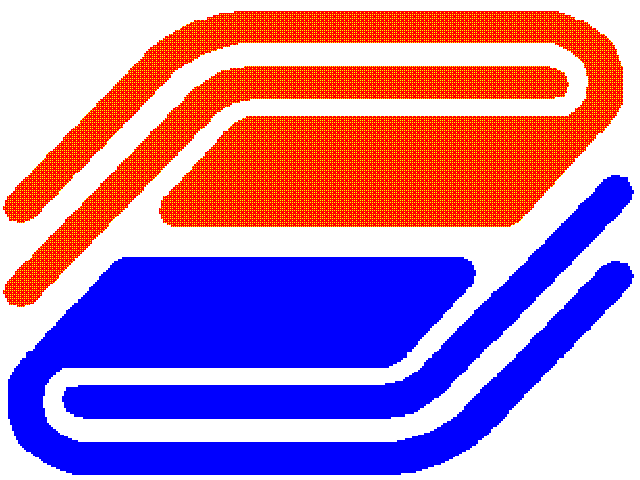
**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ УПРАВЛЕНИЯ»**

**(ГУУ)**



**Отчёт №2**

**«Змейка»**

Выполнил студент института ИИС

группы ПИ 3-2

Теняков Н.А.

Ссылка на GitHub:

https://github.com/KoSiChOk3D/Merdjen-E.

Москва-2021

**Игра «Змейка»**

from turtle import Turtle, Screen

import random

import time

SIZE = 20

class Square:

def \_\_init\_\_(self, x, y):

self.x = x

self.y = y

def drawself(self, turtle):

""" Рисуем черный ящик, оставляя место между кубиками """

turtle.goto(self.x - SIZE // 2 - 1, self.y - SIZE // 2 - 1)

turtle.begin\_fill()

for \_ in range(4):

turtle.forward(SIZE - SIZE // 10)

turtle.left(90)

turtle.end\_fill()

class Food:

def \_\_init\_\_(self, x, y):

self.x = x

self.y = y

self.is\_blinking = True

def changelocation(self):

""" Смена местоположения """

self.x = random.randint(0, SIZE) \* SIZE - 200

self.y = random.randint(0, SIZE) \* SIZE - 200

def drawself(self, turtle):

""" Мигающий квадрат для еды """

if self.is\_blinking:

turtle.goto(self.x - SIZE // 2 - 1, self.y - SIZE // 2 - 1)

turtle.begin\_fill()

for \_ in range(4):

turtle.forward(SIZE - SIZE // 10)

turtle.left(90)

turtle.end\_fill()

def changestate(self):

""" частота мигания """

self.is\_blinking = not self.is\_blinking

class Snake:

def \_\_init\_\_(self):

self.headposition = [SIZE, 0] """ Отслеживание, куда идти """

self.body = [Square(-SIZE, 0), Square(0, 0), Square(SIZE, 0)] """ Список квадратов (тела)"""

self.nextX = 1 """ Координаты, куда нужно идти """

self.nextY = 0

self.crashed = False """ Переменная для столкновений (на будущее)"""

self.nextposition = [self.headposition[0] + SIZE \* self.nextX, self.headposition[1] + SIZE \* self.nextY]

""" Добавление нового квадрата к змейке (подготовка) """

def moveOneStep(self):

if Square(self.nextposition[0], self.nextposition[1]) not in self.body:

""" попытка (неудачная) обнаружения столкновения """

self.body.append(Square(self.nextposition[0], self.nextposition[1]))

""" перемещает голову змеи на следующее место, удаляя хвост """

del self.body[0]

self.headposition[0], self.headposition[1] = self.body[-1].x, self.body[-1].y

""" сбрасывает голову и следующую позицию """

self.nextposition = [self.headposition[0] + SIZE \* self.nextX, self.headposition[1] + SIZE \* self.nextY]

else:

self.crashed = True """ больше столкновений """

def moveup(self): """ Перемещение (стрелки) """

self.nextX, self.nextY = 0, 1

def moveleft(self):

self.nextX, self.nextY = -1, 0

def moveright(self):

self.nextX, self.nextY = 1, 0

def movedown(self):

self.nextX, self.nextY = 0, -1

def eatFood(self):

""" добавляет следующую точку, не удаляя хвост, удлиняя змею на 1 """

self.body.append(Square(self.nextposition[0], self.nextposition[1]))

self.headposition[0], self.headposition[1] = self.body[-1].x, self.body[-1].y

self.nextposition = [self.headposition[0] + SIZE \* self.nextX, self.headposition[1] + SIZE \* self.nextY]

def drawself(self, turtle): """ рисует всю змею при вызове """

for segment in self.body:

segment.drawself(turtle)

class Game:

def \_\_init\_\_(self):

