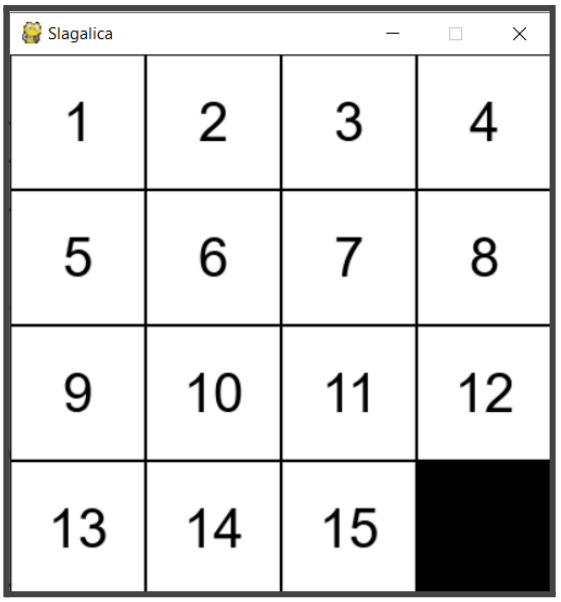
****

Igrica 2

****



OPIS IGRE

Igra predstavlja logičku slagalicu koja se slaže pomeranem pločica. Cilj je da se poredaju brojevi redom od 1 do 15. Crna rupa predstavlja slobodno mesto gde se nešto može da pomeri.

DISKUSIJA ZA IMPLEMENTACIJU

- Slagalica je 2d niz u kome se čuvaju brojevi na pravom mestu
- Pravljenje pločica je zapravo crtanje rectanglea i dodavanje rendera teksta sa brojem
- Mešanje slagalice se odvija kao da se random pomeri rupa u 1 od 4 dozvoljena smera i to se uradi 100 puta
- Pomeranje rupe zahteva zamenu rupe i polja
- Potrebno je da proverimo da li je slagalica složena tako što ispitujemo da li se na svakom polju slagalice pojavljuje broj koji je inicijalno tu napisan
- Potrebno je napraviti poruku za ispis pobede kada se slagalica složi
- Potrebno je za obradu događaja da se prati mouse button down event po type-u

IMPLEMENTACIJA:

```
#korak 1 - podešavanje prozora i neophodnih promenljivih

import pygame as pg
import pygamebg as pgbg

(sirina, visina) = (400, 400)
prozor = pgbg.open_window(sirina, visina, "Slagalica")

DIM = 4                          # dimenzija slagalice je 4x4
SIR_KOLONE = sirina // DIM  # širina kolone u pikselima
VIS_VRSTE = visina // DIM   # širina vrste u pikselima

#napravi slagalicu

#prikazi slagalicu

#igraj slagalicu
```

```
pgbg.wait_loop()

# korak 2 - pravljenje 2d niza brojeva za slagalicu
import pygamebg as pgbg

def napraviSlagalicu():
    rez = []
    for v in range(DIM):
        rez.append([])
        for k in range(DIM):
            rez[v].append(DIM*v + k + 1)
    # inicijalno slagalicu popunjavamo redom brojevima od 1 do n*n
    return rez

(sirina, visina) = (400, 400)

# njen poziv

#napravi slagalicu
slagalica = napraviSlagalicu()
print(slagalica)

# korak 3 - funkcija za prikaz slagalice

def prikazi_slagalicu():
    prozor.fill(pg.Color("white"))        # bojimo pozadinu u belo
    font = pg.font.SysFont("Arial", 40)    #font za brojeve
    # prolazimo redom kroz sva polja
    for v in range(DIM):
      for k in range(DIM):
        #okružujući pravougaonik tekućeg polja
        rect = (k*SIR_KOLONE, v*VIS_VRSTE, SIR_KOLONE, VIS_VRSTE)
        pg.draw.rect(prozor,(0,0,0),rect,1) #crtamo okvir polja
        #centar okvira polja
        (xc,yc)=(k*SIR_KOLONE+SIR_KOLONE//2,v*VIS_VRSTE+VIS_VRSTE//2)
        tekst = font.render(str(slagalica[v][k]), True,(0,0,0))
        # ispisujemo centriran tekst
        (sir_t, vis_t) = (tekst.get_width(), tekst.get_height())
        prozor.blit(tekst, (xc - sir_t//2, yc - vis_t//2))

# njen poziv
#prikazi slagalicu
prikazi_slagalicu()
```

```
#igraj slagalicu
```

SLEDECE ZA IMPLEMENTACIJU

- Dodavanje rupe
- Mešanje slagalice
- Algoritmi za mešanje
    - random.shuffle () ali on bi nam promešao spoljašnju listu
    - Klasični algoritam za shufle, kada se random 2 elementa promene
    - 100 poteza u random smeru

