Kurs: Pajton - programiranje igara sa Pygame bibliotekom
009_010 čas: pygame_state_base_programming
Teme: program Pita.py

001 Redosled događaja

Pygame registruje sve događaje (events) prouzrokovane od strane korisnika i smešta ih u redosled događaja (event queue).
Redosledu događaja se pristupa preko funkcije pygame.event.get().
Svaki element u ovom redosledu je objekat događaja.
Svaki objekat događaja mora da ima atribut type i neki drugi atribut koji je poseban baš za taj događaj.
Atribut type je celobrojno predstavljanje tipa događaja.
U pygame modulu postoje predefinisane celobrojne konstante koje predstavljaju određeni tip događaja.

002 Upravljanje događajem

Za upravljanje događajem potrebno je samo proći kroz petlju sa spiskom svih mogućih događaja.
Tako se proverava koji tip događaja se desio (poznavajući predefinisane celobrojne konstante u pygame modulu).
Na osnovu tih informacija se izvodi određena akcija.
U dosadašnjim kodovima uvek je postojalo reagovanje koda na pritisak bilo kakvog dugmeta tastature ili klika na dugme za gašenje prozora igre.

003 Događaji tastature

Postoje dva tipa događaja koji reaguju na upotrebu tastature u igri: KEYDOWN i KEYUP.
Ovi događaji imaju atribut key koji je celobrojno predstavljanje dugmeta na tastaturi.
Pygame modul ima predefinisane celobrojne konstante kojima se predstavljaju svi tasteri.
Konstante se ispisuju sa početnim velikim K, donjom crtom (_, underscore) i nazivom dugmeta.
Primeri: <- je K_BACKSPACE, a je K_a, F4 je K_F4.
U igri se ispituje da li je pritisnuto pa pušteno određeno dugme, što se kontroliše tipom događaja KEYUP:

```
elif dogadjaj.type == KEYUP:
	if dogadjaj.key == pygame.K_ESCAPE:
		sys.exit()
	elif dogadjaj.key == pygame.K_1:
		deo1 = True
	elif dogadjaj.key == pygame.K_2:
		deo2 = True
```

Pritiskom na dugme 1, 2, 3 ili 4, odgovarajuće parče pite se crta.
Kada se nacrtaju sva četiri parčeta pite, cela pita menja boju kao i svi napisani brojevi na njoj.

004 Programiranje zasnovano na stanju

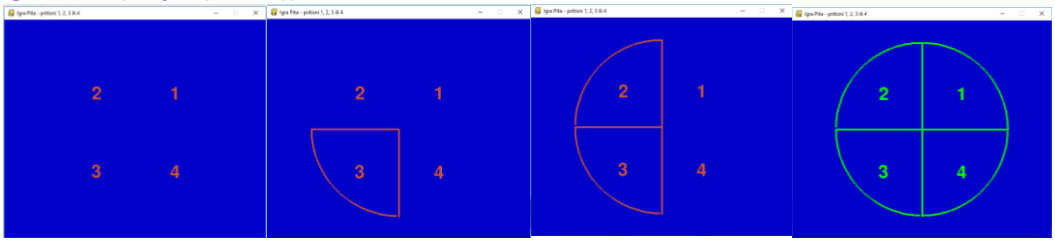
Ovde se koristi state-base programming (programiranje zasnovano na stanju).
Četiri dela pite se se crtaju automatski kada igrač pritisne odgovarajuće dugme.
Umesto toga, state flag (pokazivač (fleg) stanja) se postavlja (setuje) kada je dugme pritisnuto pa pušteno, i taj fleg se kasnije koristi za crtanje delova pite bazirano na tom flegu.

Ovaj koncept ukazuje kako se rukuje događajima i kako igračeva interakcija utiče na njih indirektno.

