|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования**  **«Московский государственный технический университет имени Н.Э. Баумана**  **(национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)** |

**Факультет «Информатика и системы управления»**

**Кафедра «Системы обработки информации и управления»**

Новицкий Ярослав ИУ5-35Б

Парадигмы и конструкции языков программирования

**ОТЧЁТ ПО**

**Домашнему заданию**

Москва

2023

**Постановка задачи.**

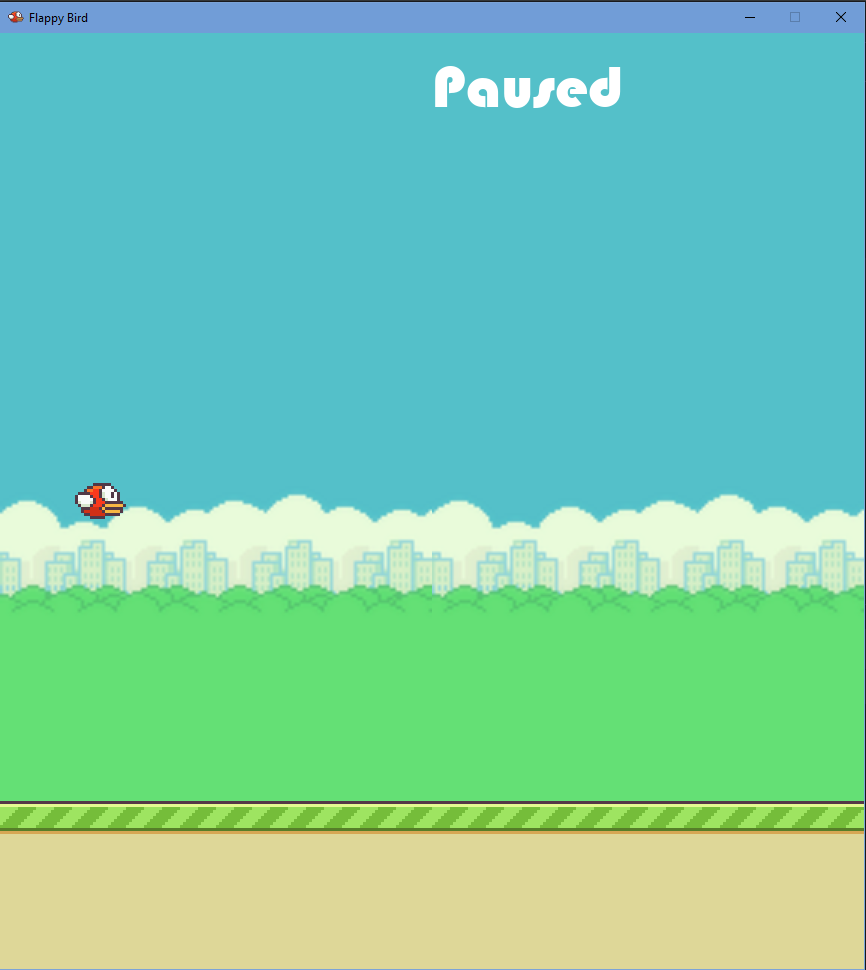
Написать полнофункциональную игру "Flappy Bird" с использованием библиотеки Pygame, обеспечивающую интерактивное управление птицей, генерацию препятствий и подсчет счета игрока.

