului snake“vanilla”,fara obstacole in interiorul nivelului.**

**Simplu dar efficient,jocul poate captiva atentia unui copil pentru ceva timp,fara a avea nevoie de grafici pompoase.**

**Cerinte de sistem**

**Platforma de operare:**

**Jocul a fost programat in Windows 7.**

**Ruleaza pe:**

**- Windows 7**

**- Windows 8**

**- Windows 8.1**

**- Windows 10**

**Placa video:**

**-Minim 512mb video dedicat/integrat**

**Spatiu necesar:**

**-Minim 10mb**

**Procesor :**

**-2gb Dual-Core sau mai bun**

**Placa de sunet:nu este necesara**

**Prezentarea Proiectului**

****

1.Click pentru a incepe un joc nou

2.Click pentru a arata intructiunile de joc

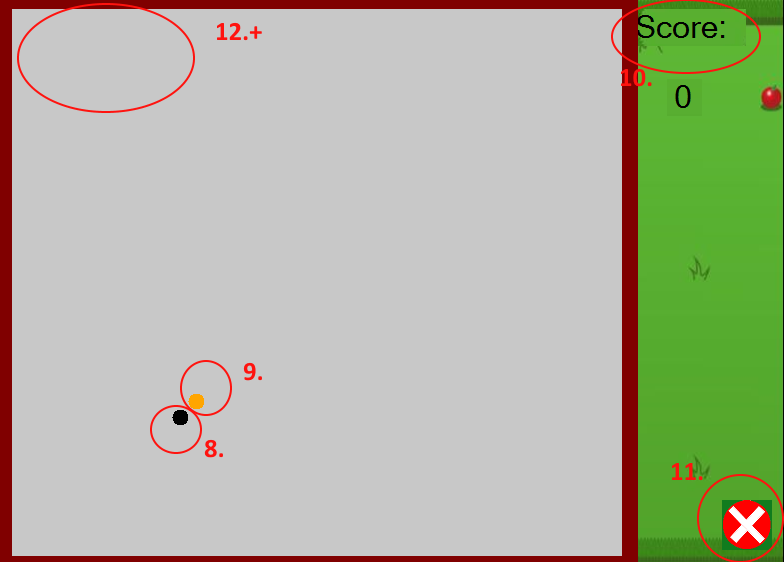
3.Click pentru a afla despre autori

4.Click pentru a afla despre joc

5.Click pentru a porni muzica

6.Click pentru a opri muzica

7.Click pentru a opri aplicatia

****

8.Sarpe(Jucator)

