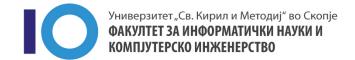
# LAB 2

#### SPACESHIP GAME

#### Програмирање на видео игри



Решение: github link

# Објаснување на функционалности:

Се иницијализира рудате, се поставува прозорец со димензии 800x600 и се вчитуваат потребните ресурси (слики, звуци, музика).

```
# Screen dimensions
WIDTH, HEIGHT = 800, 600

# Colors
WHITE = (255, 255, 255)
BLACK = (0, 0, 0)

#Materials
spaceship_img = pygame.image.load('spaceship.png')
asteroid_img = pygame.image.load('asteroid.png')
energy_crystal_img = pygame.image.load('energy_crystal.png')
background_music = 'background_music.wav'
clash_sound = pygame.mixer.Sound('clash_sound.wav')
```

## Класа Spaceship

Креира објект за контролирање на вселенското летало. Се управува со стрелките на тастатурата, движејќи се низ екран.

```
class Spaceship(pygame.sprite.Sprite):
  def init (self):
      super(). init ()
      self.image = spaceship img
      self.rect = self.image.get rect(center=(WIDTH // 2, HEIGHT // 2))
      self.speed = 5
  def update(self):
      keys = pygame.key.get_pressed()
      if keys[pygame.K LEFT] and self.rect.left > 0:
          self.rect.x -= self.speed
      if keys[pygame.K_RIGHT] and self.rect.right < WIDTH:</pre>
          self.rect.x += self.speed
      if keys[pygame.K UP] and self.rect.top > 0:
          self.rect.y -= self.speed
      if keys[pygame.K DOWN] and self.rect.bottom < HEIGHT:</pre>
          self.rect.y += self.speed
```

#### Класа Asteroid

Креира астероиди кои влегуваат од различни страни на екранот со случајна брзина. Ако излезат надвор од екранот, се отстрануваат.

```
# Asteroid class
class Asteroid(pygame.sprite.Sprite):
  def __init__(self):
      super(). init ()
      self.image = asteroid img
      side = random.choice(['left', 'right', 'top', 'bottom'])
      if side == 'left':
           self.rect = self.image.get rect(center=(random.randint(-50, 0),
random.randint(0, HEIGHT)))
      elif side == 'right':
          self.rect = self.image.get_rect(center=(random.randint(WIDTH,
WIDTH + 50), random.randint(0, HEIGHT)))
      elif side == 'top':
           self.rect = self.image.get rect(center=(random.randint(0, WIDTH))
random.randint(-50, 0)))
           self.rect = self.image.get rect(center=(random.randint(0, WIDTH))
random.randint(HEIGHT, HEIGHT + 50)))
      self.speed x = random.choice([-1, 1]) * random.uniform(0.5, 2)
      self.speed y = random.choice([-1, 1]) * random.uniform(0.5, 2)
```

# Класа EnergyCrystal

исто така влегуваат од различни страни и се движат случајно. Играчот собира кристали за да го зголеми својот резултат.

```
# EnergyCrystal class
class EnergyCrystal(pygame.sprite.Sprite):
    def __init__(self):
        super().__init__()
        self.image = energy_crystal_img
        side = random.choice(['left', 'right', 'top', 'bottom'])
        if side == 'left':
            self.rect = self.image.get_rect(center=(random.randint(-50, 0),
random.randint(0, HEIGHT)))
    elif side == 'right':
            self.rect = self.image.get_rect(center=(random.randint(WIDTH, WIDTH
+ 50), random.randint(0, HEIGHT)))
    elif side == 'top':
            self.rect = self.image.get_rect(center=(random.randint(0, WIDTH),
random.randint(-50, 0)))
    else:
            self.rect = self.image.get_rect(center=(random.randint(0, WIDTH),
```

```
random.randint(HEIGHT, HEIGHT + 50)))
    self.speed_x = random.choice([-1, 1]) * random.uniform(0.5, 2)
    self.speed_y = random.choice([-1, 1]) * random.uniform(0.5, 2)
```

## Класа Grid

Поставува позадина со мрежа што симулира движење. Мрежата се ажурира врз основа на движењето на играчот.

#### Главна логика

- Во секоја итерација на играта се проверуваат:
  - Движење на играчот, астероидите и кристалите.
  - Судири меѓу играчот и објектите.
- При судир со астероид, се активира звук и играта завршува.
- Играчот може да рестартира играта со копчето R.

```
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            sys.exit()
```

```
if not game over:
       # Spawn asteroids
       if random.randint(1, 30) == 1:
          asteroid = Asteroid()
           all sprites.add(asteroid)
           asteroids.add(asteroid)
       # Spawn energy crystals
       if random.randint(1, 60) == 1:
           energy crystal = EnergyCrystal()
           all sprites.add(energy_crystal)
           energy_crystals.add(energy_crystal)
       # Update
       dx = dy = 0
       keys = pygame.key.get pressed()
       if keys[pygame.K LEFT]:
           dx = player.speed
       elif keys[pygame.K RIGHT]:
           dx = -player.speed
       if keys[pygame.K UP]:
           dy = player.speed
       elif keys[pygame.K DOWN]:
           dy = -player.speed
       grid.update(dx, dy)
       all_sprites.update()
       # Move asteroids and crystals with the grid
       for sprite in asteroids.sprites() + energy_crystals.sprites():
           sprite.rect.x += dx
           sprite.rect.y += dy
       # Check collisions
       if pygame.sprite.spritecollideany(player, asteroids):
           clash sound.play()
           game over = True
       collected crystals = pygame.sprite.spritecollide(player,
energy crystals, True)
      score += len(collected crystals)
```