SNACK GAME IN C

Toshika Verma
0801CS21095
B.tech 2nd year
SGSITS INDORE

November 20, 2022

1 objective of project

In this game, the main objective of the player is to catch the maximum number of fruits without hitting the wall or itself. Creating a snake game can be taken as a challenge on learning on phython language

2 These are the following function are used in project:

2.0.1 setup

This setup function used for the set boundrey or outline of snack game.

2.0.2 draw

This draw function is used for basically to draw the head and food of snack.

2.0.3 input

THIS input function is for the momemt of snack . we take input from thwe user in form key a,b,c,d.

2.0.4 makelogic

We press the key a then moment in upward y axis ,key b for downdard direction,key c for the right direction,key d for the left direction.

2.0.5 highest

highest function is used for basically we want to print highest score'

2.0.6 name

name function is for print the name of snack.

2.0.7 colour

we give the colour of snack.

2.0.8 winner

who will reach the highest score get the gold model.

2.0.9 tail

when snack eat the food there tail should be increase then because we use the function tail.

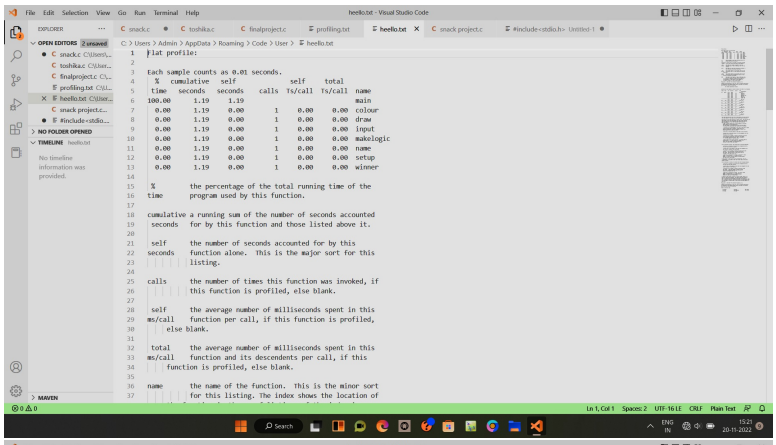
2.0.10 food

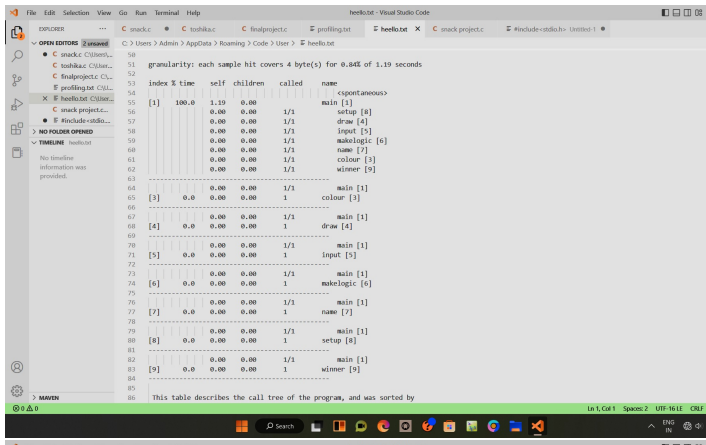
when snack the food but we genreate the new food so we the function food.