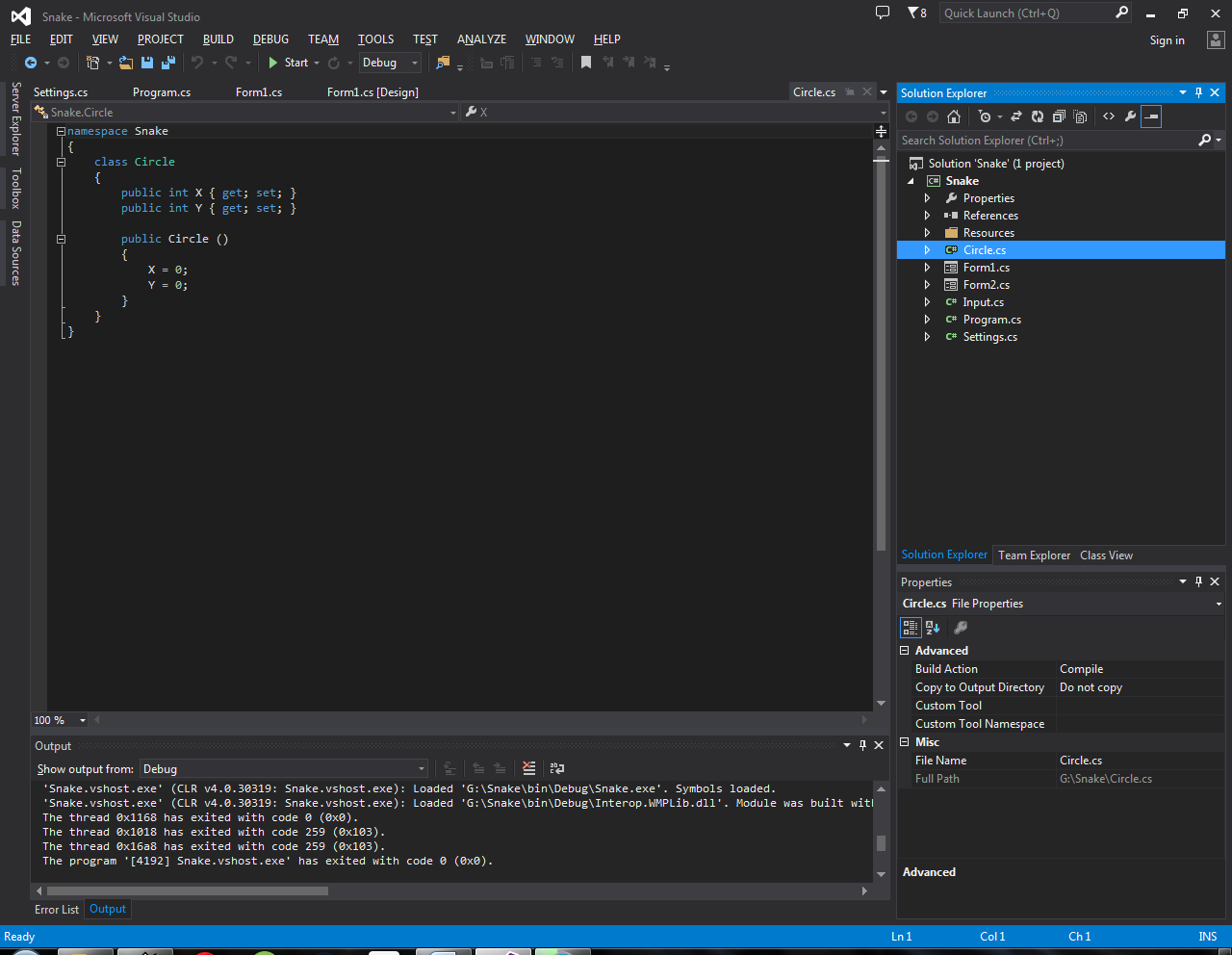
9.Mancare

10.Scor,aici vor fi afisate punctele pe care lestrangi in repriza curenta

11.Opreste jocul

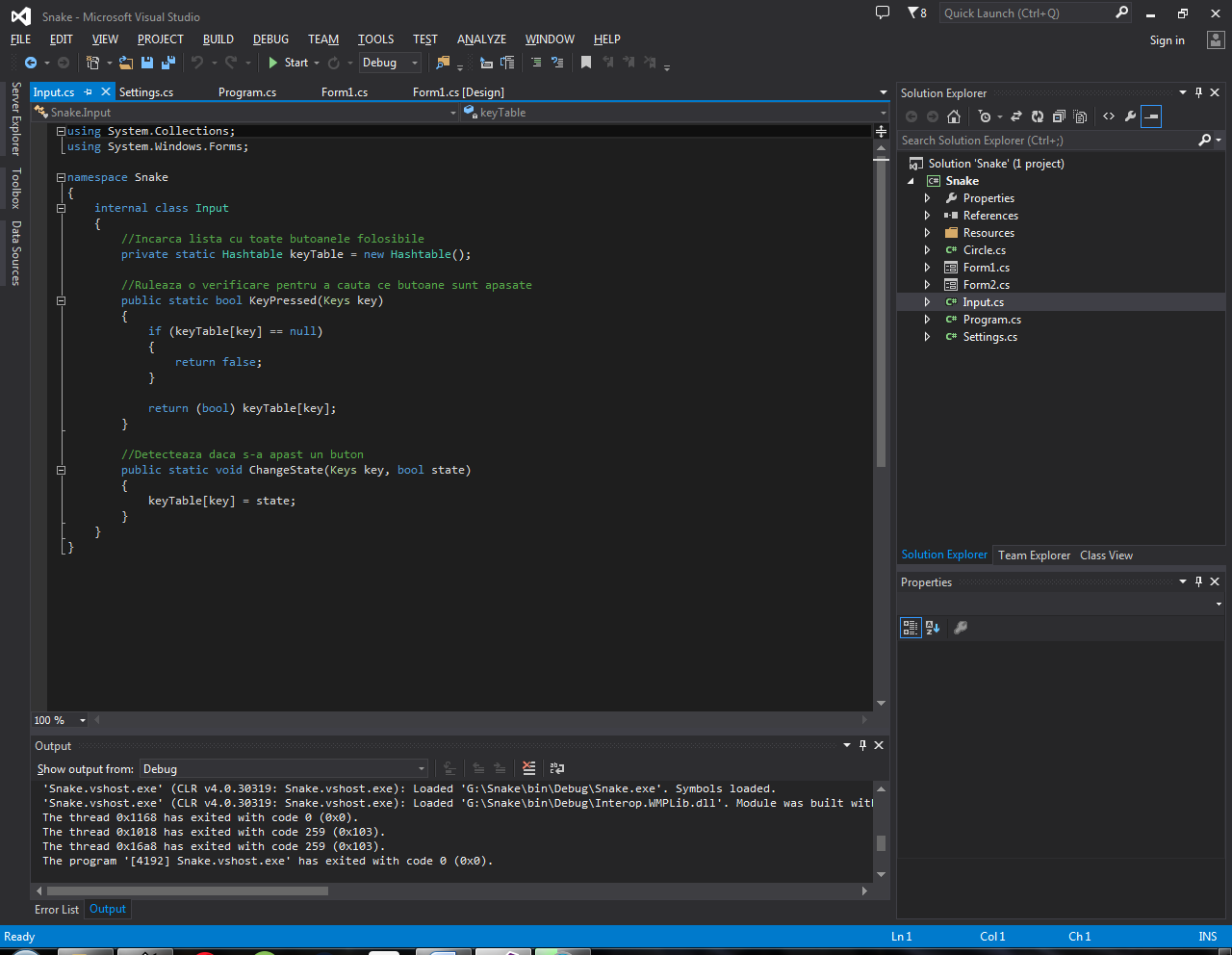
12.Aici se va afisa la sfarsitul rundei punctele obtinute si alte informatii

Clasa Circle



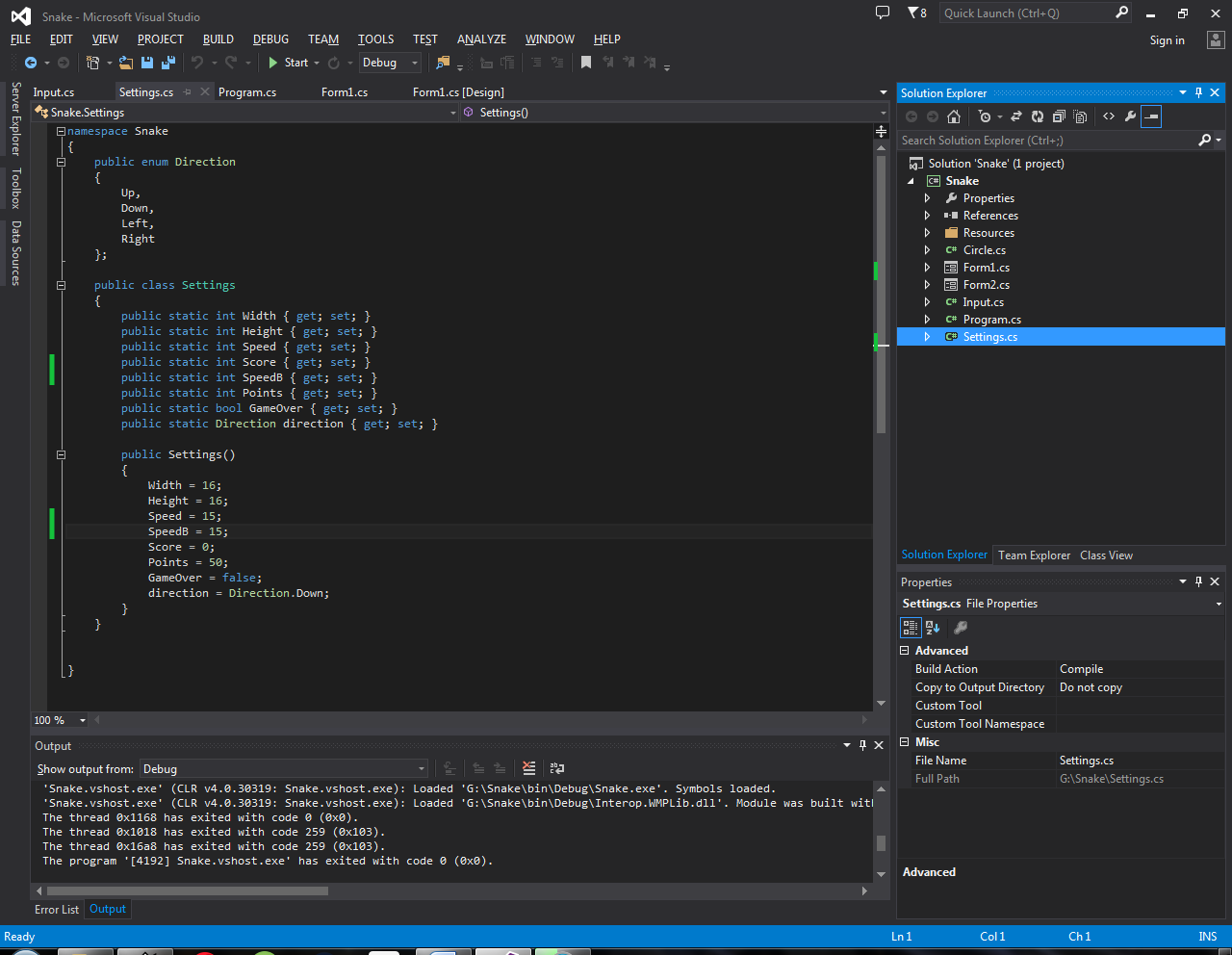
Clasa care reprezinta corpurile din joc,mai exact jucatorul si mancarea pe care o va colecta.In aceasta clasa vom avea variabilele ce vor servi la localizarea pe plansa de joc a obiectelor ,coordonatele X si Y.

