*Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования*

|  |  |
| --- | --- |
|  | ***«Московский государственный технический университет  имени Н.Э. Баумана»***  ***(МГТУ им. Н.Э. Баумана)*** |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*ФАКУЛЬТЕТ «ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ»*

*КАФЕДРА «ИУ-10»*

**Отчет**

**по домашней работе № 2 часть 2**

**Дисциплина:** Объектно – ориентированное программирование

**Тема:** «PyGame и GitHub»

*Выполнил:*

Студент 1 курса группы ИУ10-22

Литвинов Владимир Михайлович

*Проверил:* Буркацкий Кирилл Александрович

МОСКВА, 2022

# **ЦЕЛЬ РАБОТЫ**

Изучить работу с git и github, а также с библиотекой pygame.draw

# **ХОД РАБОТЫ**

## ***Задача №1:***

Написание вспомогательных функций и функций рисования изображений в файл func.py

## *Код программы:*

import pygame  
from pygame.draw import \*  
from math import \*  
pygame.init()  
FPS = 30  
  
ginger = (200, 115, 5)  
  
  
  
def input\_int(): # Ввод числа и перевод его в тип int  
 s\_number = int(input())  
 return s\_number  
  
  
def input\_many(): # Ввод нескольких чисел, перевод их в тип int и занесение их в массив  
 m\_numbers = []  
 m\_numbers = list(map(int, input().split()))  
 return m\_numbers  
  
  
def angry\_emoji(): # рисует злой смайлик  
 screen = pygame.display.set\_mode((400, 400))  
 screen.fill((80, 80, 80))  
 screen.fill((80, 80, 80))  
 circle(screen, (255, 255, 0), (200, 200), 170)  
 circle(screen, (0, 0, 0), (200, 200), 170, 5)  
 circle(screen, (255, 0, 0), (140, 140), 30)  
 circle(screen, (0, 0, 0), (135, 140), 15)  
 circle(screen, (255, 0, 0), (260, 140), 35)  
 circle(screen, (0, 0, 0), (260, 140), 15)  
 polygon(screen, (0, 0, 0), [(60, 85), (50, 75), (180, 115), (175, 120)])  
 polygon(screen, (0, 0, 0), [(210, 120), (215, 115), (370, 75), (380, 85)])  
 rect(screen, (0, 0, 0), (100, 260, 200, 15))  
  
  
def draw\_ellipse\_angle(surface, color, rect, angle, width=0): # функция поворота эллипса (взята с стаковерфлоу)  
 target\_rect = pygame.Rect(rect)  
 shape\_surf = pygame.Surface(target\_rect.size, pygame.SRCALPHA)  
 pygame.draw.ellipse(shape\_surf, color, (0, 0, \*target\_rect.size), width)  
 rotated\_surf = pygame.transform.rotate(shape\_surf, angle)  
 surface.blit(rotated\_surf, rotated\_surf.get\_rect(center=target\_rect.center))  
  
  
def window\_scale(x, y, i, screen): # рисует окно с параметрами  
 i = i / 10  
 x0 = 415  
 y0 = 10  
 x1 = - x0 \* i + x  
 y1 = - y0 \* i + y  
 rect(screen, (55, 208, 208), (415 \* i + x1, 10 \* i + y1, 375 \* i, 325 \* i)) # стекло  
 rect(screen, (210, 250, 250), (415 \* i + x1, 10 \* i + y1, 375 \* i, 325 \* i), 20) # рама 1  
 line(screen, (210, 250, 250), (415 \* i + x1, 130 \* i + y1), (785 \* i + x1, 130 \* i + y1), 15) # рама 2  
 line(screen, (210, 250, 250), (600 \* i + x1, 10 \* i + y1), (600 \* i + x1, 325 \* i + y1), 15) # рама 3  
  
  
def cat\_scale(x, y, i, screen, color): # рисует кота с параметрами  
 i = i / 10  
 x0 = 20  
 y0 = 370  
 x1 = - x0 \* i + x  
 y1 = - y0 \* i + y  
  
 draw\_ellipse\_angle(screen, color, [563 \* i + x1, 360 \* i + y1, 250 \* i, 80 \* i], 30, 0) # хвост  
 draw\_ellipse\_angle(screen, (0, 0, 0), [563 \* i + x1, 360 \* i + y1, 250 \* i, 80 \* i], 30, 1)  
  
