**Московский государственный технический университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

# Отчет по Лабораторной работе 3

Выполнил:
студент группы ИУ5-35Б
Пермяков Семён
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Юрий Евгеньевич
Подпись и дата:

# Постановка задачи

Создать приложение Змейка с использованием pygame, без использования спрайтов.

# Текст программы

```
import pygame
import sys
import random
import time

class Game:
    def __init__(self):
        self.screen_width = 720
        self.screen_height = 360

        self.red = pygame.Color(255, 0, 0)
        self.green = pygame.Color(0, 255, 0)
        self.black = pygame.Color(0, 0, 0)
        self.white = pygame.Color(255, 255, 255)
        self.brown = pygame.Color(165, 42, 42)

        self.fps_controller = pygame.time.Clock()

        self.score = 0

    def init_and_check_for_errors(self):
        check_errors = pygame.init()
        if check_errors[1] > 0:
            sys.exit()
        else:
            print("ok")
    def set_surface_and_title(self):
        self.play_surface = pygame.display.set_mode((self.screen_width,
self.screen_height))
        pygame.display.set_caption("Игра: Змейка")

    def event_loop(self, change_to):
        for event in pygame.event.get():
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_RIGHT or event.key == ord("d"):
                    change_to = "RIGHT"
                elif event.key == pygame.K_LEFT or event.key == ord("a"):
                    change_to = "LEFT"
                elif event.key == pygame.K_UP or event.key == ord("w"):
                    change_to = "UP"
                elif event.key == pygame.K_DOWN or event.key == ord("s"):
                    change_to = "DOWN"
                elif event.key == pygame.K_ESCAPE:
                    pygame.quit()
                    sys.exit()
        return change_to

    def refresh_screen(self):
        pygame.display.flip()
        game.fps_controller.tick(15)

    def show_score(self, choice=1):
        s_font = pygame.font.SysFont("monako", 24)
        s_surf = s_font.render("Score: {0}".format(self.score), True, self.black)
        s_rect = s_surf.get_rect()
        if choice == 1:
            s_rect.midtop = (80, 10)
        else:
            s_rect.midtop == (360, 120)
        self.play_surface.blit(s_surf, s_rect)
```

```
    def game_over(self):
        go_font = pygame.font.SysFont("monaco",72)
        go_surf = go_font.render("Game over",True,self.red)
        go_rect = go_surf.get_rect()
        go_rect.midtop = (360, 15)
        self.play_surface.blit(go_surf, go_rect)
        self.show_score(0)
        pygame.display.flip()
        time.sleep(3)
        pygame.quit()
        sys.exit()

class Snake:
    def __init__(self, snake_color):
        self.snake_head_pos = [100, 50]
        self.snake_body = [[100, 50], [90, 50], [80, 50]]
        self.snake_color = snake_color
        self.direction = "RIGHT"
        self.change_to = self.direction
    def validate_derication_and_changes(self):
        if any((self.change_to == "RIGHT" and not self.direction == "LEFT",
                self.change_to == "LEFT" and not self.direction == "RIGHT",
                self.change_to == "UP" and not self.direction == "DOWN",
                self.change_to == "DOWN" and not self.direction == "UP")):
            self.direction = self.change_to
    def change_head_position(self):
        if self.direction == "RIGHT":
            self.snake_head_pos[0] += 10
        elif self.direction == "LEFT":
            self.snake_head_pos[0] -= 10
        elif self.direction == "UP":
            self.snake_head_pos[1] -= 10
        elif self.direction == "DOWN":
            self.snake_head_pos[1] += 10
    def snake_body_mechanism(self, score, food_pos, screen_width, screen_height):
        self.snake_body.insert(0, list(self.snake_head_pos))
        if (self.snake_head_pos[0] == food_pos[0] and self.snake_head_pos[1] ==
food_pos[1]):
            food_pos = [random.randrange(1, screen_width/10) * 10,
random.randrange(1, screen_height/10) * 10]
            score += 1
        else:
            self.snake_body.pop()
        return score, food_pos
    def draw_snake(self, play_surface, surfase_color):
        play_surface.fill(surfase_color)#
        for pos in self.snake_body:
            pygame.draw.rect(play_surface, self.snake_color, pygame.Rect(pos[0],
pos[1], 10, 10))
    def check_for_boundaries(self, game_over, screen_width, screen_height):
        if any((self.snake_head_pos[0] > screen_width-10 or self.snake_head_pos[0]
< 0,
            self.snake_head_pos[1] > screen_height-10 or self.snake_head_pos[1] <
0)):
            game_over()
        for block in self.snake_body[1:]:
            if(block[0] == self.snake_head_pos[0] and block[1] ==
self.snake_head_pos[1]):
                game_over()


class Food:
    def __init__(self, food_color, screen_width, screen_height):
        self.food_color = food_color
        self.food_size_x = 10
        self.food_size_y = 10
```

```
        self.food_pos = [random.randrange(1, screen_width/10) * 10,
random.randrange(1, screen_height/10) * 10]
    def draw_food(self, play_surface):
        pygame.draw.rect(play_surface, self.food_color,
pygame.Rect(self.food_pos[0], self.food_pos[1], self.food_size_x,
self.food_size_y))

game = Game()
snake = Snake(game.green)
food = Food(game.brown, game.screen_width, game.screen_height)

game.init_and_check_for_errors()
game.set_surface_and_title()

while True:
    snake.change_to = game.event_loop(snake.change_to)
    snake.validate_derication_and_changes()
    snake.change_head_position()
    game.score, food.food_pos = snake.snake_body_mechanism(game.score,
food.food_pos, game.screen_width, game.screen_height)
    snake.draw_snake(game.play_surface, game.white)

    food.draw_food(game.play_surface)

    snake.check_for_boundaries(game.game_over, game.screen_width,
game.screen_height)
    game.show_score()
    game.refresh_screen()
```

## Анализ результатов


Игра: Змейка
Score: 4
Game over