Самостоятельное задание №2

Инвариантная часть

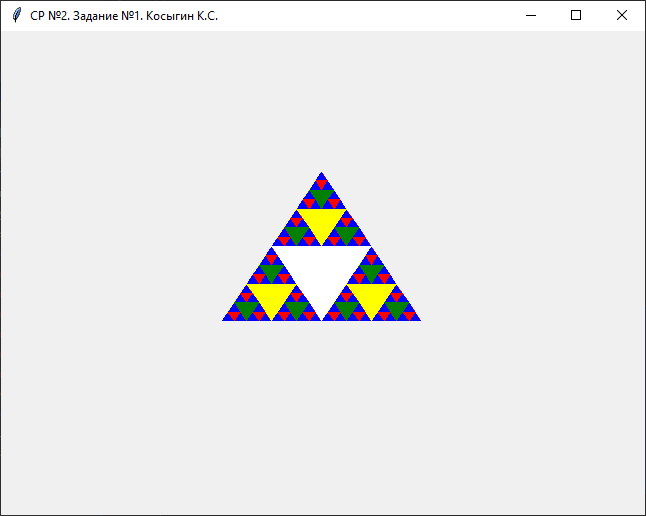
Задание 1: Разработать математическую модель и программную реализацию задачи по вычерчиванию произвольного числа треугольников, располагаемых внутри друг друга.

Математическая модель: каждая следующая вершина внутреннего треугольника – середина вершины, прилегающей к стороне внешнего треугольника.

Код программы:

def sierpinski(points, depth):  
 colors = COLORS  
  
 p1, p2, p3 = points  
 x1, y1 = p1  
 x2, y2 = p2  
 x3, y3 = p3  
 canvas.create\_polygon(x1, y1, x2, y2, x3, y3, fill=colors[depth])  
  
 if depth > 0:  
 depth -= 1  
 points1 = [  
 points[0],  
 get\_mid(points[0], points[1]),  
 get\_mid(points[0], points[2])  
 ]  
 sierpinski(points1, depth)  
  
 points2 = [  
 points[1],  
 get\_mid(points[0], points[1]),  
 get\_mid(points[1], points[2])  
 ]  
 sierpinski(points2, depth)  
  
 points3 = [  
 points[2],  
 get\_mid(points[2], points[1]),  
 get\_mid(points[0], points[2])  
 ]  
 sierpinski(points3, depth)

Протокол работы программы:



Задание 2: Разработать программу для вычерчивания следующей последовательности отрезков прямых линий:

из точки (1.0, 6.0) в точку (1.0, 1.0);

из точки (1.0, 5.8) в точку (1.2, 1.0);

из точки (1.0, 5.6) в точку (1.4, 1.0);

.....

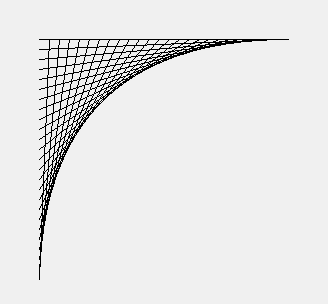
из точки (1.0, 1.0) в точку (6.0, 1.0);

Математическая модель: соответствует поставленной задаче

Код программы:

x1, y1 = (1.0, 6.0)  
x2, y2 = (1.0, 1.0)  
  
canvas.create\_line(x1, y1, x2, y2)  
  
while y1 > 1.0:  
 y1 -= round(BIAS, 1)  
 x2 += round(BIAS, 1)  
 canvas.create\_line(x1\*SCALE, y1\*SCALE, x2\*SCALE, y2\*SCALE)

Протокол работы программы:



Задание 3: Разработать математическую модель и программную реализацию задачи по вычерчиванию 30 стрелок на окружности с центром в (x,y).

Математическая модель: угол вершины рассчитывается по формуле:

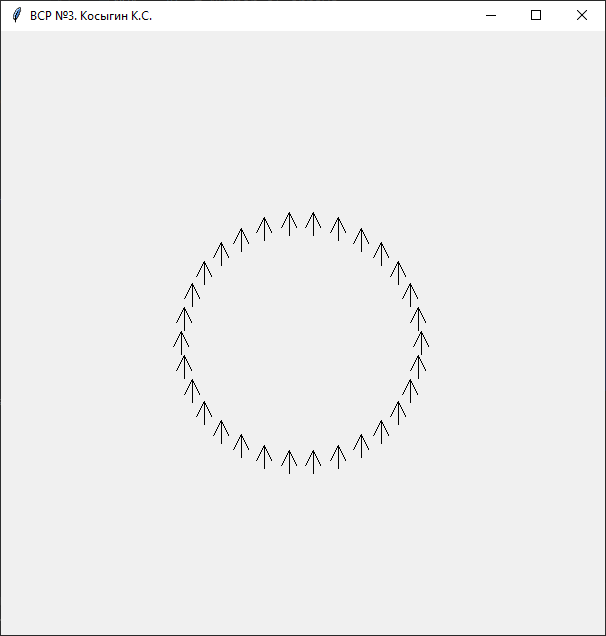
α – угол вершины, i = текущий номер из n вершин на окружности. Соответственно вершина:

xi, yi – точки вершины, r – радиус окружности, α – угол вершины.

Код программы:

def main():  
 for i in range(0, NOA+1):  
 angle = 2 \* PI \* i / NOA  
 xi = round(R \* cos(angle), 5)  
 yi = round(R \* sin(angle), 5)  
  
 arrow\_point = d2s((xi, yi), CS)  
  
 a1 = d2s((xi+XB, yi-YB), CS)  
 a2 = d2s((xi-XB, yi-YB), CS)  
 a3 = d2s((xi, yi-AL), CS)  
  
 canvas.create\_line(\*arrow\_point, \*a1)  
 canvas.create\_line(\*arrow\_point, \*a2)  
 canvas.create\_line(\*arrow\_point, \*a3)

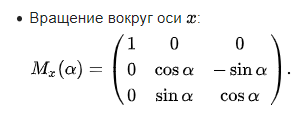
Протокол работы программы:



Инвариантная часть:

Задание: Разработать математическую модель и программную реализацию задачи вращения геометрической фигуры на заданный угол вокруг оси х.

Математическая модель: вращение вокруг оси x осуществляется с помощью мтарицы



Код программы:

def main():  
 for i in range(0, NOA+1):  
 angle = 2 \* PI \* i / NOA  
 xi = round(R \* cos(angle), 5)  
 yi = round(R \* sin(angle), 5)  
  
 arrow\_point = d2s((xi, yi), CS)  
  
 a1 = d2s((xi+XB, yi-YB), CS)  
 a2 = d2s((xi-XB, yi-YB), CS)  
 a3 = d2s((xi, yi-AL), CS)  
  
 canvas.create\_line(\*arrow\_point, \*a1)  
 canvas.create\_line(\*arrow\_point, \*a2)  
 canvas.create\_line(\*arrow\_point, \*a3)

Протокол работы программы:

