**Федеральное государственное образовательное**

**бюджетное учреждение высшего образования**

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ**

**ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**

**ИНСТИТУТ ЗАОЧНОГО И ОТКРЫТОГО ОБРАЗОВАНИЯ**

**ДЕПАРТАМЕНТ АНАЛИЗА ДАННЫХ И МАШИННОГО ОБУЧЕНИЯ**

**ПРАКТИЧЕСКАЯ РАБОТА № 2**

**по дисциплине «Алгоритмы и структуры данных в Python»**

**Вариант №4**

Направление подготовки - 09.03.03 «Прикладная информатика»

Студент(ка) **Белоусов А.А.**

(Ф.И.О.)

Курс2 № группы **ЗБ-ПИ19-1**

Преподаватель **Бочаров М.И.**

(Ф.И.О.)

Москва 2020

Задание

3.2.4. Разработайте реализацию API своего типа *Rectangle* из предыдущих упражнений, где прямоугольник представляется координатами левого нижнего и правого верхнего углов. *Не изменяйте* API.

Код программы

Файл main.py

from stdlib\_python import stddraw  
from Rectangle import Rectangle  
  
  
def drawRectangles(x0, y0, cellSize, width, height):stddraw.setPenColor(stddraw.BLUE)  
 rects = []  
 for i in range(5):  
 rx0 = x0 + i \* cellSize + (cellSize - width) // 2  
 ry0 = y0 + (cellSize - height) // 2rects.append(Rectangle(rx0, ry0, rx0 + width, ry0 + height))  
 rects[i].draw()  
return rects  
  
  
def drawAnotherRects(x0, y0, cellSize, width, height):  
 stddraw.setPenColor(stddraw.RED)  
 rects = []  
 *# center* rx0 = x0 + (cellSize - width) // 2  
 ry0 = y0 + (cellSize - height) // 2  
 rects.append(Rectangle(rx0, ry0, rx0 + width, ry0 + height))  
 rects[-1].draw()  
  
 *# right top* rx0 = x0 + cellSize + (cellSize - width)  
 ry0 = y0 + (cellSize - height)  
 rects.append(Rectangle(rx0, ry0, rx0 + width, ry0 + height))  
 rects[-1].draw()  
  
 *# right bottom* rx0 = x0 + 2 \* cellSize + (cellSize - width)  
 ry0 = y0  
 rects.append(Rectangle(rx0, ry0, rx0 + width, ry0 + height))  
 rects[-1].draw()  
 *# left top* rx0 = x0 + 3 \* cellSize  
 ry0 = y0 + (cellSize - height)  
 rects.append(Rectangle(rx0, ry0, rx0 + width, ry0 + height))  
 rects[-1].draw()  
  
 *# left bottom* rx0 = x0 + 4 \* cellSize  
 ry0 = y0  
 rects.append(Rectangle(rx0, ry0, rx0 + width, ry0 + height))  
 rects[-1].draw()  
  
 return rects  
  
def writeStats(rects, another):  
 print(**"Площади синих: "** + **" "**.join(map(lambda x: str(x.area()), rects)))  
 print(**"Площади красных: "** + **" "**.join(map(lambda x: str(x.area()), another)))  
 print(**"Периметры синих: "** + **" "**.join(map(lambda x: str(x.perimeter()), rects)))  
 print(**"Периметры красных: "** + **" "**.join(map(lambda x: str(x.perimeter()), another)))  
 print(**"Синий пересекает красный: "**+**" "**.join(map(lambda x, y: str(x.intersects(y)), rects,another)))  
 print(**"Синий содержит красный: "**+**" "**.join(map(lambda x, y: str(x.contains(y)), rects,another)))  
  
  
def writeAllStats(rects, another):  
 text1 = {0:**"Правый верхний угол"**,1:**"Правый нижний угол"**, 2:**"Левый нижний угол"**,3:**"Левый верхний угол"**}  
 text2 = {0: **"Нижняя строка"**, 1: **"Верхняя строка"**}  
 for i in range(len(rects)):  
 print(text1[i])  
 for j in range(2):  
 print(text2[j])  
 writeStats(rects[i][j], another[i][j])  
 print()  
 print()  
 print()  
  
