УНИВЕРЗИТЕТ “СВ. КИРИЛ И МЕТОДИЈ” - СКОПЈЕ

ФАКУЛТЕТ ЗА ЕЛЕКТРОТЕХНИКА И ИНФОРМАЦИСКИ ТЕХНОЛОГИИ

Glass Game

Проектна задача по предметот Мрежно програмирање

Изработиле:

Тамара Андоновска 144/2018

Сара Алексоска 142/2018

Професори:

Д-р. Марија Календар

Д-р. Марко Порјазоски

Содржина

Вовед 3
Серверска страна 3
Клиентска страна 5
Демонстрација на играта 10
Заклучок и можни подобрувања 16

Вовед

Играта Glass Game се состои од 8 реда со по 2 полиња и му дава на корисникот да избере по едно поле од секој ред. Трикот е во тоа што едното од полињата во редот е кршливо стакло, а другото не е. Доколку стапне на кршливо стакло, играчот губи еден живот од трите кои ги има на располагање. Идејата на играта е играчот да стигне од едниот до другиот крај без да ги изгуби сите животи.

Серверска страна

Серверската страна нуди поврзување на повеќе клиенти со користење на TCP сокет. Притоа, за да може серверот да опслужува повеќе клиенти истовремено, односно повеќе различни играчи да може да ја играат играта истовремено, се креира нов сокет и нитка по клиент кои ќе бидат одговорни за тој дел.

Се користат библиотеките:

- socket – за работа со сокетите
- _thread – за креирање на нова нитка за секој нов играч
- randrange од random – за рандомизација на полињата во играта

```
1  import socket, _thread
2  from random import randrange
```

На серверска страна, најпрво се креира TCP сокет за IPv4, односно од фамилијата AF_INET. Поврзувањето на самиот сокет се прави на loopback адресата, зашто и клиентот и серверот нѝ се наоѓаат на истата машина, а портата е рандом избрано бројче (1245 во случајов). Серверот е ограничен на 5 клиенти во својот ред на чекање.

```
60  s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
61  s.bind(('127.0.0.1', 1245))
62  s.listen(5)
```

При прифаќање на барање за конекција од клиент, серверот креира нова нитка, која ќе извршува посебна функција igrac, и нов сокет conn.

```
63  while True:
64      conn, addr = s.accept()
65      _thread.start_new_thread(igrac,(conn,))
```

igrac е функцијата која ќе ја извршува новокреираната нитка која го опслужува клиентот што побарал конекција до серверот, односно побарал да ја игра играта на серверот. Функцијата прима сокет како аргумент.

За секој играч во самата функција се креира листа a[] кое се пополнува со помош на бројач, така што парните елементи од листата се пополнуваат со рандом вредност добиена преку користење на randrange(0,2) функцијата која генерира random int вредност од [0, 2). Непарните елементи од листата ќе добијат вредност 1 доколку претходниот парен елемент бил 0, или 0 во обратниот случај.

Се води сметка и за бројот на животи, а со помош на бројачот br се внимава да не се излезе надвор од листата.

```
def igrac(s):
    a=[]
    i = 0
    red = 1
    while i < 16:
        br = randrange(0,2)
        a.append(br)
        if (br == 1):
            a.append(0)
        else:
            a.append(1)
        print("%d: %d %d" % (red, a[i], a[i+1]))
        i += 2
        red += 1
    tekst = "You have 3 lives and 8 rows of glass to cross. Each of the rows is made from one regular glass and one tempered glass. The regular glass b
    s.send(tekst.encode())
    brzivoti = 3
    br = 0
```

Комуникацијата меѓу серверот и клиентот, односно скриптата server.py и player.py се прави со користење на функциите send и recv, со тоа што се внимава истите да бидат усогласени.

При секој избор на корисникот на лево или десно поле во играта, серверот прима известување за избраното поле (лево или десно) и соодветно проверува во листата a[] дали стаклото на тоа поле било кршливо или не. Доколку не е кршливо, ја испраќа пораката "Good choice. The glass you stepped on was tempered.", а во спротивно се намалува бројот на животи и се печати пораката "You stepped on regular glass. Number of lives left: x", каде x е бројот на животи кој се чува во променливата brzivoti.

```
    while brzivoti > 0 and br < len(a):
        pole = s.recv(1024).decode()
        if (pole == "l"):
            print("levo")
            if (a[br] == 0):
                brzivoti -= 1
                if brzivoti == 0:
                    tekst = "No more lives left."
                else:
                    tekst = "You stepped on regular glass. Number of lives left: " + str(brzivoti)
                s.send(tekst.encode())
                br += 2
            else:
                br += 2
                if br == len(a):
                    tekst = "You have successfully reached the end."
                else:
                    tekst = "Good choice. The glass you stepped on was tempered."
                s.send(tekst.encode())
        else:
            print("desno")
            br += 1
            if (a[br] == 0):
                brzivoti -= 1
                if brzivoti == 0:
                    tekst = "No more lives left."
                else:
                    tekst = "You stepped on regular glass. Number of lives left: " + str(brzivoti)
                s.send(tekst.encode())
                br += 1
            else:
                br += 1
                if br == len(a):
                    tekst = "You have successfully reached the end."
                else:
                    tekst = "Good choice. The glass you stepped on was tempered."
                s.send(tekst.encode())
```

При секој избор се прави проверка на бројот на животи и доколку истиот е нула се печати пораката "No more lives left.". Се прави и проверка за тоа дали играчот стигнал успешно до крајот. Во тој случај, се печати "You have successfully reached the end.".