**Текст программы.**

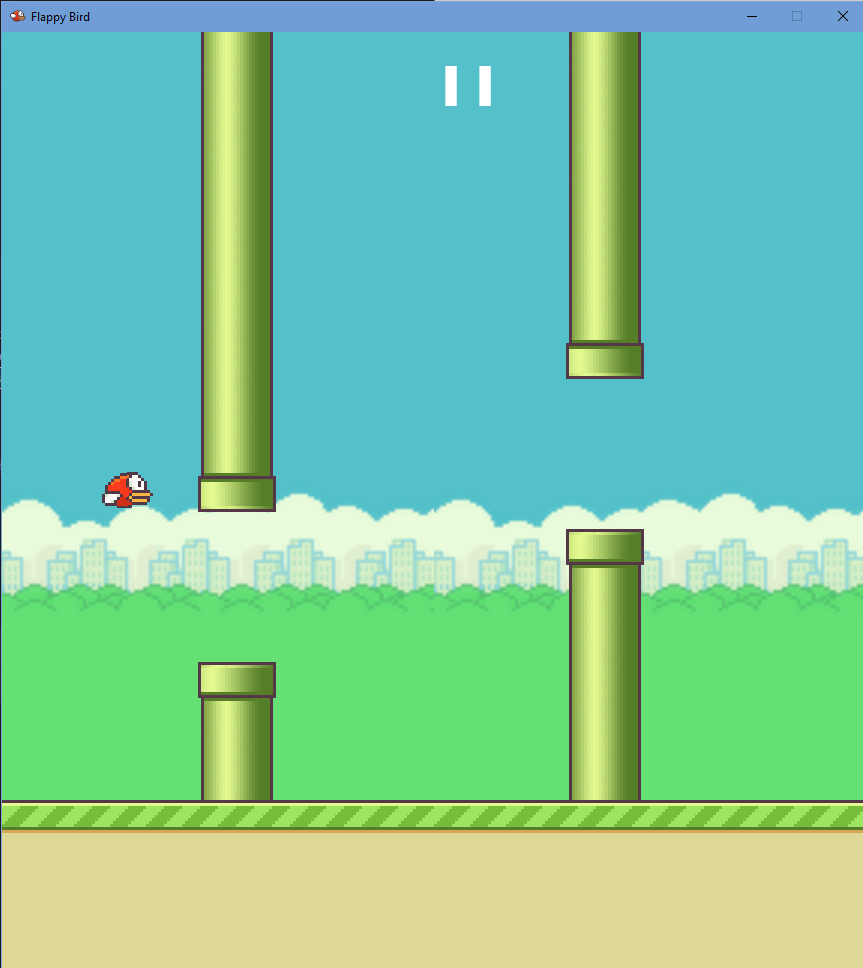
import pygame  
import random  
  
pygame.init()  
  
clock = pygame.time.Clock()  
fps = 60  
  
screen\_width = 864  
screen\_height = 936  
  
screen = pygame.display.set\_mode((screen\_width, screen\_height))  
pygame.display.set\_caption('Flappy Bird')  
  
#define font  
font = pygame.font.SysFont('Bauhaus 93', 60)  
  
#define colours  
white = (255, 255, 255)  
  
#define game variables  
ground\_scroll = 0  
scroll\_speed = 4  
flying = False  
game\_over = False  
pipe\_gap = 150  
pipe\_frequency = 1500 #milliseconds  
last\_pipe = -pipe\_frequency  
score = 0  
pause\_time = 0  
stop\_time = 0  
pass\_pipe = False  
  
  
#load images  
bg = pygame.image.load('img/bg.png')  
ground\_img = pygame.image.load('img/ground.png')  
button\_img = pygame.image.load('img/restart.png')  
game\_over\_img = pygame.image.load('img/game\_over2.png')  
  
  
def draw\_text(text, font, text\_col, x, y):  
 img = font.render(text, True, text\_col)  
 screen.blit(img, (x, y))  
  
  
def reset\_game():  
 pipe\_group.empty()  
 flappy.rect.x = 100  
 flappy.rect.y = int(screen\_height / 2)  
 score = 0  
 return score  
  
  
  
class Bird(pygame.sprite.Sprite):  
 def \_\_init\_\_(self, x, y):  
 pygame.sprite.Sprite.\_\_init\_\_(self)  
 self.images = []  
 self.index = 0  
 self.counter = 0  
 for num in range(1, 4):  
 img = pygame.image.load(f'img/bird{num}.png')  
 self.images.append(img)  
 self.image = self.images[self.index]  
 self.rect = self.image.get\_rect()  
 self.rect.center = [x, y]  
 self.vel = 0  
 self.clicked = False  
  
 def update(self):  
  
 if flying == True:  
 #gravity  
 self.vel += 0.5  
 if self.vel > 8:  
 self.vel = 8  
 if self.rect.bottom < 768:  
 self.rect.y += int(self.vel)  
  
 if game\_over == False:  
 #jump  
 if pygame.mouse.get\_pressed()[0] == 1 and self.clicked == False:  
 self.clicked = True  
 self.vel = -10  
 if pygame.mouse.get\_pressed()[0] == 0:  
 self.clicked = False  
  
 #handle the animation  
 self.counter += 1  
  
 if self.counter > 5:  
 self.counter = 0  
 self.index += 1  
 if self.index >= len(self.images):  
 self.index = 0  
 self.image = self.images[self.index]  
  
 #rotate the bird  
 self.image = pygame.transform.rotate(self.images[self.index], self.vel \* -2)  
 else:  
 self.image = pygame.transform.rotate(self.images[self.index], -90)  
  
