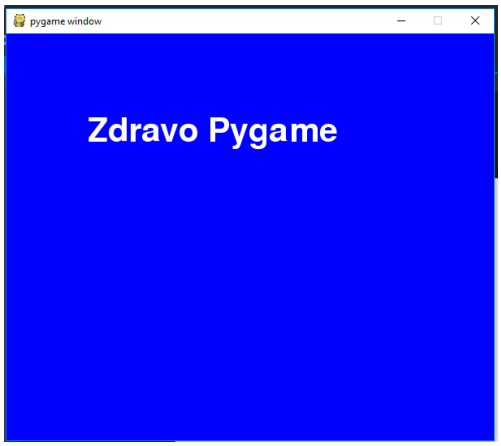
Kurs: Pajton - programiranje igara sa Pygame bibliotekom
003_004 čas: pygame_crtanje_teksta
Teme: program Pozdrav.py, pygame biblioteka, crtanje teksta, main game loop, program Crtanje_Linije.py

program Pozdrav.py

```
import pygame, sys
from pygame.locals import *
pygame.init()
ekran = pygame.display.set_mode((600,500))
moj_font = pygame.font.Font(None,60)
bela = 255,255,255
plava = 0,0,255
tekst_slika = moj_font.render("Zdravo Pygame", True, bela)
ekran.fill(plava)
ekran.blit(tekst_slika, (100,100))
pygame.display.update()
while True:
    for dogadjaj in pygame.event.get():
        if dogadjaj.type in (QUIT, KEYDOWN):
            sys.exit()
    ekran.fill(plava)
    ekran.blit(tekst_slika, (100,100))
    pygame.display.update()
```



001 Pygame Bbiblioteka

Biblioteka za programiranje igara se naziva Pygame i razvijena je da bi se crtala grafika, koristio korisnički ulaz, animacija i koristio tajmer koji čini da se igra odvija u konstantnom frejm rejtu. Pygame omogućava i rad sa audio plejbekom i korišćenje miša i tastature.

002 Importovanje pygame i sys modula

```
import pygame, sys
```

Prvi korak je import pygame i sys modula u Pajton program a time se omogućava da naš program koristi funkcije koje se nalaze u ovim modulima.

Sve Pygame funkcije koje rade sa grafikom, zvukom i drugim mogućnostima se nalaze u pygame modulu.

Istovremeno sa importovanjem pygame modula, importuju se svi ostali moduli koji su unutar pygame modula, poput pygame.images i pygame.mixer.music.

Zato nema potrebe posebno importovati module unutar ovog modula.

003 Importovanje konstanti iz pygame.locals funkcije

```
from pygame.locals import *
```

Sledeće je import svih konstanti u Pygame tako da su spremne za korišćenje u kodu.

I ova linija predstavlja import naredbu ali koristi drugačiji format: from ime_modula import *. Obično kada se poziva funkcija koja je u ovom modulu, koristi se format: ime_modula.ime_funkcije(), posle importovanja modula.

Ali, ako se koristi ovakav drugačiji format import naredbe, može se preskočiti korišćenje ime_modula i samo upotrebljavati ime_funkcije() (kao kada se koriste ugrađene funkcije u Pajtonu).

Ovaj format se koristi zato što pygame.locals sadrži nekoliko konstanti promenjivih koje se lako identifikuju kao deo pygame.locals modula pa i nema potrebe kucati taj deo importa. Ipak, za sve ostale module je bolje koristiti klasičan format importovanja.

004 Inicijalizacija importovanih Pygame modula

```
pygame.init()
```

U liniji `pygame.init()` je poziv funkcije koji se mora obaviti posle importovanja pygame modula ali pre pozivanja bilo koje druge Pygame funkcije.

Ovime se inicijalizuju svi importovani pygame moduli.

005 Kreiranje objekta za prozor

```
ekran = pygame.display.set_mode((600,500))
```

Sledeće je pristup sistemu prikaza i kreiranje prozora.

Funkcija `pygame.display.set_mode()` vraća pygame.Surface objekat za prozor i koristi se za predstavljanje bilo koje slike (image).

U zagradi ((600, 500)) je torka koja govori funkciji set_mode() koje su dimenzije prozora u pikselima.

