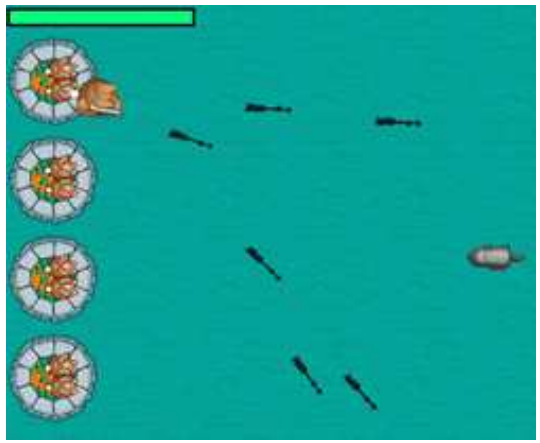
Python i Pygame

Programiranje arkadnih igrica u Pythonu koristeći Pygame

Bunnies and Badgers

Bunnies and Badgers?

- Jednostavna igrica u kojoj zeko brani dvorce od jazavaca
- Zeko se nalazi ispred dvoraca i može se okretati lijevo desno
- Jazavci se kreću prema dvorcima i ako do njih stignu, smanjuju zekinu energiju.
- Zekin cilj je strelicama pogoditi jazavce prije nego što stignu do dvoraca



Osnovna struktura igrice

- #1 - Uključivanje dodatnih modula (biblioteka)

```
#1 - Ukljuci dodatne module
import random, sys, copy, os, pygame
# Eksplicitno ukljuci konstante iz pygame namespace-a
from pygame.locals import *
```

- #2 – Osnovne postavke

```
# 2 - Inicijaliziraj igricu
# Inicijaliziraj pygame
pygame.init()
# Postavi naslov prozora
pygame.display.set_caption('Bunnies and Badgers')
# Definiraj velicinu prozora igrice
width, height = 640, 480
screen=pygame.display.set_mode((width, height))
# Da li izvrsavamo program 0=ne 1=Da
running = 1
# Nacin zavrsetka programa 0=poraz 1=pobjeda
exitcode = 0
```

- #3 – Učitavanje likova

```
# 3 - Ucitaj likove
# Zeko
player = pygame.image.load("bunniesbadgers/images/dude.png")))
```

Osnovna struktura igrice, nastavak

python™

- #4 – Glavna petlja igrice

```
# 4 - Glavna petlja programa
while not done:
    # 5 - Obrisi ekran prije ponovnog crtanja
    screen.fill(0)

    # 6 - Crtanje elemenata ekrana
    # nacrtaj zeku na poziciji 100, 200
    screen.blit(player, (100, 200))

    # 7 - Azuriraj ekran
    pygame.display.flip()

    # 8 - Hvatanje akcije igraca
    for event in pygame.event.get():
        if event.type == pygame.QUIT: # Igrac je zatvorio prozor
            done = True # Kraj programa

# Po izlasku iz petlje zatvori ekran
pygame.quit()
```

Dodavanje grafičkih elemenata

- Dodavanje trave i dvoraca – u dijelu #3 učitaj likove iz datoteka

```
# Dvorac
castle = pygame.image.load("bunniesbadgers/images/castle.png")
# Trava
grass = pygame.image.load("bunniesbadgers/images/grass.png")
```

- Crtanje trave i dvoraca na ekran – u dijelu #6 prije crtanja zeke

```
# Nartaj travu - ponavljaj duz x i y osi ovsno o dimenzijama lika trave
for x in range(int(width/grass.get_width())+1):
    for y in range(int(height/grass.get_height())+1):
        screen.blit(grass,(x*100,y*100))

# Nacrtaj 4 dvorca
screen.blit(castle,(0,30))
screen.blit(castle,(0,135))
screen.blit(castle,(0,240))
screen.blit(castle,(0,345))
```

Pokreni zeku

- Zeku ćemo pokretati tipkama gore, dolje, lijevo i desno
- Dodavanje niza u kojeg ćemo spremati info koje tipke su pritisnute, i varijable u koju ćemo spremati zekinu poziciju – u dijelu #2
  keys[0] – tipka za gore, keys[1] – tipka za lijevo, keys[2] – tipka za dolje
  keys[3] – tipka za desno

```
# Polje za spremanje informacije o pritisnutim tipkama
keys = [False, False, False, False]
# Zekina pozicija
playerpos=[100,200]
```

- Iskoristi varijablu playerpos pri crtanju zeke

```
# Nacrtaj zeku na poziciji playerpos
screen.blit(player, playerpos)
```

Pokreni zeku, nastavak

- Tipke za pomicanje zeke: w – gore, a- lijevo, s – dolje, d - desno
- Detektiraj koje tipka je pritisnuta, i ažuriraj polje keys – u dijelu #8

```
if event.type == pygame.KEYDOWN: # Igrac je pritisnuo tipku
    if event.key==K_w:
        keys[0]=True
    elif event.key==K_a:
        keys[1]=True
    elif event.key==K_s:
        keys[2]=True
    elif event.key==K_d:
        keys[3]=True
if event.type == pygame.KEYUP: # Igrac je otpustio tipku
    if event.key==pygame.K_w:
        keys[0]=False
    elif event.key==pygame.K_a:
        keys[1]=False
    elif event.key==pygame.K_s:
        keys[2]=False
    elif event.key==pygame.K_d:
        keys[3]=False
```

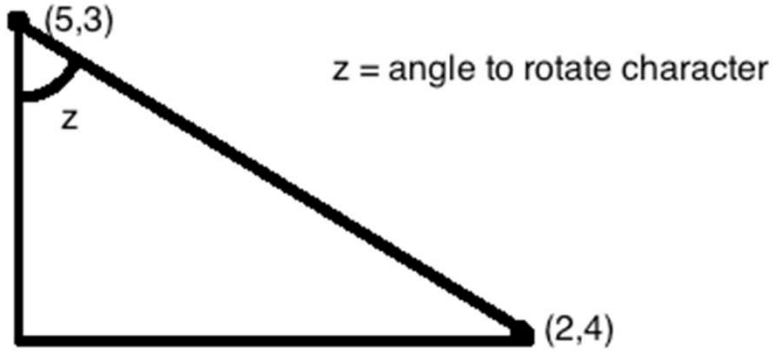
Pokreni zeku, nastavak

- Ovisno o pritisnutoj tipki ažuriraj varijablu playerpos – dio #9 na kraju

```
# 9 - Pomakni zeku
if keys[0]:
    playerpos[1]-=5
elif keys[2]:
    playerpos[1]+=5
elif keys[1]:
    playerpos[0]-=5
elif keys[3]:
    playerpos[0]+=5
```

Rotiraj zeku

- Želimo rotirati zeku pomoću miša – da bi to napravili moramo odrediti kut između trenutne pozicije zeke i pozicije kursora miša. To ćemo napraviti pomoću funkcije atan2(a, b) koja daje kut kojeg tvore a i b. Primjer:



atan2(diff x, diff y) = z
atan2(5-2, 3-4) = z
atan2(3, -1) = z

Rotiraj zeku, nastavak

- Da bi mogli koristiti funkciju atan2 treba uključiti biblioteku math – u dijelu #1

```
# Ukljuci biblioteku math
import math
```

- Pročitaj trenutnu poziciju miša – na kraju dijela #6

```
# Procitaj poziciju misa
position = pygame.mouse.get_pos()
```

x koordinata je u position[0], a y u position[1]

- Pomoću funkcije atan2 izračunaj kut između zeke i trenutne pozicije miša

```
# Izracunaj kut izmedju zeke i misa
angle = math.atan2(position[1]-(playerpos[1]+32),position[0]-(playerpos[0]+26))
```

- Pretvori kut iz radijana u stupnjeve

```
# Pretvori kut iz radijana u stupnjeve
playerrot = pygame.transform.rotate(player, 360-angle*57.29)
```

- Izračunaj novu poziciju zeke i prikazi je na ekranu

```
# Nova pozicija zeke
playerpos1 = (playerpos[0]-playerrot.get_rect().width/2, playerpos[1]-
playerrot.get_rect().height/2)
# Nacrtaj zeku na poziciji playerpos1
screen.blit(playerrot, playerpos1)
```

Zeko puca

- Dodaj polje acc koje prati preciznost igrača (acc[0] – broj pogođenih dabrova, acc[1] – broj ispaljenih strelica) i polje arrows u koje spremamo informacije o svim strelicama – u dijelu #2

```
# Brojac pogodjenih dabrova i ispaljenih strelica
acc = [0,0]
# Info o strelicama
# Za svaku strelicu: Kut izmedju zeke i pozicije misa, x kooordinata zeke, y koordinata zeke
arrows = []
```

- Dodaj sliku strelice – u dijelu #3

```
# Strelica
arrow = pygame.image.load("bunniesbadgers/images/bullet.png")
```