  
  
class Pipe(pygame.sprite.Sprite):  
 def \_\_init\_\_(self, x, y, position):  
 pygame.sprite.Sprite.\_\_init\_\_(self)  
 self.image = pygame.image.load('img/pipe.png')  
 self.rect = self.image.get\_rect()  
 #position 1 is from the top, -1 is from the bottom  
 if position == 1:  
 self.image = pygame.transform.flip(self.image, False, True)  
 self.rect.bottomleft = [x, y - int(pipe\_gap / 2)]  
 if position == -1:  
 self.rect.topleft = [x, y + int(pipe\_gap / 2)]  
  
 def update(self):  
 self.rect.x -= scroll\_speed  
 if self.rect.right < 0:  
 self.kill()  
  
  
class Button():  
 def \_\_init\_\_(self, x, y, image):  
 self.image = image  
 self.rect = self.image.get\_rect()  
 self.rect.topleft = (x, y)  
  
 def draw(self):  
  
 action = False  
  
 #get mouse position  
 pos = pygame.mouse.get\_pos()  
  
 #check if mouse is over the button  
 if self.rect.collidepoint(pos):  
 if pygame.mouse.get\_pressed()[0] == 1:  
 action = True  
  
 #draw button  
 screen.blit(self.image, (self.rect.x, self.rect.y))  
  
 return action  
  
bird\_group = pygame.sprite.Group()  
pipe\_group = pygame.sprite.Group()  
  
flappy = Bird(100, int(screen\_height / 2))  
  
bird\_group.add(flappy)  
  
#create restart button instance  
button = Button(screen\_width // 2 - 50, screen\_height // 2 - 100, button\_img)  
  
run = True  
while run:  
  
 clock.tick(fps)  
  
 #draw background  
 screen.blit(bg, (0,0))  
  
 bird\_group.draw(screen)  
 bird\_group.update()  
 pipe\_group.draw(screen)  
  
 #draw the ground  
 screen.blit(ground\_img, (ground\_scroll, 768))  
  
 #check the score  
 if len(pipe\_group) > 0:  
 if bird\_group.sprites()[0].rect.left > pipe\_group.sprites()[0].rect.left\  
 and bird\_group.sprites()[0].rect.right < pipe\_group.sprites()[0].rect.right\  
 and pass\_pipe == False:  
 pass\_pipe = True  
 if pass\_pipe == True:  
 if bird\_group.sprites()[0].rect.left > pipe\_group.sprites()[0].rect.right:  
 score += 1  
 pass\_pipe = False  
  
 if flying == True or game\_over == True:  
 draw\_text(str(score), font, white, int(screen\_width / 2), 20)  
  
 #look for collision  
 if pygame.sprite.groupcollide(bird\_group, pipe\_group, False, False) or flappy.rect.top < 0:  
 game\_over = True  
  
 #check if bird has hit the ground  
 if flappy.rect.bottom >= 768:  
 game\_over = True  
 flying = False  
  
  
 if game\_over == False and flying == True:  
  
 #generate new pipes  
  
 time\_now = pygame.time.get\_ticks() - pause\_time  
 print(time\_now)  
 print(last\_pipe)  
 if time\_now - last\_pipe > pipe\_frequency:  
 pipe\_height = random.randint(-100, 100)  
 btm\_pipe = Pipe(screen\_width, int(screen\_height / 2) + pipe\_height, -1)  
 top\_pipe = Pipe(screen\_width, int(screen\_height / 2) + pipe\_height, 1)  
 pipe\_group.add(btm\_pipe)  
 pipe\_group.add(top\_pipe)  
 last\_pipe = time\_now  
  
  
 #draw and scroll the ground  
 ground\_scroll -= scroll\_speed  
 if abs(ground\_scroll) > 35:  
 ground\_scroll = 0  
  
 pipe\_group.update()  
  
  
 #check for game over and reset  
 if game\_over == True:  
 screen.blit(game\_over\_img, (screen\_width // 2 - 217, screen\_height // 2 - 270))  
 if button.draw() == True :  
 game\_over = False  
 score = reset\_game()  
 last\_pipe = -pipe\_frequency  
 score = 0  
 pause\_time = 0  
 stop\_time = 0  
  
  
 if flying == False and game\_over == False:  
 draw\_text('Paused', font, white, int(screen\_width / 2), 20)  
  
  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 run = False  
 if event.type == pygame.MOUSEBUTTONDOWN and flying == False and game\_over == False:  
 flying = True  
 pause\_time += pygame.time.get\_ticks() - stop\_time  
 if event.type == pygame.KEYDOWN and event.key == pygame.K\_SPACE and flying == True and game\_over == False:  
 flying = False  
 stop\_time = pygame.time.get\_ticks()  
  
  
 pygame.display.update()  
  
pygame.quit()

**Экранные формы:**

Старт/пауза



Полёт



Окончание игры