Clasa Input



In aceata clasa se vor determina daca sunt apasate butoane pe tastatura si ce butoane sunt apasate pentru a putea misca jucatorul pe ecran sau pentru a putea incepe un nou joc in cazul in care runda a luat sfarsit.

Clasa Settings si Enumeratia Directions

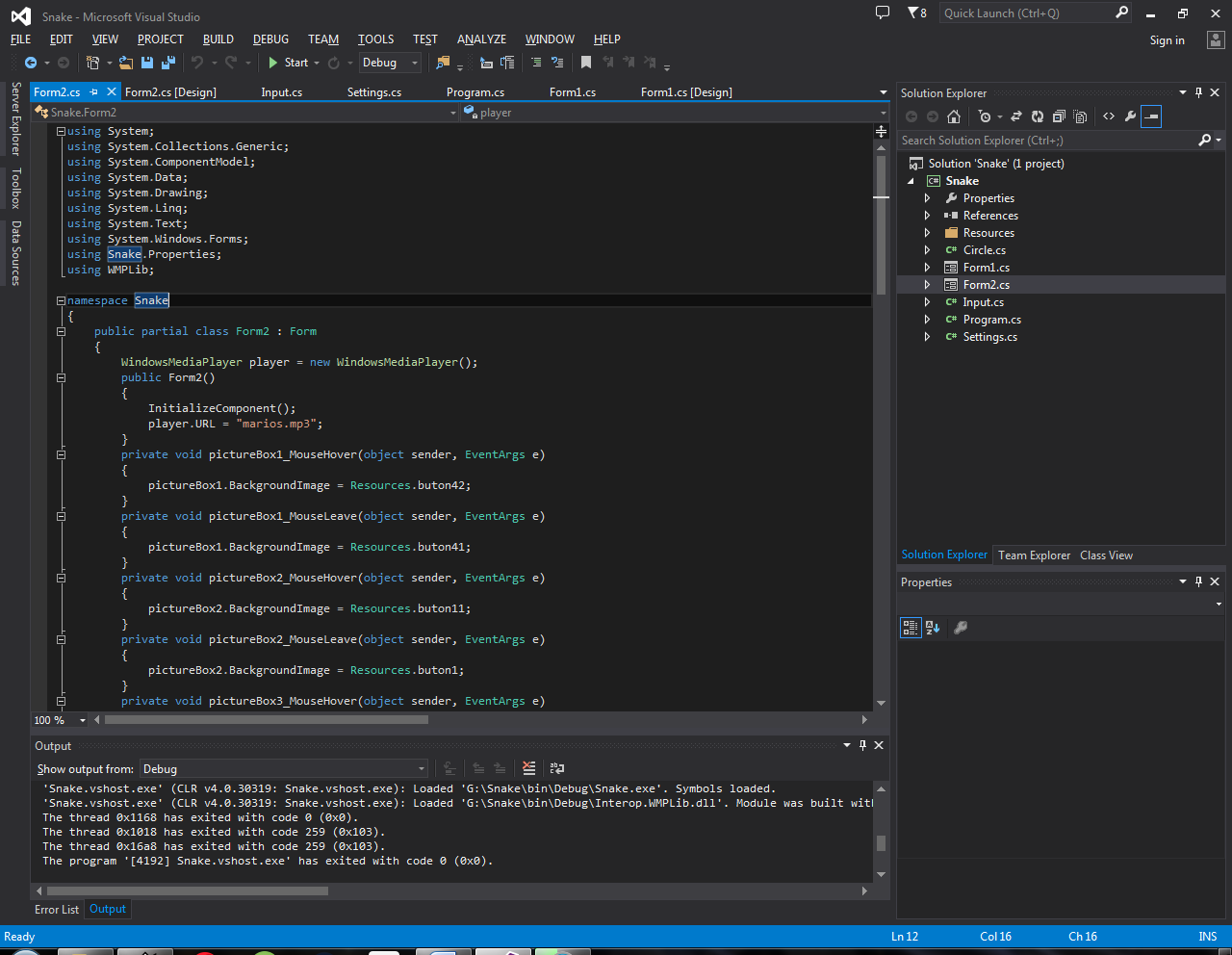


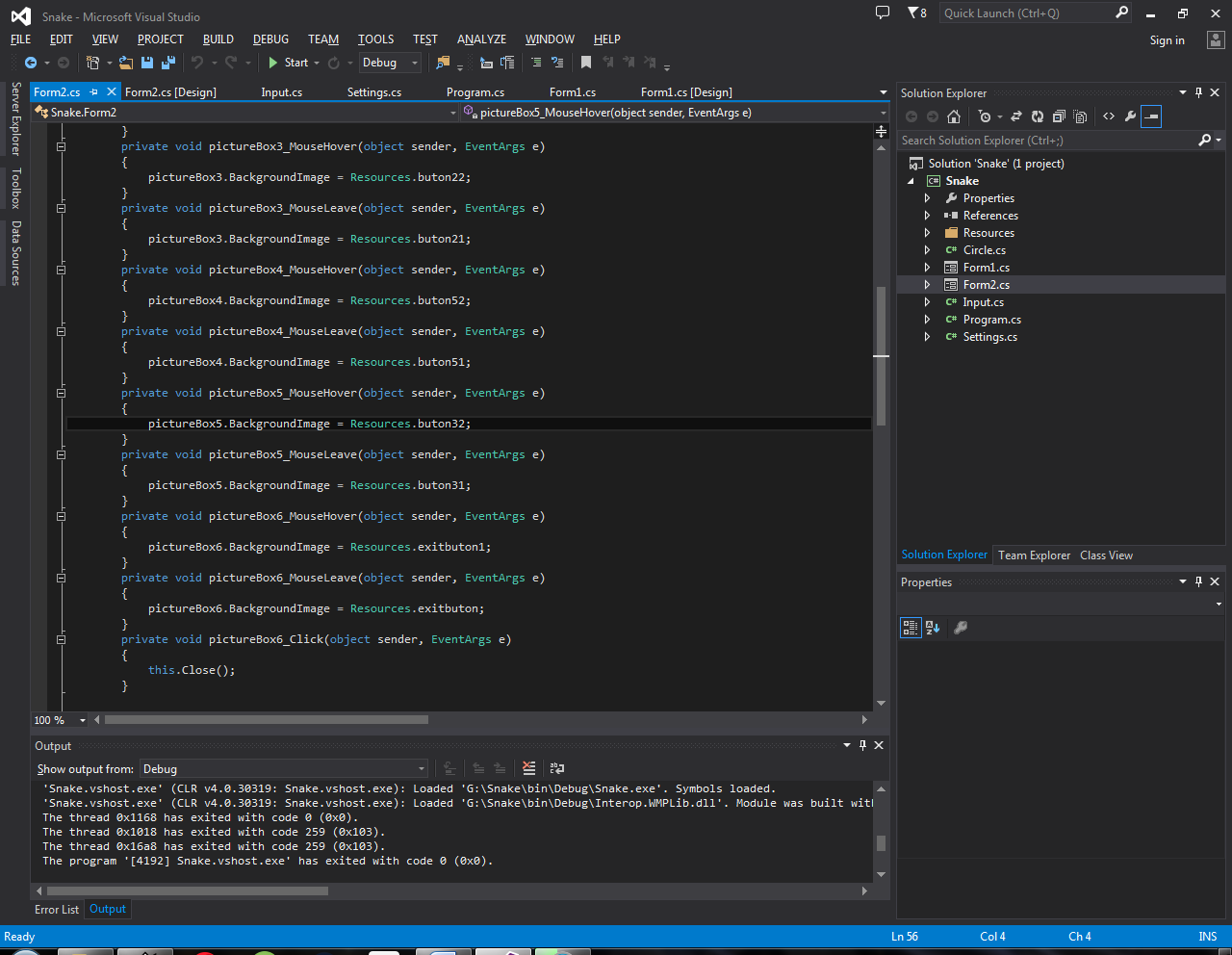
In aceasta clasa avem clasa ce va desemna setarile caracterului si ale jocului precum viteza de miscare,inaltimea si latimea caracterului,scorul,numarul de puncte obtinute cat si enumeratia directiilor de miscare ale personajului(sus,jos,stanga,dreapta).