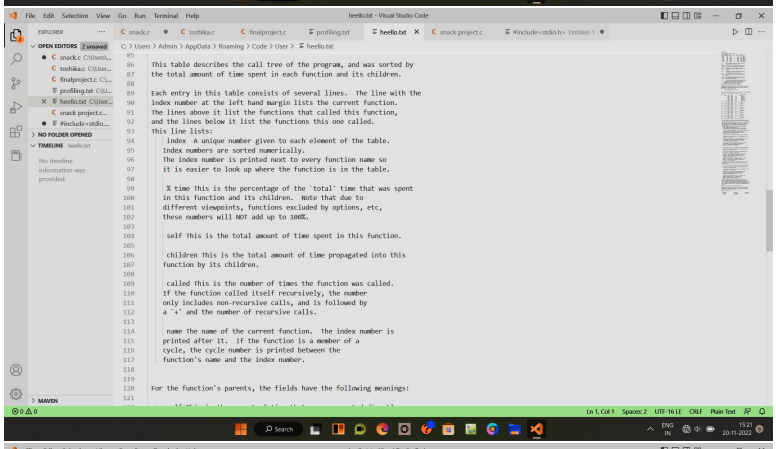
3 descriptionn

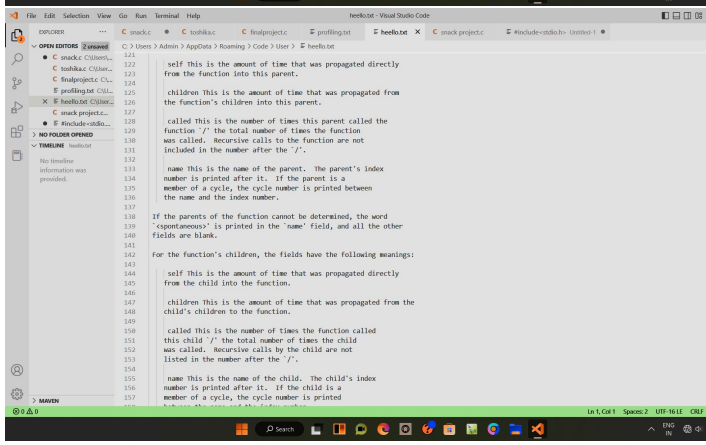
The player controls a long, thin creature, resembling a snake, which roams around on a bordered plane, picking up food (or some other item), trying to avoid hitting its own tail or the edges of the playing area. Each time the snake eats a piece of food, its tail grows longer, making the game increasingly difficult