IMPLEMENTACIJA

```
# korak 1 - dodavanje crne rupe

#promenljive za rupu, treba da se nalazi u startu u donjem desnom uglu
(rupa_v, rupa_k) = (DIM-1, DIM-1)
prikazi_slagalicu()

#adaptacija iscrtavanja slagalice u zavisnosti gde je rupa
def prikazi_slagalicu():
    …
        #okružujući pravougaonik tekućeg polja
        rect = (k*SIR_KOLONE, v*VIS_VRSTE, SIR_KOLONE, VIS_VRSTE)
        if v == rupa_v and k == rupa_k:   #ako je na polju rupA
           pg.draw.rect(prozor,(0,0,0),rect)# crtamo crni kvadrata
        else:
           pg.draw.rect(prozor,(0,0,0),rect,1) #crtamo okvir polja
           #centar okvira polja
…
#istestirati kako izgleda i kako ide prikazivanje

#korak 2 - potrebno je da se slagalica promeša

# nasumično mešamo slagalicu
def promesaj_slagalicu():
     global rupa_v, rupa_k, slagalica # globalne promenljive koje ćemo
menjati
     smerovi = [(-1, 0), (1, 0), (0, -1), (0, 1)]# četiri moguća smera
pomeranja
    max_broj_pomeranja = 100   # mešamo tako što rupu pomerimo 100 puta
    broj_pomeranja = 0         # broj izvršenih pomeranja
    while broj_pomeranja < max_broj_pomeranja:
       #nasumično biramo smer u kojem se pomera rupa
       (smer_v, smer_k) = smerovi[random.randint(0, 3)]
```

```
        #određujemo novu potencijalnu poziciju rupe
        (polje_v, polje_k) = (rupa_v + smer_v, rupa_k + smer_k)
        if pomeri_rupu(polje_v, polje_k): #pomeramo rupu na novo polje
             broj_pomeranja += 1

# pokušavamo da pomerimo rupu na polje_v, polje_k
def pomeri_rupu(polje_v, polje_k):
    global rupa_v, rupa_k, slagalica # globalne promenljive koje menjamo
    if (0 <= polje_v and polje_v < DIM and 0 <= polje_k and polje_k <
DIM):
       # polje mora biti unutar slagalice
       (dv, dk) = (abs(polje_v - rupa_v), abs(polje_k - rupa_k))
       # polje mora biti susedno rupi
       if (dv == 1 and dk == 0) or (dv == 0 and dk == 1):
           # razmenjujemo broj na polju i broj na rupi
            (slagalica[rupa_v][rupa_k], slagalica[polje_v][polje_k]) =
(slagalica[polje_v][polje_k], slagalica[rupa_v][rupa_k])
            (rupa_v, rupa_k) = (polje_v, polje_k)   # pomeramo rupu
             return True                           # uspešno smo pomerili
rupu
    return False

# njen poziv
#prikazi slagalicu
promesaj_slagalicu()
prikazi_slagalicu()

# testiranje za vece dimenzije

# u tom slucaju broj 100 ne bi trebalo da bude fiksiran, nego u funkciji
od DIM
max_broj_pomeranja = DIM*25
```

SLEDECE DA SE NAPRAVI

- Provera složenosti slagalice
- Iscrtavanje poruke za kraj
- Hvatanje klika miša i zamena polja i rupe

IMPLEMENTACIJA

```
#korak 1 - provera kraja kada je slagalica složena

# provera da li je slagalica složena ispravno
def slagalica_slozena():
```

```
    # proveravamo da li postoji polje na kom se nalazi pogrešan broj
    for v in range(DIM):
        for k in range(DIM):
            if slagalica[v][k] != v*DIM+k+1:
                return False  # broj na polju[v][k] je pogrešan
    return True  # nismo naišli na pogrešan broj

def prikazi_cestitku():
    prozor.fill(bela) # bojimo pozadinu prozora u belo
    font = pg.font.SysFont("Arial", 60) # font kojim će biti tekst
    poruka = "Bravo složio si!"         # poruka koja će se ispisivati
    tekst = font.render(poruka, True, crna)
    # određujemo veličinu tog teksta
    (sir_teksta, vis_teksta) = (tekst.get_width(), tekst.get_height())
    # položaj određujemo tako da tekst bude centriran
    (x, y) = ((sirina - sir_teksta) / 2, (visina - vis_teksta) / 2)
    prozor.blit(tekst, (x, y))

# njen poziv

def crtaj():
      if not slagalica_slozena(): # slagalica još nije složena, pa je
prikazujemo
        prikazi_slagalicu()
    else:  # slagalica je složena uspešno, pa čestitamo igraču
        prikazi_cestitku()

# poziv funkcije crtaj() umesto wait_loopa ide
pgbg.event_loop(crtaj, obradi_dogadjaj)

def obradi_dogadjaj(dogadjaj):
    global v, k, slagalica
    if slagalica_slozena():
          # ako je slagalica složena, ne obrađujemo više događaje
        return
    if dogadjaj.type == pg.MOUSEBUTTONDOWN:
        (x, y) = dogadjaj.pos           # koordinate na koje je kliknuto
        v = y // VISINA_VRSTE           # vrsta na koju je kliknuto
        k = x // SIRINA_KOLONE          # kolona na koju je kliknuto
        if pomeri_rupu(v, k):           # ako pomerimo rupu na to polje
            return True                      # treba ponovo da crtamo
    return False                             # ne treba ponovo da crtamo
```