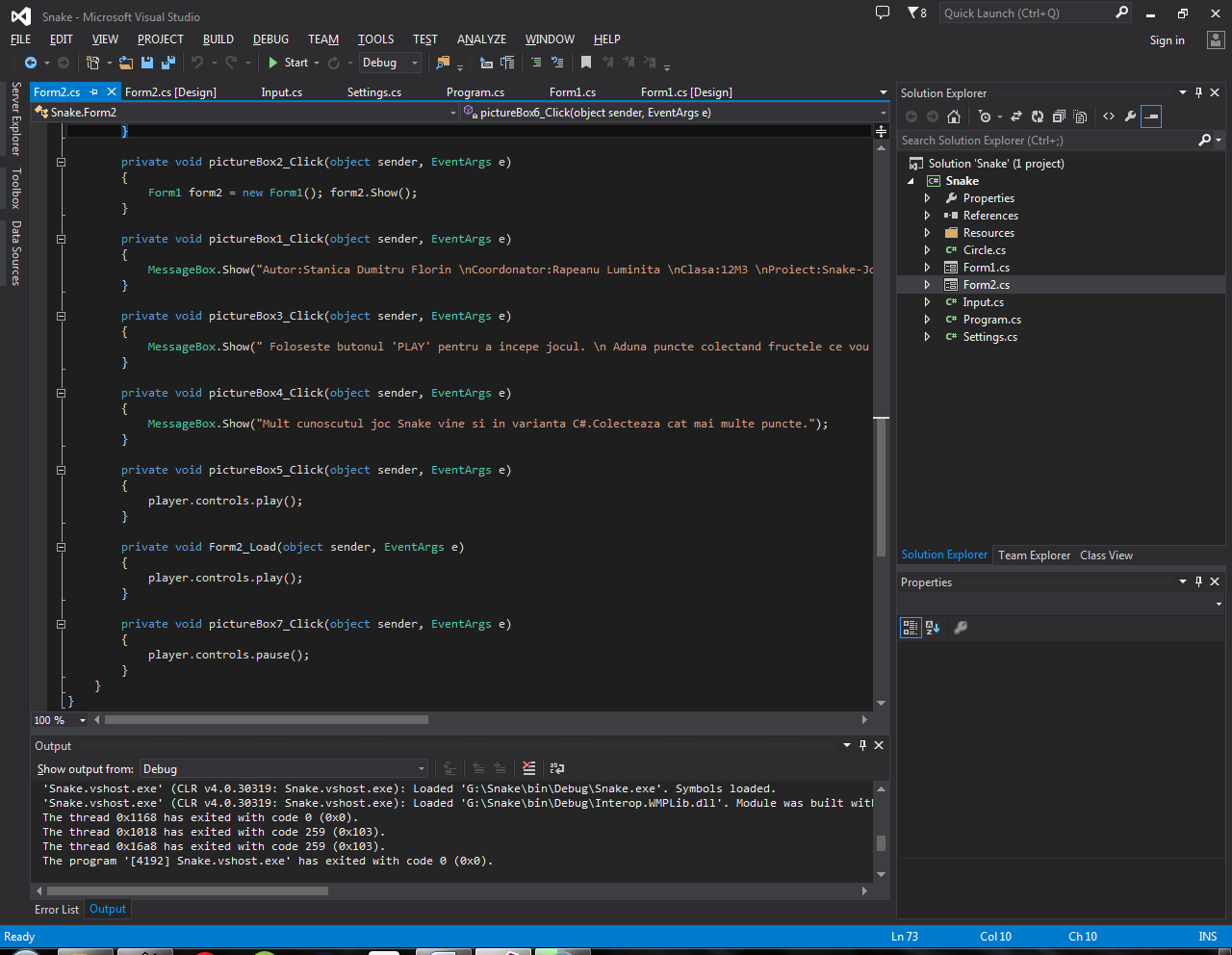
De asemenea avem variabilele setate pentru inceperea unui nou joc pentru variabilele enumerate mai sus.

Codul din spatele interfatei

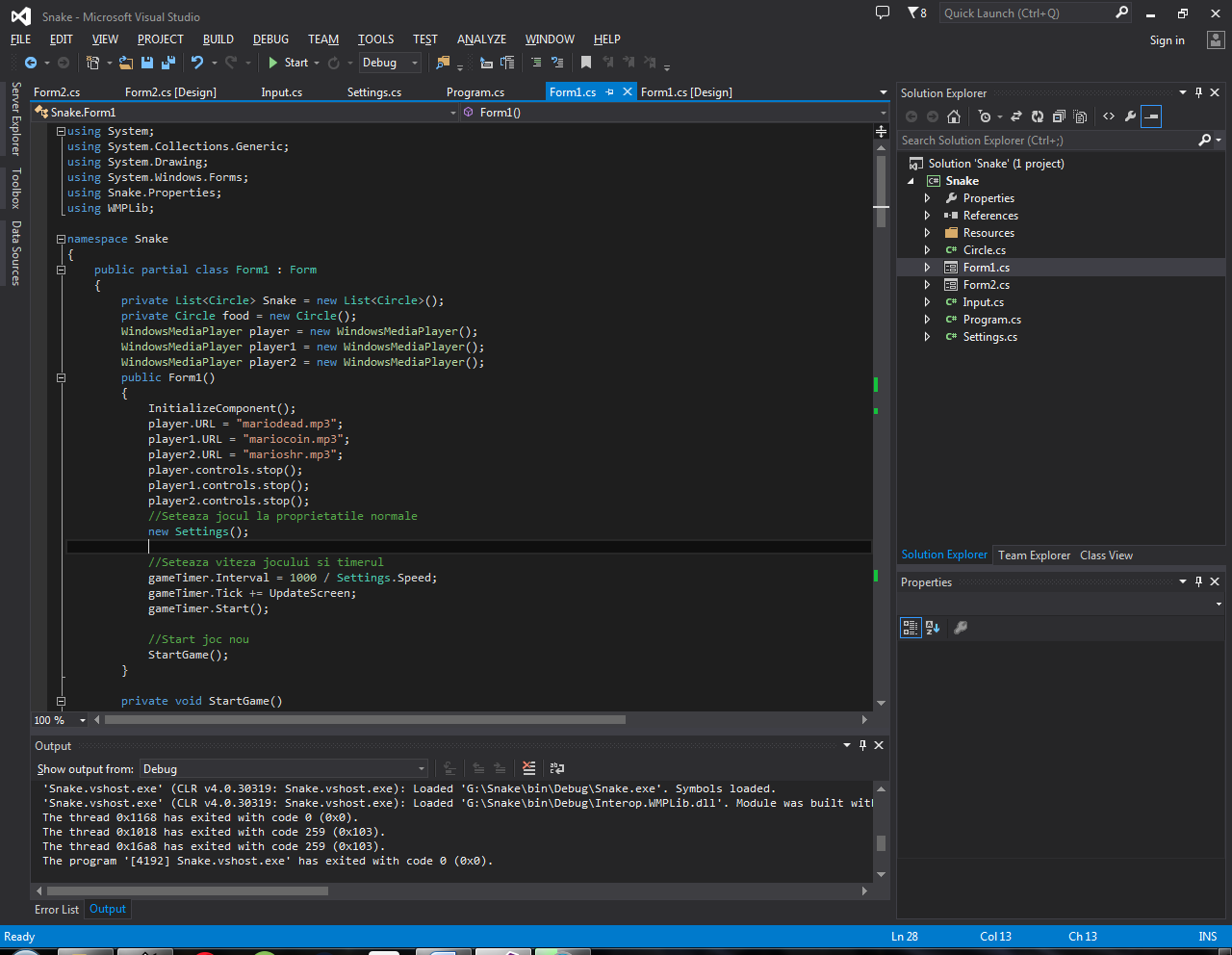
Aici se afla codul ce perimite “luminarea”butoanelor cand mouse-ul se afla peste.