  
def draw():  
 stddraw.setCanvasSize(1000, 400)  
 stddraw.setXscale(-500, 500)  
 stddraw.setYscale(-200, 200)  
 stddraw.clear(stddraw.CYAN)  
 stddraw.setPenColor(stddraw.ORANGE)  
 stddraw.filledRectangle(0, 0, 500, 200)  
 stddraw.filledRectangle(-500, -200, 500, 200)  
  
 rt = []  
 rt.append(drawRectangles(0, 0, 100, 35, 25))  
 rt.append(drawRectangles(0, 100, 100, 70, 50))  
  
 rb = []  
 rb.append(drawRectangles(0, -200, 100, 35, 25))  
 rb.append(drawRectangles(0, -100, 100, 70, 50))  
  
 lb = []  
 lb.append(drawRectangles(-500, -200, 100, 35, 25))  
 lb.append(drawRectangles(-500, -100, 100, 70, 50))  
  
  
 lt = []  
 lt.append(drawRectangles(-500, 0, 100, 35, 25))  
 lt.append(drawRectangles(-500, 100, 100, 70, 50))rects = [rt, rb, lb, lt]  
  
 rt = []  
 rt.append(drawAnotherRects(0, 0, 100, 40, 25))  
 rt.append(drawAnotherRects(0, 100, 100, 80, 60))  
  
 rb = []  
 rb.append(drawAnotherRects(0, -200, 100, 35, 30))  
 rb.append(drawAnotherRects(0, -100, 100, 35, 40))  
  
 lb = []  
 lb.append(drawAnotherRects(-500, -200, 100, 35, 25))  
 lb.append(drawAnotherRects(-500, -100, 100, 35, 35))  
  
  
 lt = []  
 lt.append(drawAnotherRects(-500, 0, 100, 25, 20))  
 lt.append(drawAnotherRects(-500, 100, 100, 70, 50))  
  
 another =[rt, rb, lb, lt]writeAllStats(rects, another)  
 stddraw.show()  
  
  
def main():draw()  
  
  
if \_\_name\_\_ == **"\_\_main\_\_"**:  
 main()

Файл Rectangle.py

from stdlib\_python import stddraw  
  
  
class Rectangle:  
 def \_\_init\_\_(self, x0, y0, x1, y1):  
 self.\_x0 = min(x0, x1)  
 self.\_y0 = min(y0, y1)  
 self.\_x1 = max(x0, x1)  
 self.\_y1 = max(y0, y1)  
  
 def area(self):  
 return (self.\_x1 - self.\_x0) \* (self.\_y1 - self.\_y0)  
  
 def perimeter(self):  
 return ((self.\_x1 - self.\_x0) + (self.\_y1 - self.\_y0)) \* 2  
  
 def intersects(self, other):  
 return other.\_x1 >= self.\_x0 and other.\_x0 <= self.\_x1 and other.\_y1 >= self.\_y0 and other.\_y0 <= self.\_y1  
  
 def contains(self, other):  
 return other.\_x0 >= self.\_x0 and other.\_y0 >= self.\_y0 and other.\_x1 <= self.\_x1 and other.\_y1 <= self.\_y1  
  
 def draw(self):  
 stddraw.filledRectangle(self.\_x0, self.\_y0, self.\_x1 - self.\_x0, self.\_y1 - self.\_y0)