Pišu se duple zagrade pošto u funkcijsku zagradu se smešta podatak kao jedna torka sa dva elementa.

U promenjivu ekran se smešta vraćeni pygame.Surface objekat.

006 Crtanje teksta

Pygame podržava izlaz teksta na grafički prozor korišćenjem pygame.font.

Da bi se crtao tekst, prvo se mora kreirati font objekat:

```
moj_font = pygame.font.Font(None,60)
```

Parametri konstruktora funkcije pygame.font.Font() su string koji predstavlja naziv fonta koji će se koristiti i ceo broj koji predstavlja veličinu fonta za prikaz.

Naziv TrueType fonta se može predati u pygame.font.Font() konstruktor, poput „Arial“ ali korišćenjem None izaziva da se koristi difolt Pygame font.

Tekst se u Pygame mora prvo renderovati na površinu koja se zatim crta na ekranu.

Pošto je ovo proces koji traje neko vreme, lakše je prvo kreirati tekst površinu (ili sliku) u memoriji a zatim nacrtati tekst kao sliku.

Ako se tekst neće menjati, bolje je prerenderovati tekst u sliku.

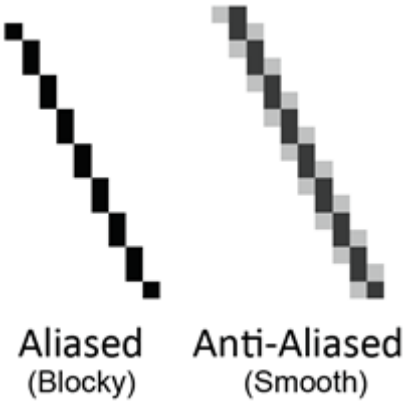
007 Kreiranje objekta sa nacrtanim tekstom

```
bela = 255,255,255
plava = 0,0,255
tekst_slika = moj_font.render("Zdravo Pygame", True, bela)
```

Sa poslednjom linijom se kreira pygame.Surface objekat tekst_slika sa nacrtanim tekstom na njoj.

To se izvodi pozivanjem objektove Font metode render().

Prvi parametar je string teksta koji će se rederovati, drugi parametar je Bulova vrednost vezana za anti-aliasing, treći parametar je boja teksta (RGB vrednost) a poslednji je boja pozadine (ako je nema uzima se providna pozadina po difoltu).
Anti-aliasing je grafička tehnika kojom kreirani tekstova i oblici izgledaju manje kockasti, tako što se na njihove ivice dodaje blago zamućenje.
Ovom tehnikom se povećava vreme potrebno za iscrtavanje oblika.



008 Kreiranje objekta sa nacrtanim tekstom

Da bi se nacrtao tekst, obično se koristi proces čišćenja ekrana, crtanja a zatim osvežavanje ekrana:

```
ekran.fill(plava)
ekran.blit(tekst_slika, (100,100))
pygame.display.update()
```

Objekat tekst_slika je zapravo površina na koju se crta sa screen.blit().
Ako bi se u ovom momentu startovao program, video bi se prozor za tren a odmah i nestao pošto nije definisano odlaganje (delay) prikaza.
Metoda fill() nije funkcija već metoda pygame.Surface objekata.
Ona će popuniti ceo objekat sa vrednošću parametra koji predstavlja željenu boju.
Funkcija blit() crta sadržaj jednog Sufrace objekta u drugi Surface objekat.
U ovom slučaju se kopira objekat tekst_slika u objekat ekran.
Pri tome se koriste dva parametra, prvi je izvor Surface objekta koji će biti kopiran u drugi objekat; drugi parametar je torka sa dva cela broja za x i y vrednosti (u odnosu na gornji levi ugao drugog objekta).

009 Kontrolisana petlja

Postoje dva problema: nestala slika se samo jednom pojavila pa je nestala i nema mogućnosti unosa od strane korisnika.
Da bi se rešio prvi problem, koristiće se u kodu while petlja koja će realizovati kod sve dok je uslov tačan: while True:.
Sledeće, kreiraće se menadžer događaja (event handler).
Potrebno je da prozor ostane na ekranu sve dok korisnik ga ne zatvori.
Događaj zatvaranja prozora će biti klik na X u gornjem desnom uglu prozora, ili samo pritisak na bilo koje dugme tastature.