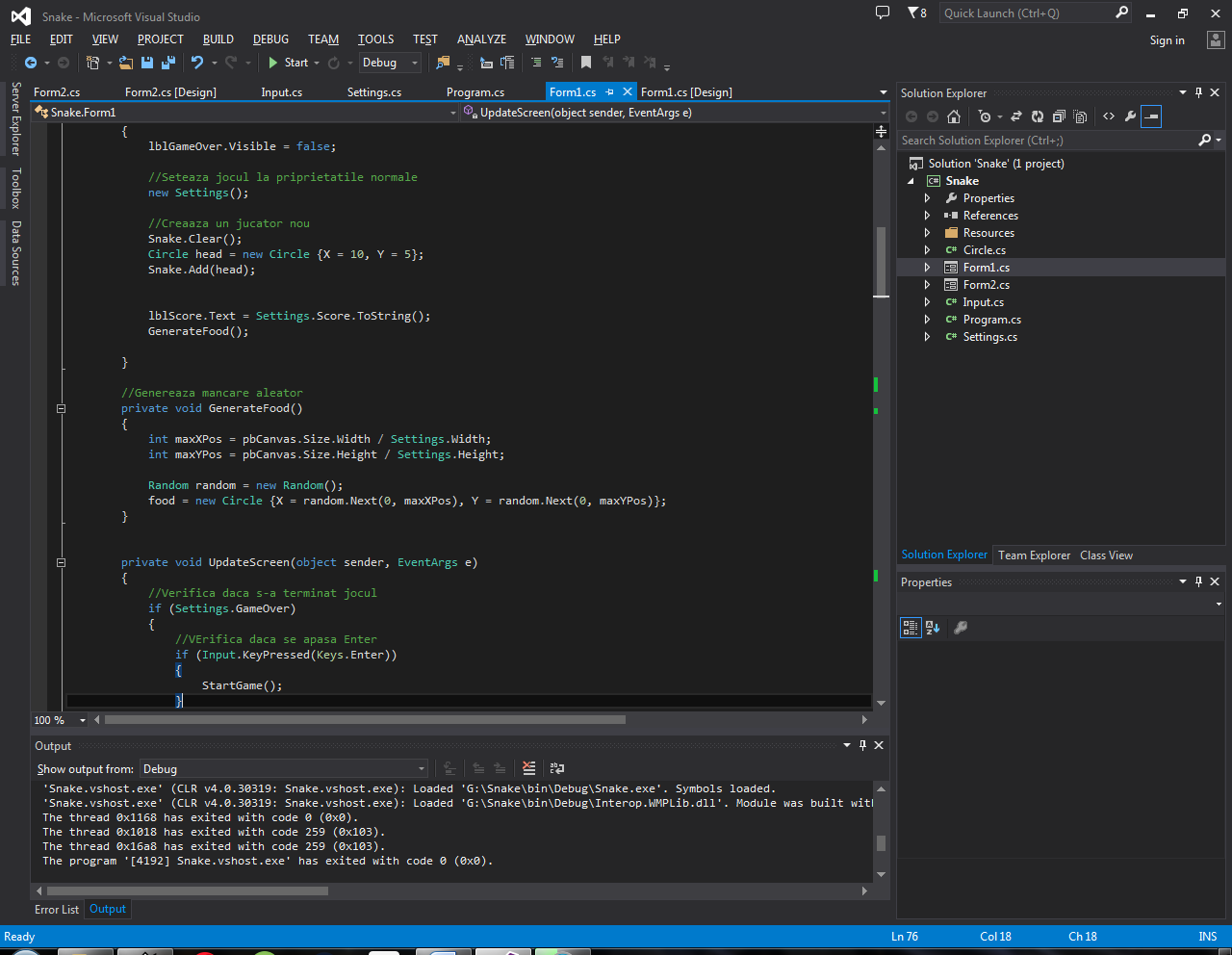




Codul pentru deschiderea MessageBox-urilor informationale.