 ellipse(screen, color, (100 \* i + x1, 360 \* i + y1, 530 \* i, 165 \* i)) # тело  
 ellipse(screen, (0, 0, 0), (100 \* i + x1, 360 \* i + y1, 530 \* i, 165 \* i), 1)  
  
 ellipse(screen, color, (60 \* i + x1, 430 \* i + y1, 60 \* i, 65 \* i)) # лапа под головой  
 ellipse(screen, (0, 0, 0), (60 \* i + x1, 430 \* i + y1, 60 \* i, 65 \* i), 1)  
  
 ellipse(screen, color, (20 \* i + x1, 375 \* i + y1, 220 \* i, 110 \* i)) # голова  
 ellipse(screen, (0, 0, 0), (20 \* i + x1, 375 \* i + y1, 220 \* i, 110 \* i), 1)  
  
 ellipse(screen, color, (130 \* i + x1, 480 \* i + y1, 130 \* i, 50 \* i)) # лапа передняя  
 ellipse(screen, (0, 0, 0), (130 \* i + x1, 480 \* i + y1, 130 \* i, 50 \* i), 1)  
  
 ellipse(screen, color, (450 \* i + x1, 425 \* i + y1, 180 \* i, 115 \* i)) # лапа задняя (бедро)  
 ellipse(screen, (0, 0, 0), (450 \* i + x1, 425 \* i + y1, 180 \* i, 115 \* i), 1)  
  
 ellipse(screen, color, (607 \* i + x1, 480 \* i + y1, 50 \* i, 90 \* i)) # лапа задняя (бедро)  
 ellipse(screen, (0, 0, 0), (607 \* i + x1, 480 \* i + y1, 50 \* i, 90 \* i), 1)  
  
 polygon(screen, color,  
 [(x, y), (63 \* i + x1, 392 \* i + y1), (30 \* i + x1, 410 \* i + y1)]) # левое ухо снаружи  
 polygon(screen, (0, 0, 0), [(x, y), (63 \* i + x1, 392 \* i + y1), (30 \* i + x1, 410 \* i + y1)], 1)  
  
 polygon(screen, (250, 180, 240),  
 [(27 \* i + x1, 377 \* i + y1), (55 \* i + x1, 392 \* i + y1), (34 \* i + x1, 403 \* i + y1)]) # левое ухо внутри  
 polygon(screen, (0, 0, 0), [(27 \* i + x1, 377 \* i + y1), (55 \* i + x1, 392 \* i + y1), (34 \* i + x1, 403 \* i + y1)],  
 1)  
  
 polygon(screen, color, [(210 \* i + x1, 355 \* i + y1), (197 \* i + x1, 395 \* i + y1),  
 (160 \* i + x1, 382 \* i + y1)]) # правое ухо снаружи  
 polygon(screen, (0, 0, 0),  
 [(210 \* i + x1, 355 \* i + y1), (197 \* i + x1, 395 \* i + y1), (160 \* i + x1, 382 \* i + y1)], 1)  
  
 polygon(screen, (250, 180, 240), [(202 \* i + x1, 363 \* i + y1), (192 \* i + x1, 388 \* i + y1),  
 (172 \* i + x1, 381 \* i + y1)]) # правое ухо внутри  
 polygon(screen, (0, 0, 0),  
 [(202 \* i + x1, 363 \* i + y1), (192 \* i + x1, 388 \* i + y1), (172 \* i + x1, 381 \* i + y1)], 1)  
  
 ellipse(screen, (114, 158, 47), (55 \* i + x1, 410 \* i + y1, 55 \* i, 40 \* i)) # левый глаз  
 ellipse(screen, (0, 0, 0), (55 \* i + x1, 410 \* i + y1, 55 \* i, 40 \* i), 1)  
  
 ellipse(screen, (114, 158, 47), (150 \* i + x1, 410 \* i + y1, 55 \* i, 40 \* i)) # правый глаз  
 ellipse(screen, (0, 0, 0), (150 \* i + x1, 410 \* i + y1, 55 \* i, 40 \* i), 1)  
  
 ellipse(screen, (0, 0, 0), (85 \* i + x1, 410 \* i + y1, 10 \* i, 40 \* i)) # левый зрачок  
  