010 Main Game Loop

```
for dogadjaj in pygame.event.get():
        if dogadjaj.type in (QUIT, KEYDOWN):
            sys.exit()
```

U većini pygame igara koristi se main game loop (petlja glavnog dela igre).

Unutar ove petlje se odigravaju tri dela koda naizmenično: upravljanje (menadžment) događajima (event handling), updejtovanje stanja igre, crtanje stanja igre na ekranu.

Stanje igre predstavlja set vrednosti za sve promenjive u kodu.
Bilo kakva promena u vrednostima promenjivih u kodu igre se naziva promena stanja igre (game state).
Izlazak iz igre ili pauziranje igre sprečavaju da se promeni stanje igre.
Stanje igre se updejtuje kao odgovor na događaje (klik mišem ili pritisak na tastaturi) ili protok vremena, dok petlja igre stalno proverava da li je došlo do bilo kakve promene u stanju igre.
Unutar petlje glavnog dela igre postoji kod koji proverava koji su se događaji desili (u Pygame to se izvodi pozivanjem funkcije pygame.event.get()).
Ova petlja takođe ima kod koji updejtuje stanje igre u zavisnosti od događaja u igri i to se naziva menadžment događajima.
Svaki put kada korisnik uradi neku od akcija tokom igre, poput pritiska tastature ili pomeranje mišem unutar programskog prozora, kreira se pygame.event.ime_dogadjaja objekat unutar Pygame bibiloteke za pamćenje ovakvog događaja.
Ovo je tip objekta koji postoji u event modulu, koji je unutar pygame modula.
Može se otkriti koji se to događaj desio pozivanjem funkcije pygame.event.get(), koja vraća listu pygame.event.ime_dogadjaja objekata (ponekad se kratko nazivaju Event objekti).
Ova lista Event objekata sadrži svaki događaj koji se desio od poslednjeg pozivanja pygame.event.get() funkcije.
U kodu, for petlja će iterirati preko liste Event objekata koje vraća pygame.event.get().
Na svakoj iteraciji promenjiva dogadjaj će dobiti vrednost sledećeg Event objekta sa liste.
Redosled događaja na listi je prema redosledu dešavanja događaja u igri.
Ako se nije desio nikakav događaj, vraća se prazna lista.
Event objekti imaju člana promenjivu (atributi ili osobenosti, properties) koja se naziva type, koja govori koji tip događaja objekat predstavlja.
Pygame ima konstantne promenjive za svaki od mogućih tipova u pygame.locals modulima.
Linija `if dogadjaj.type in (QUIT, KEYDOWN):` proverava da li je Event objekat type identičan sa konstantama QUIT ili sa KEYDOWN.
Ako jeste, desiće se poziv funkcije sys.exit() za prekid programa.
Konstanta QUIT označava klik na X prozora, dok KEYDOWN označava pritisak na bilo koje dugme tastature.
U ovom slučaju, ne postoji kod za menadžera događaja, pošto se ne proveravaju dešavanja ostalih mogućih događaja.

Na kraju petlje se dodaje kod za crtanje (nije neophodan pošto nema promena na ekranu) i osvežavanje ekrana:

```
ekran.fill(plava)
ekran.blit(tekst_slika, (100,100))
pygame.display.update()
```

011 Crtanje linija

Prave se mogu crtati korišćenjem pygame.draw.line() funkcije.

Crtanje linija je donekle kompleksnije od crtanja drugih oblika, pošto se moraju navesti i početna pozicija i krajnja pozicija linije.

Crta se linija u bilo kojoj boji i sa željenom debljinom.