Codul ce ruleaza in spatele jocului

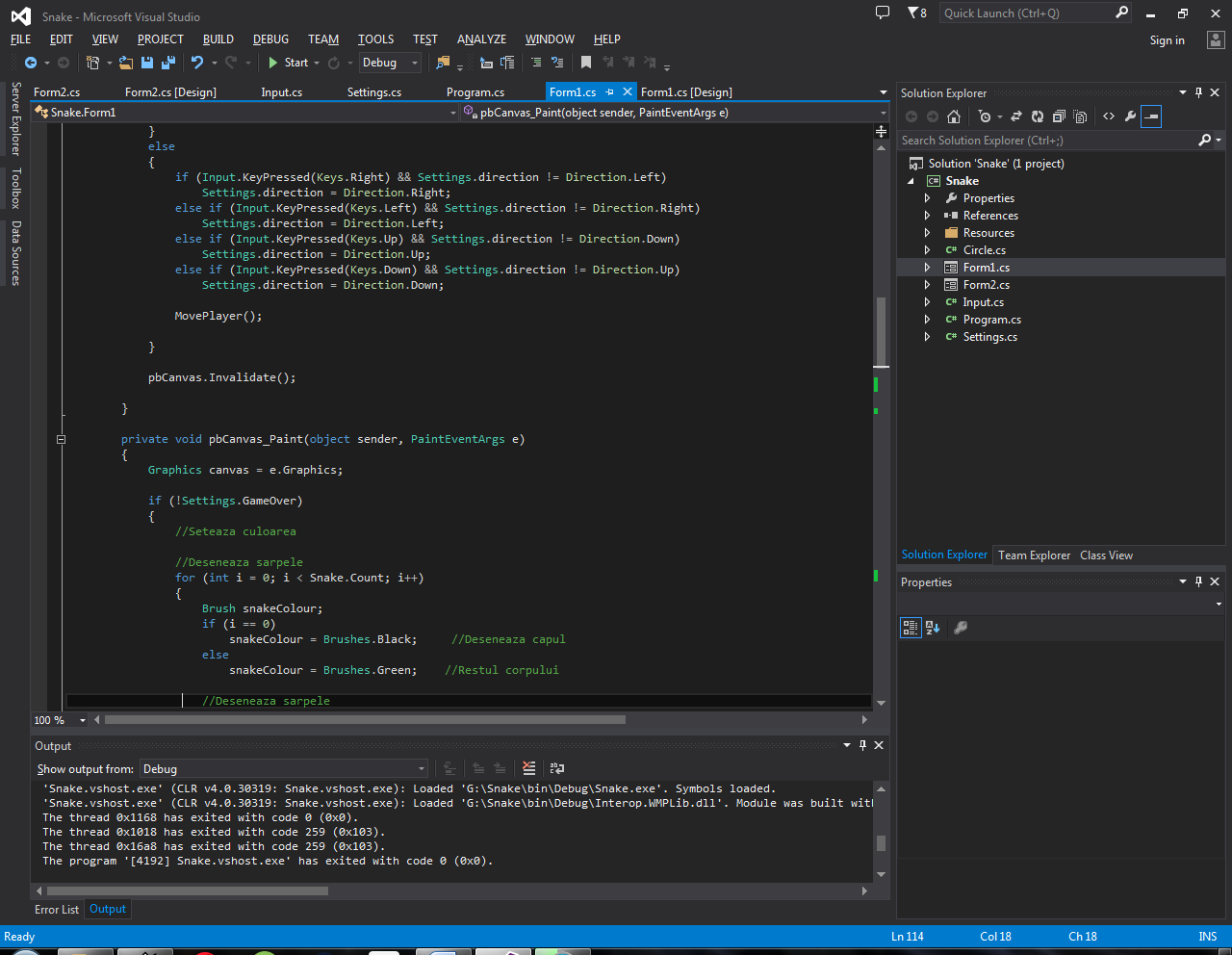
Avem de fata inceputul jocului in sine si incarcarea muzicii de fundal in format .mp3,apelarea functiilor ce vor seta jocul si inceperea jocului.



De fata avem inceputul functiilor:

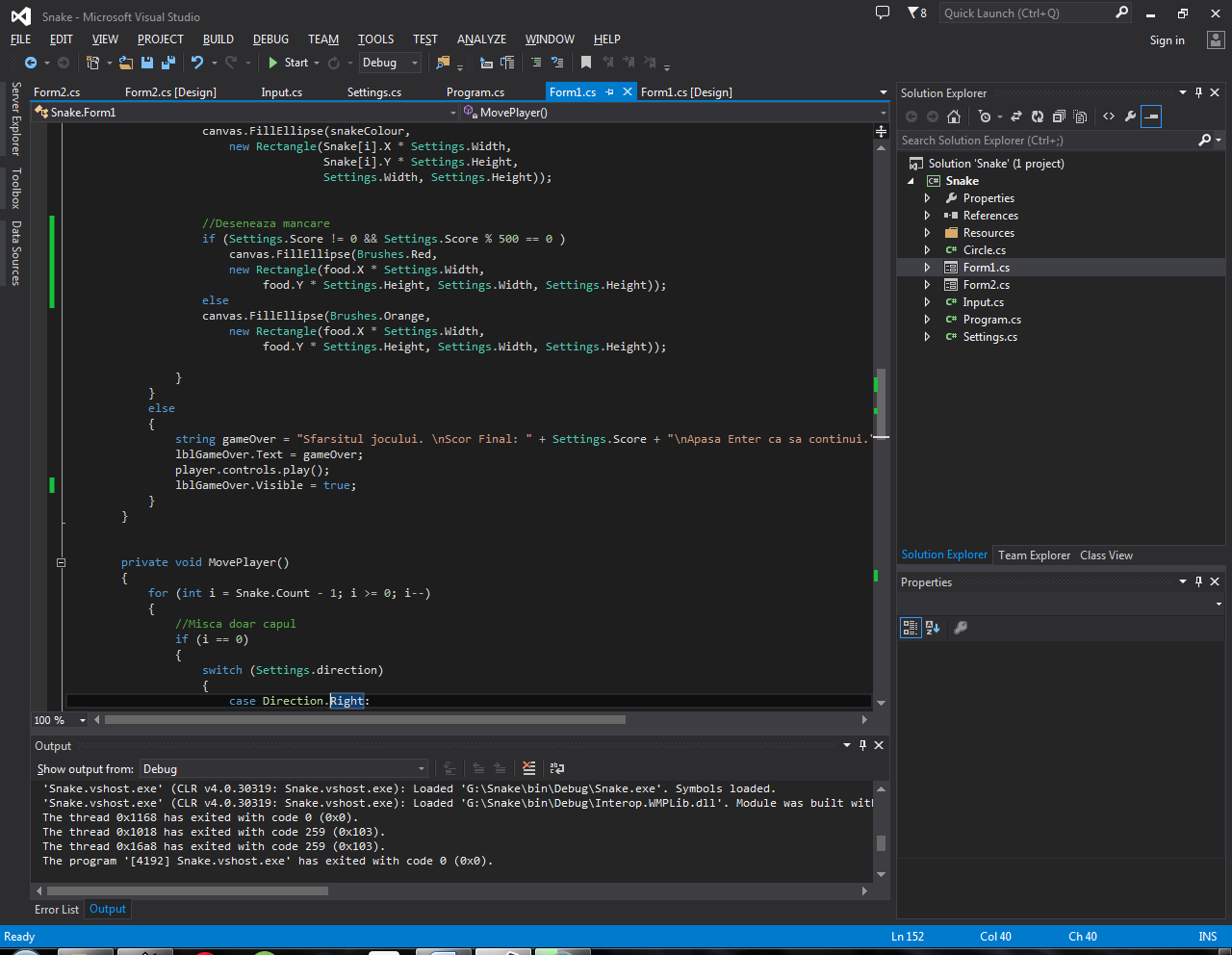
-prima functie creeaza noul jucator,setarile la default si porneste generarea mancarii

-a doua functie genereaza mancare folosind parametrii aleatori,ce nu vor depasi dimensiunile plasei de joc