 ellipse(screen, (0, 0, 0), (180 \* i + x1, 410 \* i + y1, 10 \* i, 40 \* i)) # правый зрачок  
  
 draw\_ellipse\_angle(screen, (255, 255, 255), [73 \* i + x1, 410 \* i + y1, 8 \* i, 20 \* i], 220, 0) # левый блик  
  
 draw\_ellipse\_angle(screen, (255, 255, 255), [168 \* i + x1, 410 \* i + y1, 8 \* i, 20 \* i], 220, 0) # правый блик  
  
 polygon(screen, (250, 180, 240),  
 [(127 \* i + x1, 450 \* i + y1), (139 \* i + x1, 450 \* i + y1), (127 \* i + x1, 457 \* i + y1),  
 (115 \* i + x1, 450 \* i + y1)]) # нос  
 polygon(screen, (0, 0, 0),  
 [(127 \* i + x1, 450 \* i + y1), (139 \* i + x1, 450 \* i + y1), (127 \* i + x1, 457 \* i + y1),  
 (115 \* i + x1, 450 \* i + y1)], 1)  
  
 line(screen, (0, 0, 0), (127 \* i + x1, 457 \* i + y1), (127 \* i + x1, 467 \* i + y1), 1)  
  
 arc(screen, (0, 0, 0), [127 \* i + x1, 462 \* i + y1, 16 \* i, 10 \* i], pi, 15 \* pi / 8) # рот  
 arc(screen, (0, 0, 0), [111 \* i + x1, 462 \* i + y1, 16 \* i, 10 \* i], pi, 15 \* pi / 8)  
  
 # усы  
 arc(screen, (0, 0, 0), [-80 \* i + x1, 444 \* i + y1, 700 \* i, 500 \* i], 1.55, 1.9)  
 arc(screen, (0, 0, 0), [-78 \* i + x1, 455 \* i + y1, 700 \* i, 200 \* i], 1.55, 1.9)  
 arc(screen, (0, 0, 0), [-145 \* i + x1, 463 \* i + y1, 700 \* i, 200 \* i], 1.3, 1.7)  
  
 arc(screen, (0, 0, 0), [-375 \* i + x1, 440 \* i + y1, 700 \* i, 500 \* i], 1.2, 1.65)  
 arc(screen, (0, 0, 0), [-410 \* i + x1, 448 \* i + y1, 700 \* i, 200 \* i], 1.1, 1.5)  
 arc(screen, (0, 0, 0), [-345 \* i + x1, 460 \* i + y1, 700 \* i, 200 \* i], 1.3, 1.75)  
  
  
def ball\_scale(x, y, i, screen): # рисует клубок с параметрами  
 i = i / 10  
 x0 = 400  
 y0 = 560  
 x1 = - x0 \* i + x  
 y1 = - y0 \* i + y  
  
 ellipse(screen, (180, 180, 180), (400 \* i + x1, 560 \* i + y1, 135 \* i, 100 \* i)) # клубок  
 ellipse(screen, (0, 0, 0), (400 \* i + x1, 560 \* i + y1, 135 \* i, 100 \* i), 1)  
 arc(screen, (0, 0, 0), (420 \* i + x1, 570 \* i + y1, 100 \* i, 75 \* i), 0, pi / 2)  
 arc(screen, (0, 0, 0), (405 \* i + x1, 580 \* i + y1, 100 \* i, 50 \* i), 0, pi / 2)  
 arc(screen, (0, 0, 0), (325 \* i + x1, 590 \* i + y1, 185 \* i, 100 \* i), 0, pi / 2)  
 arc(screen, (0, 0, 0), (430 \* i + x1, 600 \* i + y1, 80 \* i, 80 \* i), 2 \* pi / 3, pi)  
 arc(screen, (0, 0, 0), (450 \* i + x1, 610 \* i + y1, 80 \* i, 80 \* i), 2 \* pi / 3, pi)  
  
 arc(screen, (180, 180, 180), (310 \* i + x1, 590 \* i + y1, 300 \* i, 60 \* i), pi, 3 \* pi / 2) # нить  
 arc(screen, (180, 180, 180), (160 \* i + x1, 600 \* i + y1, 150 \* i, 40 \* i), 0, pi)  
  