program Pita.py

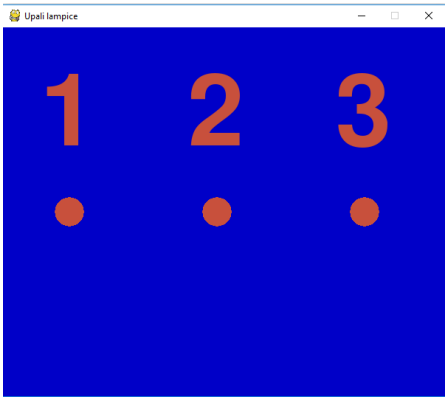
```
import math, pygame, sys
from pygame.locals import *
pygame.init()
ekran = pygame.display.set_mode((600,500))
pygame.display.set_caption("Igra Pita - pritisni 1, 2, 3 ili 4")
moj_font = pygame.font.Font(None, 60)
boja = 200, 80, 60
debljina_linije = 4
x = 300
y = 250
poluprecnik = 200
pozicija = x - poluprecnik, y - poluprecnik, poluprecnik * 2, poluprecnik * 2
deo1 = False
deo2 = False
deo3 = False
deo4 = False
while True:
    for dogadjaj in pygame.event.get():
        if dogadjaj.type == QUIT:
            sys.exit()
        elif dogadjaj.type == KEYUP:
            if dogadjaj.key == pygame.K_ESCAPE:
                sys.exit()
            elif dogadjaj.key == pygame.K_1:
                deo1 = True
            elif dogadjaj.key == pygame.K_2:
                deo2 = True
            elif dogadjaj.key == pygame.K_3:
                deo3 = True
            elif dogadjaj.key == pygame.K_4:
                deo4 = True
    #ciscenje ekrana
    ekran.fill((0,0,200))
    #crtanje cetiri broja
    slika1 = moj_font.render("1", True, boja)
    ekran.blit(slika1, (x + poluprecnik / 2 - 20, y - poluprecnik / 2))
    slika2 = moj_font.render("2", True, boja)
    ekran.blit(slika2, (x - poluprecnik / 2, y - poluprecnik / 2))
    slika3 = moj_font.render("3", True, boja)
    ekran.blit(slika3, (x - poluprecnik / 2, y + poluprecnik / 2 - 20))
    slika4 = moj_font.render("4", True, boja)
    ekran.blit(slika4, (x + poluprecnik / 2 - 20, y + poluprecnik / 2 - 20))
    #odlucivanje o crtanju dela pite
    if deo1:
        ugao_start = math.radians(0)
        ugao_kraj = math.radians(90)
        pygame.draw.arc(ekran, boja, pozicija, ugao_start, ugao_kraj, debljina_linije)
        pygame.draw.line(ekran, boja, (x, y), (x, y - poluprecnik), debljina_linije)
        pygame.draw.line(ekran, boja, (x, y), (x + poluprecnik, y), debljina_linije)
    if deo2:
        ugao_start = math.radians(90)
        ugao_kraj = math.radians(180)
        pygame.draw.arc(ekran, boja, pozicija, ugao_start, ugao_kraj, debljina_linije)
        pygame.draw.line(ekran, boja, (x, y), (x, y - poluprecnik), debljina_linije)
        pygame.draw.line(ekran, boja, (x, y), (x - poluprecnik, y), debljina_linije)
    if deo3:
        ugao_start = math.radians(180)
        ugao_kraj = math.radians(270)
```

```
        pygame.draw.arc(ekran, boja, pozicija, ugao_start, ugao_kraj, debljina_linije)
        pygame.draw.line(ekran, boja, (x, y), (x - poluprecnik, y), debljina_linije)
        pygame.draw.line(ekran, boja, (x, y), (x, y + poluprecnik), debljina_linije)
    if deo4:
        ugao_start = math.radians(270)
        ugao_kraj = math.radians(360)
        pygame.draw.arc(ekran, boja, pozicija, ugao_start, ugao_kraj, debljina_linije)
        pygame.draw.line(ekran, boja, (x, y), (x, y + poluprecnik), debljina_linije)
        pygame.draw.line(ekran, boja, (x, y), (x + poluprecnik, y), debljina_linije)
    #odlucivanje da li je pita gotova
    if deo1 and deo2 and deo3 and deo4:
        boja = 0,255,0
    pygame.display.update()
```

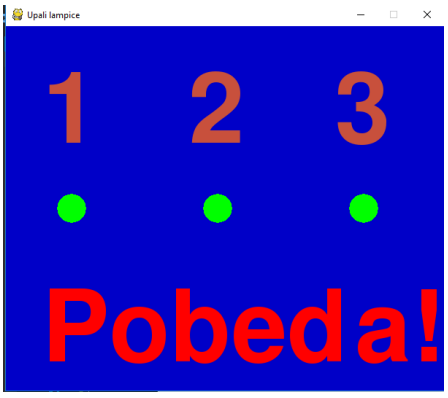


Zadaci

010) Modifikovati kod programa Pita.py tako da početni ekran igre Upali_lampice.py izgleda ovako:



Kada igrač pritisne bilo kojim redosledom tastere 1, 2 i 3, treba da se na ekranu pokaže:



011) Modifikovati kod programa Upali_lampice.py tako da ako igrač upali lampicu 2 na ekranu se pojavi poruka o pobedi i kraju igre. Ako igrač upali bilo koju drugu lampicu, dobija se poruka o grešci i kraju igre.