**Space Scavenger**

Изработил:  
Филип Стамболиев

215026

class Spaceship:  
 def \_\_init\_\_(self):  
 self.image = pygame.transform.scale(spaceship\_img, (60, 60))  
 self.x = screen\_width // 2  
 self.y = screen\_height - 100  
 self.speed = 5  
  
 def move(self, keys):  
 if keys[pygame.K\_LEFT] and self.x > 0:  
 self.x -= self.speed  
 if keys[pygame.K\_RIGHT] and self.x < screen\_width - 60:  
 self.x += self.speed  
  
 def draw(self):  
 screen.blit(self.image, (self.x, self.y))

**Spaceship**

* **Опис:** Ги управува движењата и изгледот на вселенскиот брод.
* **Методи:**
  + \_\_init\_\_(): Иницијализира координати, брзина и слика на вселенскиот брод.
  + move(keys): Движење лево или десно според внесот од тастатурата.
  + draw(): Прикажување на вселенскиот брод на екранот.

class Asteroid:  
 def \_\_init\_\_(self):  
 self.image = pygame.transform.scale(asteroid\_img, (50, 50))  
 self.x = random.randint(0, screen\_width - 50)  
 self.y = -50  
 self.speed = random.randint(3, 6)  
  
 def move(self):  
 self.y += self.speed  
  
 def draw(self):  
 screen.blit(self.image, (self.x, self.y))

**Asteroid**

* **Опис:** Генерира астероиди што се движат надолу и треба да се избегнат.
* **Методи:**
  + \_\_init\_\_(): Иницијализира почетни координати и брзина на астероидот.
  + move(): Движење на астероидот надолу.
  + draw(): Прикажување на астероидот на екранот.

class Crystal:  
 def \_\_init\_\_(self):  
 self.image = pygame.transform.scale(crystal\_img, (40, 40))  
 self.x = random.randint(0, screen\_width - 40)  
 self.y = -40  
 self.speed = random.randint(2, 4)  
  
 def move(self):  
 self.y += self.speed  
  
 def draw(self):  
 screen.blit(self.image, (self.x, self.y))

**Crystal**

* **Опис:** Генерира енергетски кристали што треба да се соберат.
* **Методи:**
  + \_\_init\_\_(): Иницијализира почетни координати и брзина на кристалот.
  + move(): Движење на кристалот надолу.
  + draw(): Прикажување на кристалот на екранот.

def game\_loop():  
 clock = pygame.time.Clock()  
 spaceship = Spaceship()  
 asteroids = [Asteroid() for \_ in range(5)]  
 crystals = [Crystal() for \_ in range(3)]  
 score = 0  
 running = True  
  
 while running:  
 screen.fill((0, 0, 0))  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 running = False  
  
 keys = pygame.key.get\_pressed()  
 spaceship.move(keys)  
  
 for asteroid in asteroids:  
 asteroid.move()  
 if asteroid.y > screen\_height:  
 asteroids.remove(asteroid)  
 asteroids.append(Asteroid())  
 if spaceship.x < asteroid.x + 50 and spaceship.x + 60 > asteroid.x and spaceship.y < asteroid.y + 50 and spaceship.y + 60 > asteroid.y:  
 pygame.mixer.Sound.play(clash\_sound)  
 running = False  
  
 for crystal in crystals:  
 crystal.move()  
 if crystal.y > screen\_height:  
 crystals.remove(crystal)  
 crystals.append(Crystal())  
 if spaceship.x < crystal.x + 40 and spaceship.x + 60 > crystal.x and spaceship.y < crystal.y + 40 and spaceship.y + 60 > crystal.y:  
 score += 1  
 crystals.remove(crystal)  
 crystals.append(Crystal())  
  
 spaceship.draw()  
 for asteroid in asteroids:  
 asteroid.draw()  
 for crystal in crystals:  
 crystal.draw()  
  
 score\_text = font.render(f"Score: {score}", True, white)  
 screen.blit(score\_text, (10, 10))  
  
 pygame.display.flip()  
 clock.tick(60)  
  
 pygame.quit()

**game\_loop()**

* **Опис:** Главната играчка петља која управува со целокупната логика на играта.
* **Акции:**
  + Обработка на внес од тастатурата.
  + Движење на вселенскиот брод, астероидите и кристалите.
  + Проверка за судири (со астероиди или кристали).
  + Ажурирање на резултатот.
  + Прикажување на елементите на екранот.

**Други функционалности**

* **Позадинска музика и звуци:**
  + Се поставува позадинска музика (background\_music.wav) која се пушта бесконечно.
  + Звучен ефект (clash\_sound.wav) се активира при судир со астероид.