- Kada igrač pritisne lijevu tipku miša, ispali strelicu – u dijelu #8

```
if event.type==pygame.MOUSEBUTTONDOWN: # Igrac je pritisnuo lijevu tipku misa
            # Procitaj poziciju misa
            position=pygame.mouse.get_pos()
            # Povecaj brojac ispaljenih strelica za 1
            acc[1]+=1
            # Za tekucu strelicu zapamti kut izmedju zeke i pozicije misa, x oordinatu
zeke, y koordinatu zeke
            arrows.append([math.atan2(position[1]-(playerpos1[1]+32),position[0]-
(playerpos1[0]+26)),playerpos1[0]+32,playerpos1[1]+32])
```

Zeko puca, nastavak

- Iscrtavaj svaku strelicu sa pomacima x10 pixela dok ne ispadne sa ekrana - u dijelu #6.2

```
# 6.2 Crtanje strelica
    # za svaku strelicu iz polja arrows
    for bullet in arrows:
        index=0
        # Koordinate strelice
        velx=math.cos(bullet[0])*10 # Izracunaj novu x koordinatu
        vely=math.sin(bullet[0])*10 # Izracunaj novu y koordinatu
        # Azuriraj x i y koordinate strelice
        bullet[1]+=velx
        bullet[2]+=vely
        # Ako su koordinate strelice van ekrana, ukloni strelicu iz niza arrows
        if bullet[1]<-64 or bullet[1]>640 or bullet[2]<-64 or bullet[2]>480:
            arrows.pop(index)
        # Idi na slijedeću strelicu
        index+=1
        # Za svaku strelicu iz pola arrows
        for projectile in arrows:
            # Kreiraj slitu strelice zarotiranu za kut u arrows[0]
            arrow1 = pygame.transform.rotate(arrow, 360-projectile[0]*57.29)
            # Nacrtaj sliku zarotirane strelice na koordinatama arrows[1],arrows[2]
            screen.blit(arrow1, (projectile[1], projectile[2]))
```

Jazavci

- Dodaj varijable koje prate jazavce – u dijelu #2

```
# Ukljuci biblioteku math
import math
```

- Dodaj sliku jazavca – u dijelu #3

```
# Jazavac
badguyimg1 = pygame.image.load("bunniesbadgers/images/badguy.png")
badguyimg=badguyimg1
```

Jazavci, nastavak

- Crtanje jazavaca – u dijelu #6.3

```
# 6.3 Crtanje jazavaca
    # Vrijeme tekuceg jazavca je isteklo, kreiraj novog
    if badtimer==0:
        badguys.append([640, random.randint(50,430)]) # koordinate x=640 50<y<430
# Postavi vrijeme trajanja novog jazavca
        badtimer=100-(badtimer1*2)
        # Povecavaj frekvenciju kreiranja novih jazavaca do 7.
        if badtimer1>=35:
            badtimer1=35
        else:
            badtimer1+=5
    index=0
    # Za svakog jazavca
    for badguy in badguys:
        # Ako je jazavac izasao sa ekrana ukloni ga sa liste
        if badguy[0]<-64:
            badguys.pop(index)
        # Inace ga pomakni u lijevo za 7 pixela
        badguy[0]-=7
        index+=1
    # Prikazi sve jazavce na ekranu
    for badguy in badguys:
        screen.blit(badguyimg, badguy)
```

Jazavci napadaju dvorce

- Napad jazavaca na dvorce – u dijelu #6.3.1

```
 # 6.3.1 - Jazavci napadaju dvorce
        # Prociraj dimenzije pravokutnika koji omedjuje slika jazavca
        badrect=pygame.Rect(badguyimg.get_rect())
        # Procitaj x i y koordinate jazavca
        badrect.top=badguy[1]
        badrect.left=badguy[0]
        # Ako se jazavac sudario sa dvorcem (x<64), smanji energiju igraca i
ukloni jazavca
        if badrect.left<64:
            healthvalue -= random.randint(5,20)
            badguys.pop(index)
```

Strelice ubijaju jazavce

- Kad strelica pogodi jazavca ukloni jazavca i strelicu – u dijelu #6.3.2

```
#6.3.2 - Strelice ubijaju jazavce
        index1=0
        # Za svaku strelicu
        for bullet in arrows:
            # Prociraj dimenzije pravokutnika koji omedjuje sliku strelice
            bullrect=pygame.Rect(arrow.get_rect())
            # Procitaj x i y koordinate strelice
            bullrect.left=bullet[1]
            bullrect.top=bullet[2]
            # Ako se strelica i jazavac preklapaju
            if badrect.colliderect(bullrect):
                # Uvecaj brojac pogodjenih dabrova za 1
                acc[0]+=1
                # Ukloni tekuceg dabra i strelicu
                badguys.pop(index)
                arrows.pop(index1)
            index1+=1
```