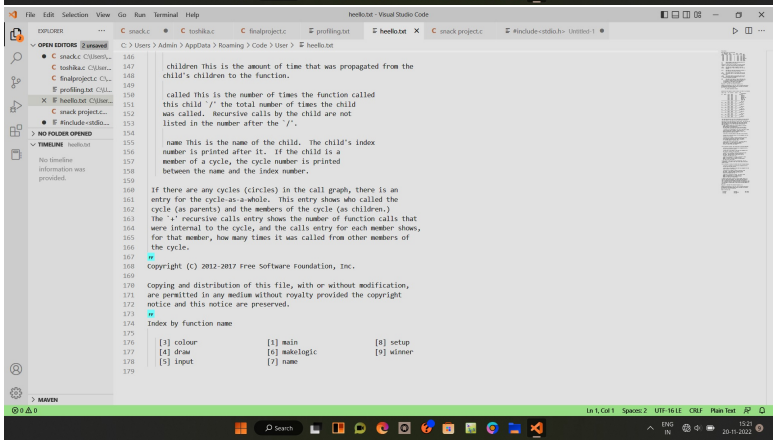
4 profilier report



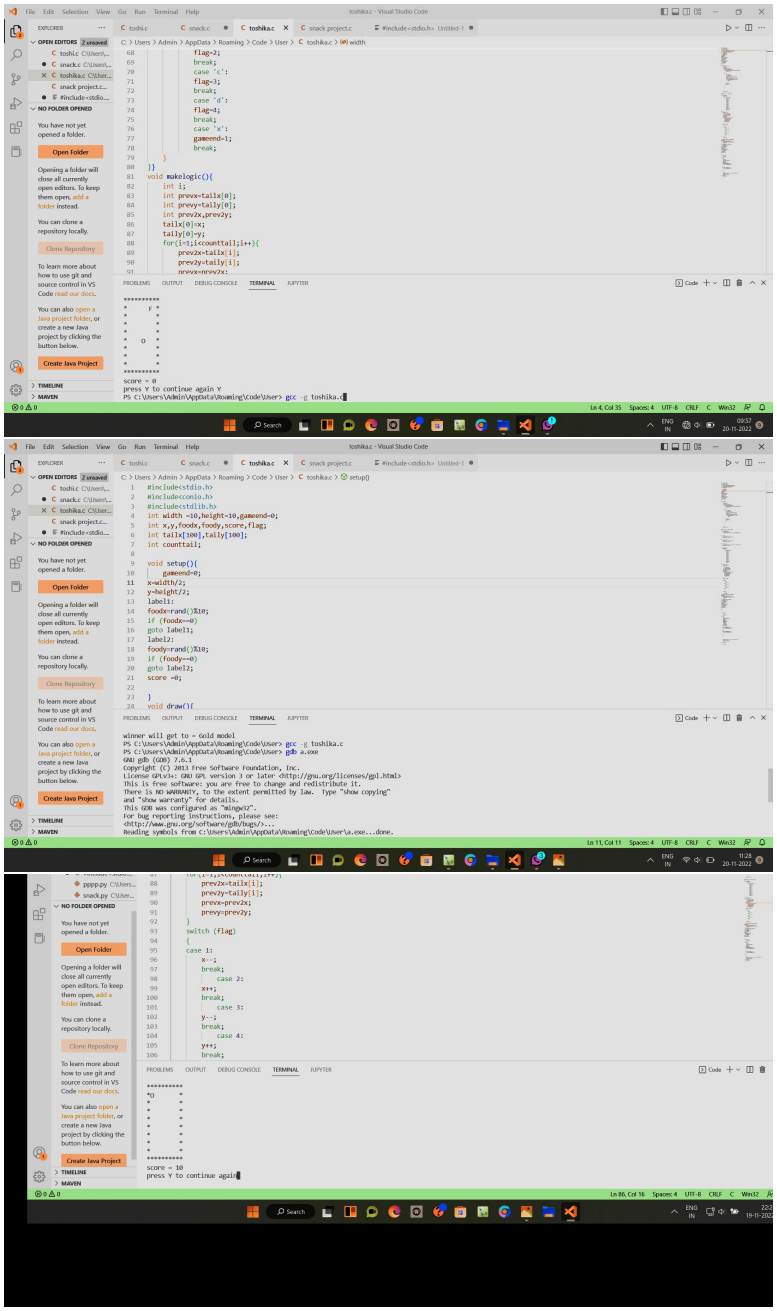








5 GBD activities



6 code with comment in c

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
int width =10,height=10,gameend=0;
\\declare the name of vairble
int x,y,foodx,foody,score,flag;
\\declare the name of vairble
int tailx[100],taily[100];
\\declare the name of vairble
int counttail;
\\declare the name of vairble

void setup()
\\we want setup the boundary of snack
{
        gameend=0;
        x=width/2;
        y=height/2;
        label1:
        foodx=rand()%10;
        if (foodx==0)
```

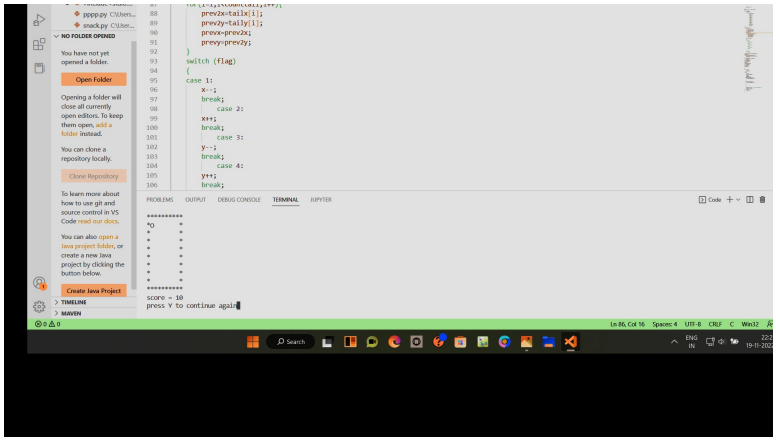
```
        goto label1;
        label2:
        foody=rand()%10;
        if (foody==0)
        goto label2;
        score =0;

}
void draw()
\\to draw the boundary of snack
{

        int i,j,k;
        system("cls");
        for(i=0;i<width;i++)
        {
        for(j=0;j<height;j++)
        {
        if(i==0||i==height-1||j==0||j==width-1)
        {
        printf("*");

        }
        else
        {
        int ch=0;
        if (i==x&&j==y)
        printf("O");
        else if(i==foodx&&j==foody)
        printf("F");
        else{for(k=0;k<counttail;k++)
        {
        if(i==tailx[k]&&j==taily[k])
        {
        printf("O");
        ch=1;
        }
        }
        if(ch==0)
        printf(" ");
        }

        }

        }

        printf("\n");
        }
        printf("score = %d\n",score);
  }
    void hightest()
    \\ to set the hightset score
 {
  int highestscore;
  printf("hightestscore = %d\n",highestscore);
 }
  void name ()
  \\ give the name of snack
  {
        char str[]="Name = kobra";
        printf("%s\n",str);

 }
  void colour()
  \\colour of the snack
```

```
{
 char str[]="colour = blue";
 printf("%s\n",str);

}
void winner ()
\\wiiner will get the gold medal
{
      char str[]="winner will get to = Gold model";
      printf("%s\n",str);
}

void input()
\\ give the input in the form key a,b,c,d
{
      if(kbhit())
      {
         switch(getch())
      {

     case 'a':
     flag =1;
     break;
     case 'b':
     flag=2;
     break;
     case 'c':
     flag=3;
     break;
     case 'd':
     flag=4;
     break;
     case 'x':
     gameend=1;
     break;
     }
     }
     }
       void makelogic()
       \\ we define the logic of snack game using pressing the key

       {

              int i;
              void tail()
              \\increase the size of the snack
          {
              int prevx=tailx[0];
              int prevy=taily[0];
              int prev2x,prev2y;
              tailx[0]=x;
              taily[0]=y;
              for(i=1;i<counttail;i++)
              {
              prev2x=tailx[i];
              prev2y=taily[i];
              prevx=prev2x;
              prevy=prev2y;
              printf("F");