""" у игрового объекта есть экран, черепаха, основная змея и еда """

self.screen = Screen()

self.artist = Turtle(visible=False)

self.artist.up()

self.artist.speed("slowest")

self.snake = Snake()

self.food = Food(100, 0)

self.counter = 0 """ Счетчик (на будущее) """

self.commandpending = False """ Как и это"""

self.screen.tracer(0)

self.screen.listen()

self.screen.onkey(self.snakedown, "Down")

self.screen.onkey(self.snakeup, "Up")

self.screen.onkey(self.snakeleft, "Left")

self.screen.onkey(self.snakeright, "Right")

def nextFrame(self):

self.artist.clear()

if (self.snake.nextposition[0], self.snake.nextposition[1]) == (self.food.x, self.food.y):

self.snake.eatFood()

self.food.changelocation() """ смена локации еды """

else:

self.snake.moveOneStep()

if self.counter == 10:

self.food.changestate() #

self.counter = 0

else:

self.counter += 1

self.food.drawself(self.artist) """ заставляет пищу медленно мигать """

self.snake.drawself(self.artist)

self.screen.update()

self.screen.ontimer(lambda: self.nextFrame(), 100)

def snakeup(self): """ отладка перемещений на стрелочки """

if not self.commandpending:

self.commandpending = True

self.snake.moveup()

self.commandpending = False

def snakedown(self):

if not self.commandpending:

self.commandpending = True

self.snake.movedown()

self.commandpending = False

def snakeleft(self):

if not self.commandpending:

self.commandpending = True

self.snake.moveleft()

self.commandpending = False

def snakeright(self):

if not self.commandpending:

self.commandpending = True

self.snake.moveright()

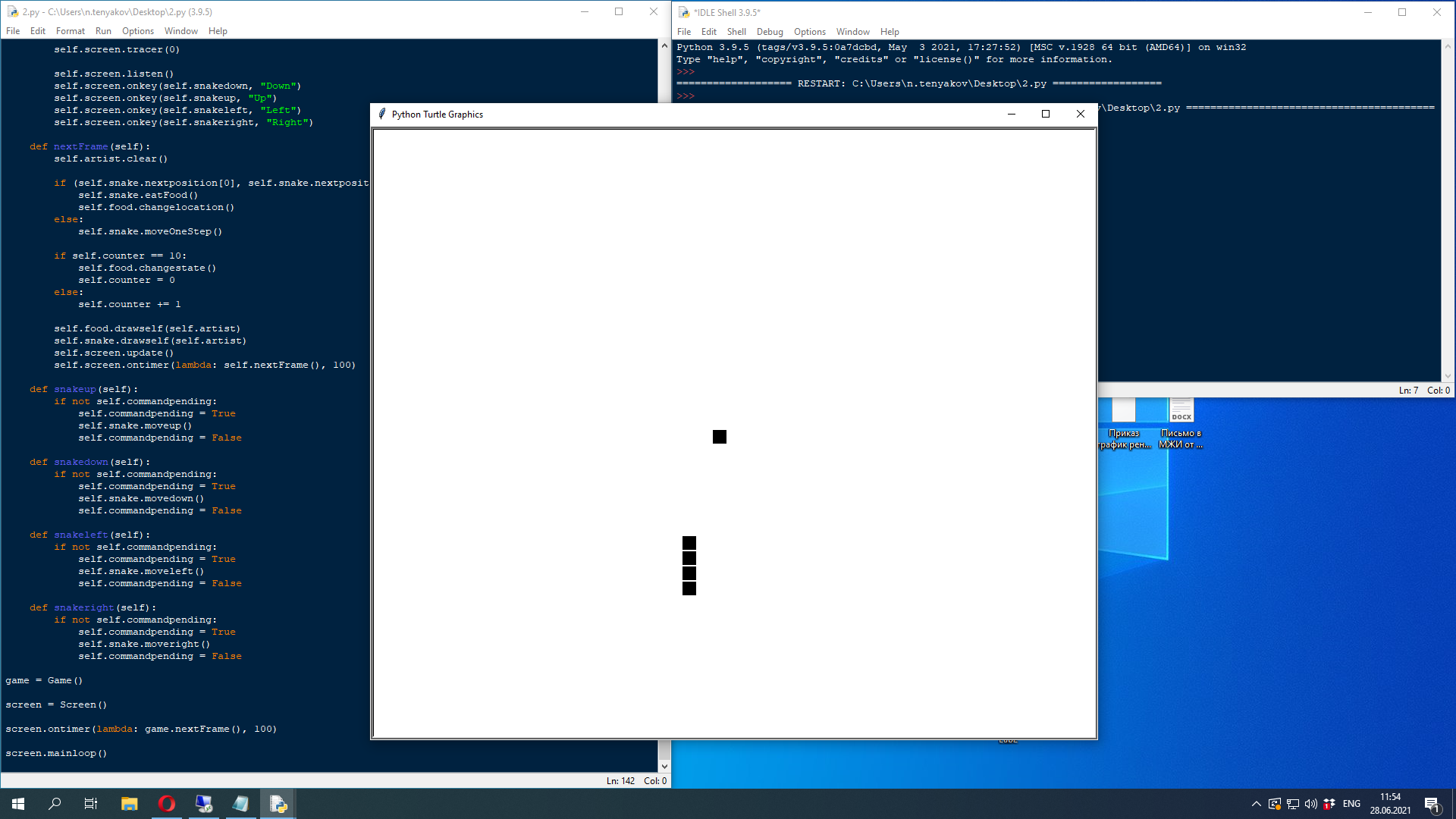
self.commandpending = False

game = Game()

screen = Screen()

screen.ontimer(lambda: game.nextFrame(), 100)

screen.mainloop()



(Рис. 1 – игра «змейка»)