Sat

- Prikaži u gornjem desnom uglu ekrana sat koji odbrojava 90s od startanja programa– u dijelu #6.3.2

```
 # 6.4 – Nacrtaj sat
    # Definiraj font za sat
    font = pygame.font.Font(None, 24)
    # Definiraj tekst za ispis
    survivedtext = font.render("Time: "+'{:02d}'.format((90000-
pygame.time.get_ticks())//1000)+" s", True, (0,0,0))
    # Procitaj dimenzije pravokutnika koji omedjuje tekst
    textRect = survivedtext.get_rect()
    # Postavi koordinate gornjeg desnog vrha pravokutnika za tekst
    textRect.topright=[635,5]
    # Nacrtaj tekst na ekranu
    screen.blit(survivedtext, textRect)
```

Energija igrača

- U gornjem lijevom uglu ćemo nacrtati stupac koji prikazuje energiju igrača
- Prvo dodajemo slike stupca i prikaza energije – u dijelu #3

```
# Stupac energije i energija
healthbar = pygame.image.load("bunniesbadgers/images/healthbar.png")
health = pygame.image.load("bunniesba
```

- Zatim prikazujemo trenutnu energiju igraca – u dijelu #6.5

```
# 6.5 - Prikazi energiju igraca
    # Prvo nacrtaj puni crveni stupac energije
    screen.blit(healthbar, (5,5))
    # Zatim nacrtaj zeleni stupac preko crvenog ovisno o trenutnoj energiji igraca
    for health1 in range(healthvalue):
        screen.blit(health, (health1+8,8))
```

Kraj igrice

- Igrica završava kad energija igrača padne ne nulu ili kad istekne 90 sekundi
  Ako je isteklo 90 sekundi a energija igrača nije pala na nulu, igrač je pobijedio
  Ako je unutar 90 sekundi energija igrača pala na nulu, igrač je izgubio
- Prvo dodajemo slike za kraj igre i pobjedu – u dijelu #3

```
# Kraj igre i pobjeda
gameover = pygame.image.load("bunniesbadgers/images/gameover.png")
youwin = pygame.image.load("bunniesbadgers/images/youwin.png")
```

- Zatim prikazujemo trenutnu energiju igraca – u dijelu #6.5

```
# 6.5 - Prikazi energiju igraca
    # Prvo nacrtaj puni crveni stupac energije
    screen.blit(healthbar, (5,5))
    # Zatim nacrtaj zeleni stupac preko crvenog ovisno o trenutnoj energiji igraca
    for health1 in range(healthvalue):
        screen.blit(health, (health1+8,8))
```

Kraj igrice, nastavak

- Provjeri da li je igrač pobijedio ili izgubio – u dijelu #10

```
#10 - Provjera da li je igrac pobjedio ili izgubio
    # Ako je proslo 90s igra je zavrsena
    if pygame.time.get_ticks()>=90000:
        running=0
        exitcode=1
    # Ako je energija igraca pala na 0 izgubio je
    if healthvalue<=0:
        running=0
        exitcode=0
    # Azuriraj preciznost igraca
    if acc[1]!=0:
        accuracy=acc[0]*1.0/acc[1]*100
    else:
        accuracy=0
```

Kraj igrice, nastavak

- Ovisno o tome da li je igrač pobijedio ili izgubio prikaži završni ekan – u dijelu #11

```
# 11 - Zavrsni ekran win/lose
if exitcode==0:
    # Poraz
    pygame.font.init()
    font = pygame.font.Font(None, 24)
    text = font.render("Accuracy: "+str(accuracy)+"%", True, (255,0,0))
    textRect = text.get_rect()
    textRect.centerx = screen.get_rect().centerx
    textRect.centery = screen.get_rect().centery+24
    screen.blit(gameover, (0,0))
    screen.blit(text, textRect)
else:
    # Pobjeda
    pygame.font.init()
    font = pygame.font.Font(None, 24)
    text = font.render("Accuracy: "+str(accuracy)+"%", True, (0,255,0))
    textRect = text.get_rect()
    textRect.centerx = screen.get_rect().centerx
    textRect.centery = screen.get_rect().centery+24
    screen.blit(youwin, (0,0))
    screen.blit(text, textRect)
```

Kraj igrice, nastavak

- Na kraju zavrti petlju koja će osigurati da završni ekran ostane prikazan dok igrač ne zatvori prozor

```
# Petlja koja osigurva prikaz zavrsnog ekrana dok igrac ne zatvori prozor
while 1:
    # Uhvati akciju igraca
    for event in pygame.event.get():
        if event.type == pygame.QUIT: # Igrac je zatvorio prozor
            # Zavrsi program
            pygame.quit()
            sys.exit()
    # Azuriraj ekran
    pygame.display.flip()
```