              }
              }
              switch (flag)
        {
              case 1:
```

```
                x--;
                break;
                case 2:
                x++;
                break;
                case 3:
                y--;
                break;
                case 4:
                y++;
                break;


                default:
                break;
        }
        void food()
        \\for genrate the food
                {
        if(x<0||x>width||y<0||y>height)
        gameend=1;
        for(i=0;i<counttail;i++)
        {
        if(x==tailx[i]&&y==taily[y])
        gameend=1;
        }
        if(x==foodx && y==foody)
   {
        label3:
        foodx=rand()%10;
        if (foodx==0)
        goto label4;
        label4:
        foody=rand()%10;
        if (foody==0)
        goto label4;
        score+=10;
        counttail++;

  }
        }

  }
int main()
        {

        int m,n;
        char c;
        label5:

        setup();
        while(!gameend)
        {
        draw();
        input();
        makelogic();
        hightest();
        name();
        colour();
        winner();
        void tail();
        void food();

        for(m=0;m<100000;m++)
        {
```

```
for(n=0;n<100000;n++)
{

}
}
for(m=0;m<100000;m++)
{
for(n=0;n<100000;n++)
{

}
}

}

printf("\npress Y to continue again");
scanf("%c",&c);
if(c=='y'||c=='Y')

goto label5;

return 0;
}
```

7 code output



8 code in phython

```
import turtle
import random
import time

delay = 0.1
score = 0
highestscore = 0
#snack bodies
bodies =[]
# getting screen of snack
s = turtle.Screen()
s.title("snack game")
s.bgcolor("green")
s.setup(width=800,height=800)
#create snack head
head = turtle.Turtle()
head.speed(0)
head.shape("square")
head.color("black")
head.fillcolor("red")
head.penup()
head.goto(0,0)
head.direction= "stop"
#snack food
food = turtle.Turtle()
food.speed(0)
food.shape("circle")
food.color("pink")
food.fillcolor("black")
food.penup()
food.ht()
food.goto(0,200)
food.st()
#score board
sb = turtle.Turtle()
sb.shape("square")
sb.fillcolor("orange")
sb.penup()
sb.ht()
sb.goto(-300,-300)
#sb.write("Score :  0     |                                                    Highestscore :
0")
def moveup():
if head.direction!="down":
head.direction = "up"
def movedown():
if head.direction!="up":
head.direction ="down"
```

```
def moveleft():
if head.direction!="right":
head.direction="left"
def moveright():
if head.direction!="left":
head.direction="right"
def movestop():
head.direction="stop"
def move():
if head.directiona=="up":
y=head.ycor()
head.sety(y+20)
if head.directiona=="down":
y=head.ycor()
head.sety(y-20)
if head.directiona=="left":
x=head.xcor()
head.setx(x+20)
if head.directiona=="right":
x=head.xcor()
head.setx(x-20)
#event handling
s.listen()
s.onkey(moveup,"Up")
s.onkey(movedown, "Down")
s.onkey(moveleft, "Left")
s.onkey(moveright,"Right")
s.onkey(movestop,"space")
#main loop
while True :
s.update()
#this is to update the screen
#check collisions with border
if head.xcor()>100:
head.setx(-100)
if head.xcor()>-100:
head.setx(100)
if head.xcor()>100:
head.sety(-100)
if head.xcor()>-100:
head.setx(100)
#check collosion with food
if head.distance(food)<10:
#move the food tp new random place
x = random.randint(-100,100)
y = random.randint(-100,100)
food.goto(x,y)
#increase the length of the snack
body = turtle.Turtle()
body.speed(0)
body.penup()
body.shape("square")
body.color("yellow")
body.fillcolor("black")
bodies.append(body)  #appned new body
#increase the score
score+=10
#change delay
delay = 0.001
#update the highest score
if score > highestscore:
highestscore = score
sb.clear()
sb.write("score :  {} Highestscore :  {}".format(score,highestscore))
#move the sncak bodies
```

```
for index in range (len(bodies)-1,0,-1):
x = bodies[index-1].xcor()
y = bodies[index-1].ycor()
bodies [index].goto(x,y)

if len(bodies)>0:
x=head.xcor()
y=head.ycor()
bodies[0].goto(x,y)
move()

#check collosion with shanck body
for body in bodies:
if body.distance(head)<20:
time.sleep(1)
head.goto(0,0)
head.direction ="stop"

#hide bodies
for body in bodies:
body.ht()
bodies.clear()
score =0
delay =0.1
#update the score
sb.clear
sb.write("score :  {}   highestscore :   {}  ".format(score, highestscore))
time.sleep(delay)
s.mainloop()
```