  
def big\_cat(): # рисует стандартных окно, кота и клубок (задание 2)  
 screen = pygame.display.set\_mode((800, 700))  
 screen.fill((95, 80, 30))  
 rect(screen, (140, 120, 50), (0, 350, 800, 350)) # стол  
 window\_scale(415, 10, 10, screen) # окно  
 cat\_scale(20, 370, 10, screen, ginger) # кот  
 ball\_scale(400, 560, 10, screen) # клубок  
  
  
def many\_cats(): # рисует картину с множеством котов (задание 3)  
 screen = pygame.display.set\_mode((800, 700))  
 screen.fill((95, 80, 30))  
 rect(screen, (140, 120, 50), (0, 350, 800, 350)) # стол  
 window\_scale(415, 10, 10, screen)  
 window\_scale(10, 10, 10, screen)  
 ball\_scale(20, 370, 10, screen)  
 ball\_scale(50, 450, 3, screen)  
 ball\_scale(280, 550, 4, screen)  
 ball\_scale(111, 650, 4, screen)  
 ball\_scale(350, 500, 3, screen)  
 ball\_scale(550, 370, 6, screen)  
 cat\_scale(20, 370, 2, screen, ginger)  
 cat\_scale(50, 450, 3, screen, ginger)  
 cat\_scale(100, 600, 2, screen, ginger)  
 cat\_scale(350, 370, 3, screen, ginger)  
 cat\_scale(700, 370, 2, screen, ginger)  
 cat\_scale(450, 560, 4, screen, ginger)  
  
  
def pygame\_done(): # основные функции работы pygame и ее завершение  
 pygame.display.update()  
 clock = pygame.time.Clock()  
 finished = False  
  
 while not finished:  
 clock.tick(FPS)  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 finished = True  
  
 pygame.quit()

## ***Задача №2:***

Написание файла main.py с вызовами функций из func.py

## *Код программы:*

import pygame  
from pygame.draw import \*  
from func import \*  
pygame.init()  
FPS = 30  
  
  
def main\_cycle(z):  
 print('Выберите номер задания:')  
 print(' 1. Вывод злого смайлика', '\n',  
 '2. Вывод картины с котом', '\n',  
 '3. Много котов (с возможностью выбора режима)')  
 task = input\_int()  
 if task == 1:  
 angry\_emoji()  
 elif task == 2:  
 big\_cat()  
 elif task == 3:  
 print('Выберите режим:', '\n',  
 '1. Заданная картинка', '\n',  
 '2. Расставить объекты самому')  
 mode = input\_int()  
 if mode == 1:  
 many\_cats()  
 elif mode == 2:  
 screen = pygame.display.set\_mode((800, 700))  
 screen.fill((95, 80, 30))  
 rect(screen, (140, 120, 50), (0, 350, 800, 350)) # стол  
 window\_scale(415, 10, 10, screen)  
 window\_scale(10, 10, 10, screen)  
 print('Введите кол-во котов и клубков через пробел')  
 a = input\_many()  
 cats, balls = a[0], a[1]  
 print('Помните, что координата "y" края стола - 350')  
 for i in range(1, cats+1):  
 print('Введите координаты и размер (стандарт - 10) кота №', i)  
 b = input\_many()  
 print('Введите цвет кота')  
 color = input()  
 cat\_scale(b[0], b[1], b[2], screen, color)  
 for i in range(1, balls+1):  
 print('Введите координаты и размер (стандарт - 10) клубка №', i)  
 b = input\_many()  
 ball\_scale(b[0], b[1], b[2], screen)  
 else:  
 print('Такого режима работы не существует!')  
 else:  
 print('Такого задания не существует!')  
 pygame\_done()  
  
  
print('Данная программа позволяет выводить на экран изображения, сделанные с помощью библиотеки PyGame', '\n')  
z = 1  
while z == 1:  
 main\_cycle(z)  
 print('Хотите продолжить?', '\n',  
 '1 - да', '\n',  
 '2 - нет')  
 z = input\_int()

# 

# **ВЫВОД ПО ПРОДЕЛАННОЙ РАБОТЕ И ПРИМЕЧАНИЯ**

Мною был изучены и поняты основные принципы работы с git, github и pygame.

Все файлы лабораторной работы можно найти в репозитории <https://github.com/SystErrr/OOP_2022>

В 3е задание добавлена возможность собственноручного задания кол-ва, размеров и нач. координат объектов.

Также для каждой функции написаны соответствующие комментарии.