Результат выполнения программы

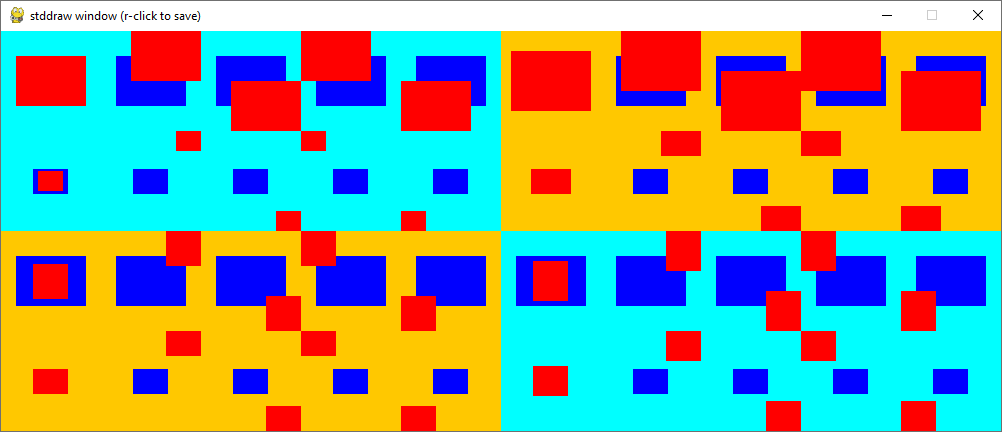


Рис.1 – Отображение прямоугольников

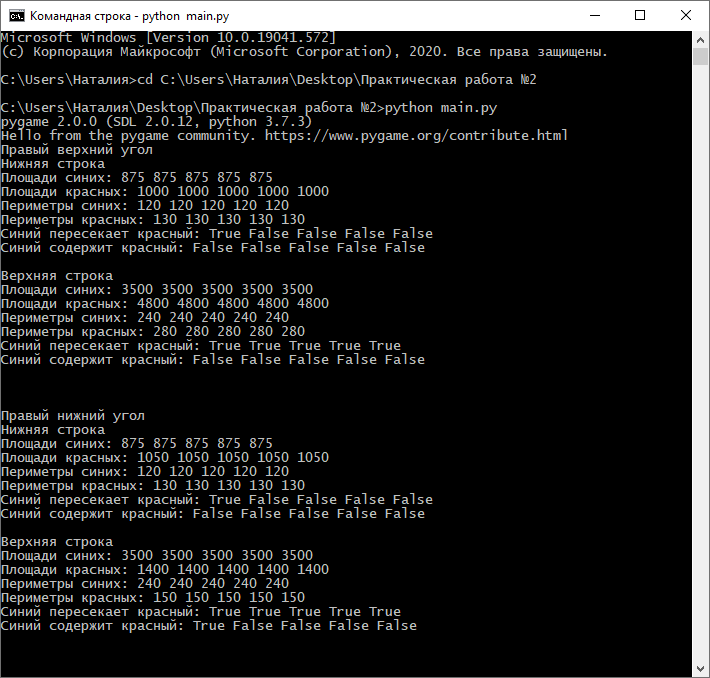


Рис.2 – Информация о прямоугольниках

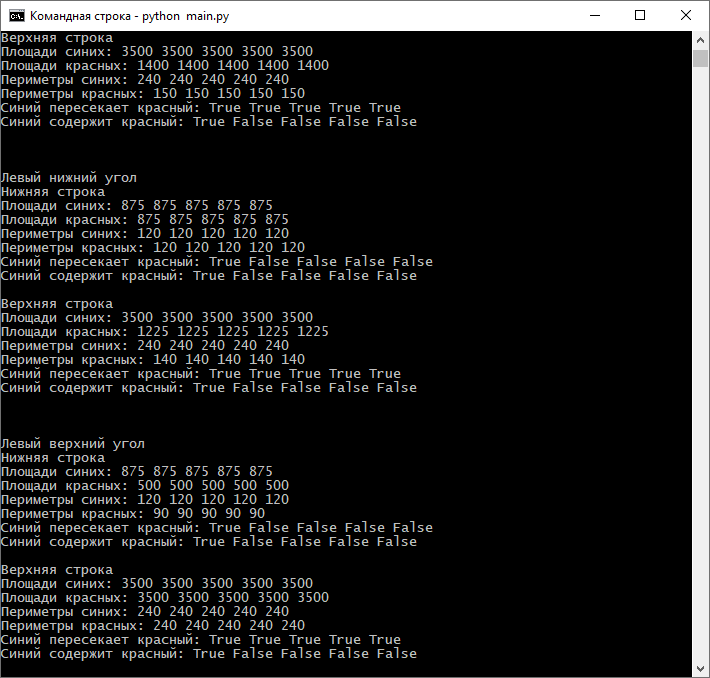


Рис.3 – Информация о прямоугольниках