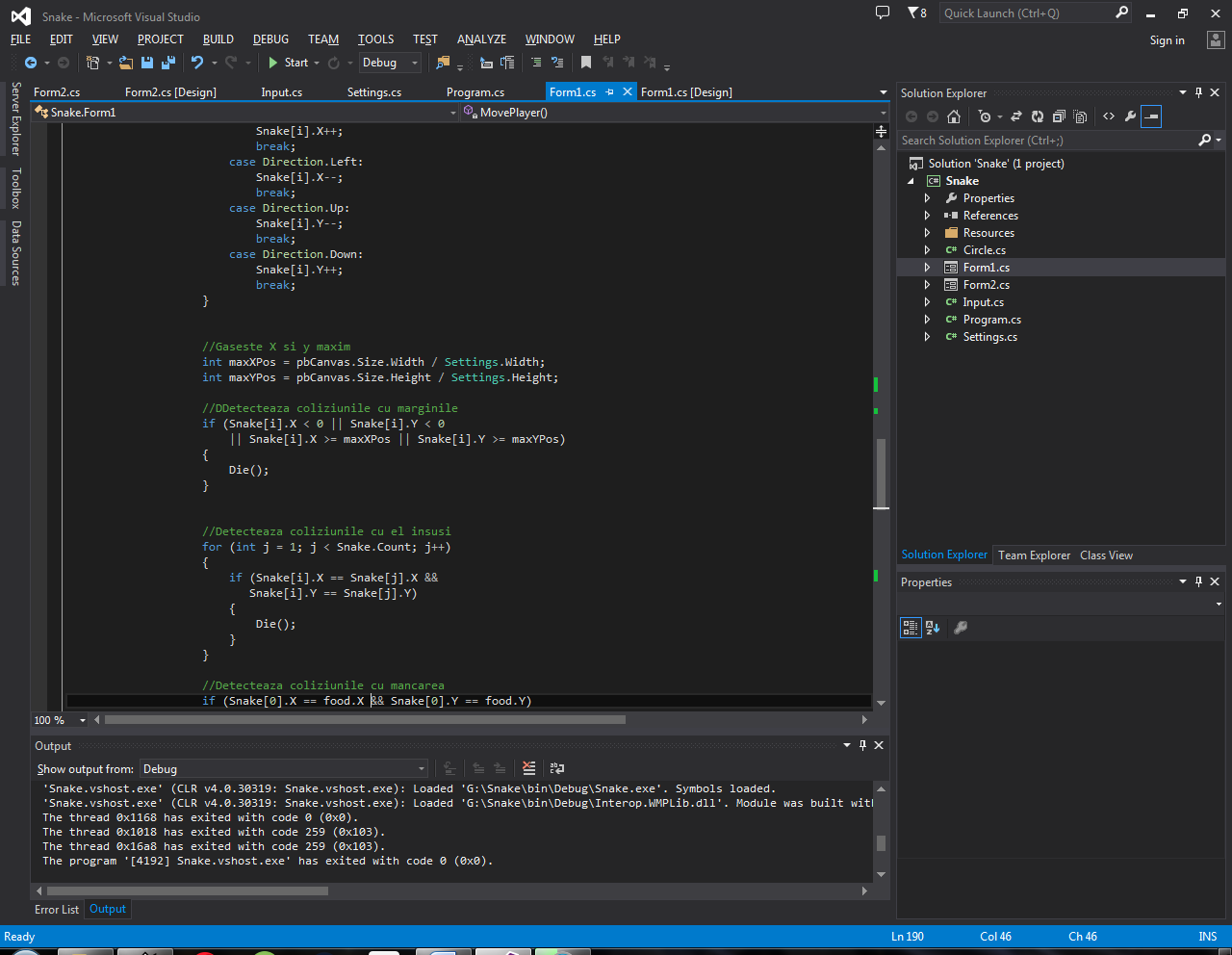


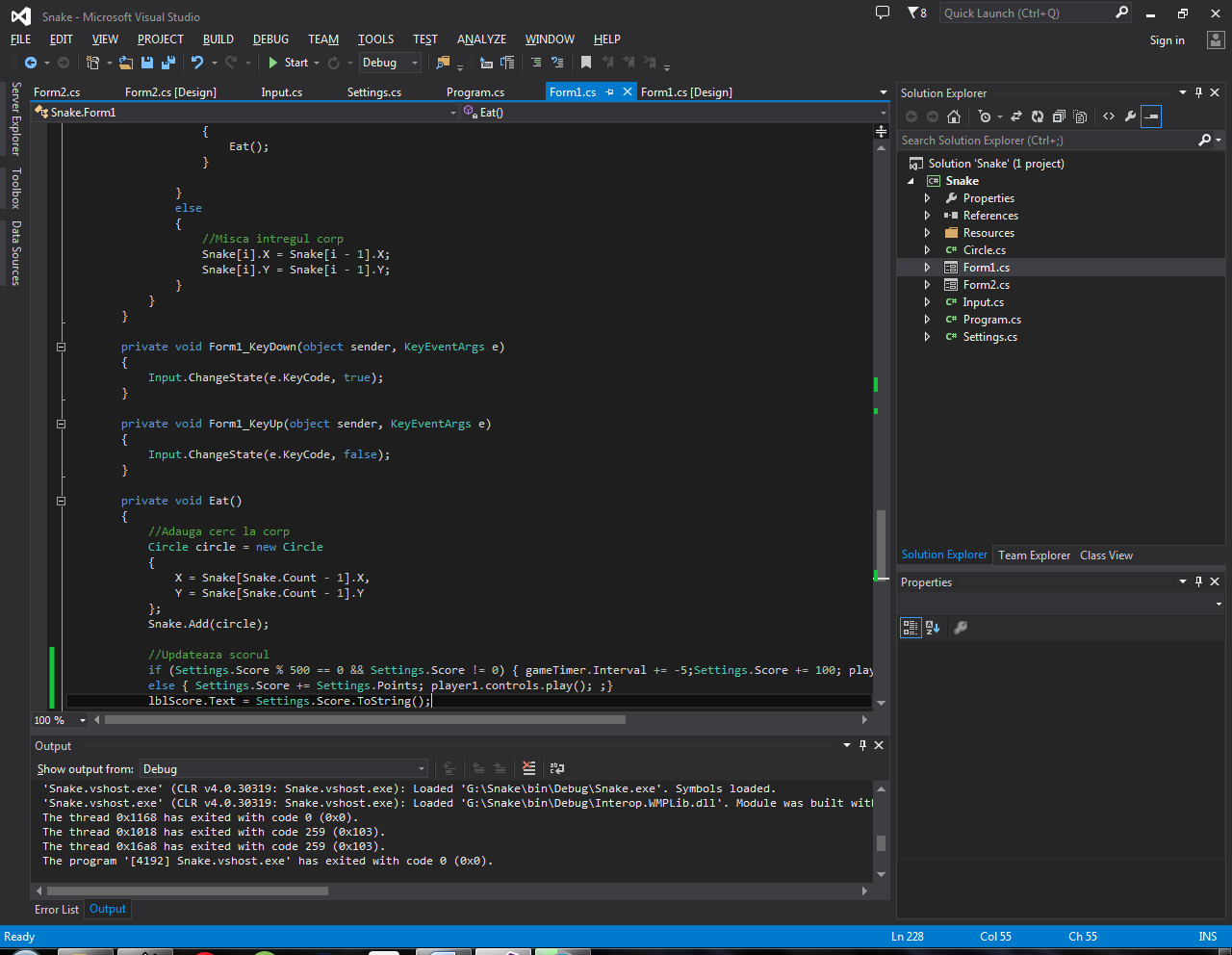
-fuctia de updatare a ecranului de joc dupa parametrii de directie

-urmatoarea functie prezinta desenarea jucatorului,corpului jucatorului si a mancarii,iar in cazul sfarsitului rundei,prezinta mesajul de sfarsit si puntele stranse in ruda respectiva