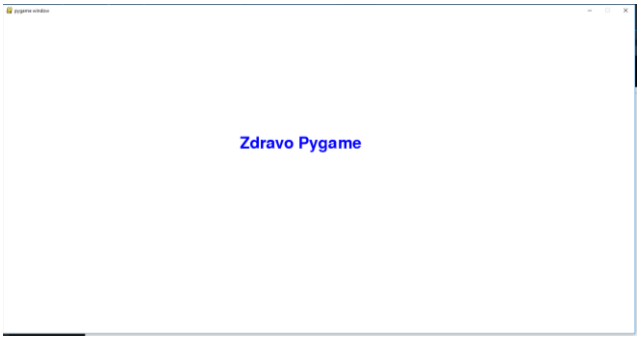
program Crtanje_Linije.py

```
import pygame, sys
from pygame.locals import *
pygame.init()
ekran = pygame.display.set_mode((600,500))
pygame.display.set_caption("Crtanje Linije")
while True:
    for dogadjaj in pygame.event.get():
        if dogadjaj.type in (QUIT, KEYDOWN):
            sys.exit()

    ekran.fill((0,80,0))
    #crta liniju
    boja = 100,255,200
    debljina_linije = 8
    pygame.draw.line(ekran, boja, (100,100), (500,400), debljina_linije)
    pygame.display.update()
```

Zadaci

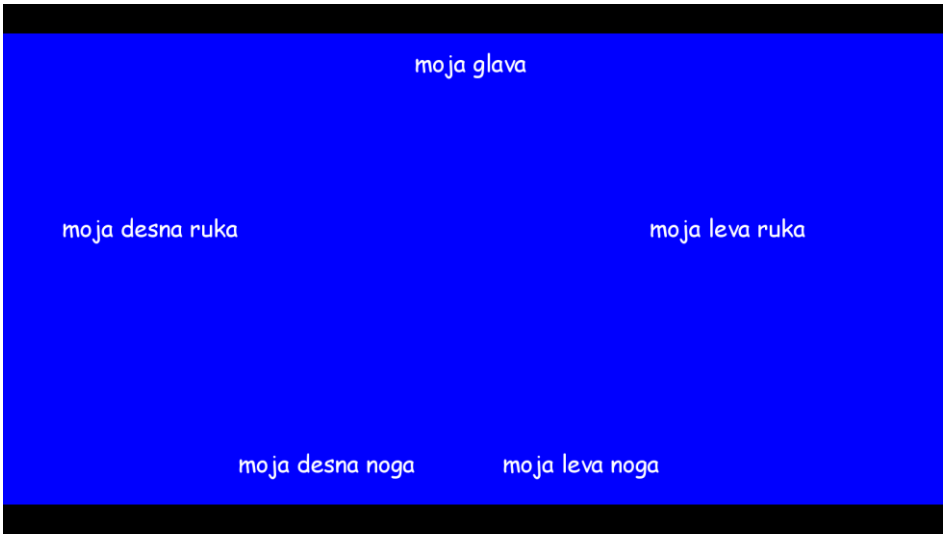
01) Izmeniti kod programa Pozdrav.py tako da pozdravna poruka bude na objektu prozor dimenzija 1600x800, tekst objekat plave boje na beloj pozadini na sredini prozora.



02) Program Zadatak_002.py realizuje prikaz prozora na celom ekranu:

```
import pygame, sys
from pygame.locals import *
pygame.init()
ekran = pygame.display.set_mode((1600,800), pygame.FULLSCREEN)
moj_font1 = pygame.font.SysFont("comicsansms",40)
moj_font2 = pygame.font.SysFont("calibribold",40)
bela = 255,255,255
plava = 0,0,255
tekst_slika1 = moj_font1.render("Ovo je gornji levi ugao", True, bela)
tekst_slika2 = moj_font2.render("Ovo je donji desni ugao", True, bela)
ekran.fill(plava)
ekran.blit(tekst_slika1, (20,20))
ekran.blit(tekst_slika2, (1400,700))
pygame.display.update()
while True:
    for dogadjaj in pygame.event.get():
        if dogadjaj.type in (QUIT, KEYDOWN):
            sys.exit()
    ekran.fill(plava)
    ekran.blit(tekst_slika1, (20,20))
    ekran.blit(tekst_slika2, (1250,700))
    pygame.display.update()
```

Izmeniti kod programa tako da se na ekranu pojavi sledeći prozor:



03) Modifikovati program Crtanje_Linije.py tako da se crta 100 linija sa slučajno izabranim početnim i krajnjim pozicijama.