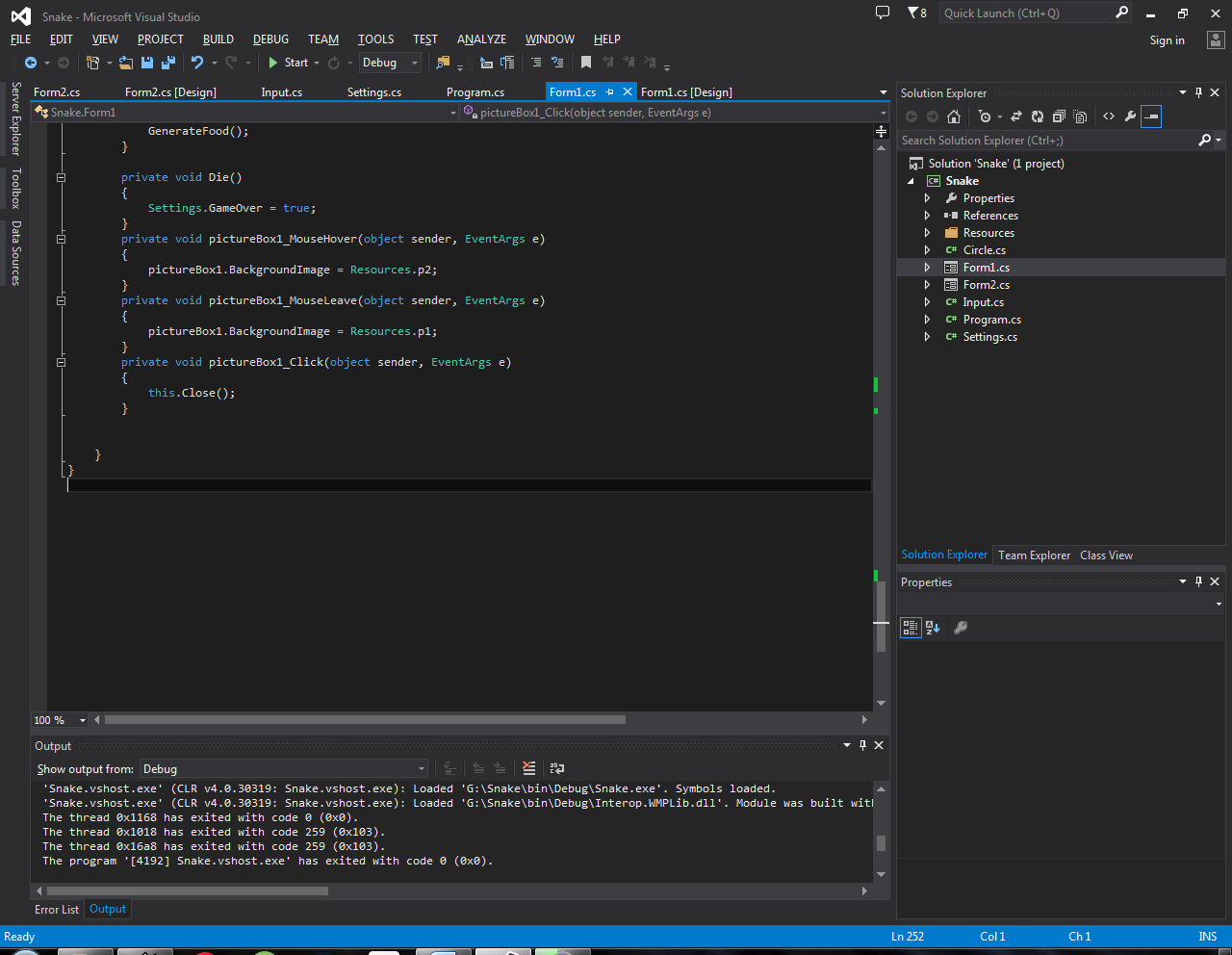
-urmeaza generarea miscarilor jucatorului si verificarea coliziunilor cu mariginile,el insusi sau mancarea





-functiea eat incrementeaza scorul in cazul coliziunilor cu mancarea si creste marimea corpului jucatorului

-functia die marcheaza sfarsitul rundei in cazul ciocniri cu celelalte obiecte



**Metoda folosita si**

**Posibile inbunatatiri**

Metoda progaramrii dinamice a fost folosita in acest proiect pentru a misca si creste corpul.

Posibile inbunatatiri sunt:

-adaugarea de noi planse cu mai multe obstacole

-creearea unei functii speciale,deoarece exista sansa ca mancarea sa fie create deasupra corpului jucatorului,asta insa nu afecteaza jucatorul in nici un fel

-incrementarea de noi obiecte din care se pot obtine puncte si ce pot modifica parametrii jocului

Metoda:

private void pbCanvas\_Paint(object sender, PaintEventArgs e)

{

Graphics canvas = e.Graphics;

if (!Settings.GameOver)

{

//Seteaza culoarea

//Deseneaza sarpele

for (int i = 0; i < Snake.Count; i++)

{

Brush snakeColour;

if (i == 0)

snakeColour = Brushes.Black; //Deseneaza capul

else

snakeColour = Brushes.Green; //Restul corpului

//Deseneaza sarpele

canvas.FillEllipse(snakeColour,

new Rectangle(Snake[i].X \* Settings.Width,

Snake[i].Y \* Settings.Height,

Settings.Width, Settings.Height));

//Deseneaza mancare

if (Settings.Score != 0 && Settings.Score % 500 == 0 )

canvas.FillEllipse(Brushes.Red,

new Rectangle(food.X \* Settings.Width,

food.Y \* Settings.Height, Settings.Width, Settings.Height));

else

canvas.FillEllipse(Brushes.Orange,

new Rectangle(food.X \* Settings.Width,

food.Y \* Settings.Height, Settings.Width, Settings.Height));

}

}

else

{

string gameOver = "Sfarsitul jocului. \nScor Final: " + Settings.Score + "\nApasa Enter ca sa continui.";

lblGameOver.Text = gameOver;

player.controls.play();

lblGameOver.Visible = true;

}

}

private void MovePlayer()

{

for (int i = Snake.Count - 1; i >= 0; i--)

{

//Misca doar capul

if (i == 0)

{

switch (Settings.direction)

{

case Direction.Right:

Snake[i].X++;

break;

case Direction.Left:

Snake[i].X--;

break;

case Direction.Up:

Snake[i].Y--;

break;

case Direction.Down:

Snake[i].Y++;

break;

}

//Gaseste X si y maxim

int maxXPos = pbCanvas.Size.Width / Settings.Width;

int maxYPos = pbCanvas.Size.Height / Settings.Height;

//DDetecteaza coliziunile cu marginile

if (Snake[i].X < 0 || Snake[i].Y < 0

|| Snake[i].X >= maxXPos || Snake[i].Y >= maxYPos)

{

Die();

}

//Detecteaza coliziunile cu el insusi

for (int j = 1; j < Snake.Count; j++)

{

if (Snake[i].X == Snake[j].X &&

Snake[i].Y == Snake[j].Y)

{

Die();

}

}

//Detecteaza coliziunile cu mancarea

if (Snake[0].X == food.X && Snake[0].Y == food.Y)

{

Eat();

}

}

else

{

//Misca intregul corp

Snake[i].X = Snake[i - 1].X;

Snake[i].Y = Snake[i - 1].Y;

}

}

}