Клиентска страна

Кај клиентот се користат библиотеките:

- socket – за работа со сокетите
- sys, os – за функционалноста на Play Again копчето
- tkinter – за GUI делот

```
1  import socket
2  import sys, os
3  import tkinter
```

Клиентот креира TCP сокет и бара да се поврзе на адресата и портата на серверот. Првата порака која ја прима клиентот е пораката која ја праќа серверот како објаснување на играта и истата се печати кај клиентот.

```
9  s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10 s.connect(('127.0.0.1', 1245))
11 data = s.recv(1024).decode() # ova e prvata poraka sto ja objasnuva igrata
12 print(data)
```

За графичкиот кориснички интерфејс го користиме Tkinter. За таа цел потребно е да се креира корен – master, кој ќе ги содржи сите графички функционалности на играта. Со title се именува самиот прозорец, а со geometry се задаваат димензиите на прозорецот. Дополнително, позадината на прозорецот е поставена на dark grey.

```
14 master=tkinter.Tk()
15 master.title("Glass game")
16 master.geometry("850x750")
17
18 master.configure(bg='dark grey')
```

Со mainloop() му се кажува на Python да направи run на Tkinter јамката. Овој метод слуша настани.

```
239 master.mainloop()
```

Се креираат копчиња кои ја извршуваат функцијата clickedBtn која како аргумент го добива текстот на кликнатото копче и истата е објаснета подолу во елаборатот. Копчињата се поставени во grid и им е зададена соодветната редица и колона.

```
left1=tkinter.Button(master, text="L1", command = lambda: clickedBtn("L1"),height=3,width=6)
left1.grid(row=1,column=1,columnspan=2,rowspan=2)

left2=tkinter.Button(master, text="L2", command = lambda: clickedBtn("L2"),height=3,width=6)
left2.grid(row=3,column=1,columnspan=2,rowspan=2)

left3=tkinter.Button(master, text="L3", command = lambda: clickedBtn("L3"),height=3,width=6)
left3.grid(row=5,column=1,columnspan=2,rowspan=2)

left4=tkinter.Button(master, text="L4", command = lambda: clickedBtn("L4"),height=3,width=6)
left4.grid(row=7,column=1,columnspan=2,rowspan=2)

left5=tkinter.Button(master, text="L5", command = lambda: clickedBtn("L5"),height=3,width=6)
left5.grid(row=9,column=1,columnspan=2,rowspan=2)

left6=tkinter.Button(master, text="L6", command = lambda: clickedBtn("L6"),height=3,width=6)
left6.grid(row=11,column=1,columnspan=2,rowspan=2)

left7=tkinter.Button(master, text="L7", command = lambda: clickedBtn("L7"),height=3,width=6)
left7.grid(row=13,column=1,columnspan=2,rowspan=2)

left8=tkinter.Button(master, text="L8", command = lambda: clickedBtn("L8"),height=3,width=6)
left8.grid(row=15,column=1,columnspan=2,rowspan=2)

right1=tkinter.Button(master, text="R1", command = lambda: clickedBtn("R1"),height=3,width=6)
right1.grid(row=1,column=3,columnspan=2,rowspan=2)

right2=tkinter.Button(master, text="R2", command = lambda: clickedBtn("R2"),height=3,width=6)
right2.grid(row=3,column=3,columnspan=2,rowspan=2)

right3=tkinter.Button(master, text="R3", command = lambda: clickedBtn("R3"),height=3,width=6)
right3.grid(row=5,column=3,columnspan=2,rowspan=2)

right4=tkinter.Button(master, text="R4", command = lambda: clickedBtn("R4"),height=3,width=6)
right4.grid(row=7,column=3,columnspan=2,rowspan=2)

right5=tkinter.Button(master, text="R5", command = lambda: clickedBtn("R5"),height=3,width=6)
right5.grid(row=9,column=3,columnspan=2,rowspan=2)

right6=tkinter.Button(master, text="R6", command = lambda: clickedBtn("R6"),height=3,width=6)
right6.grid(row=11,column=3,columnspan=2,rowspan=2)

right7=tkinter.Button(master, text="R7", command = lambda: clickedBtn("R7"),height=3,width=6)
right7.grid(row=13,column=3,columnspan=2,rowspan=2)

right8=tkinter.Button(master, text="R8", command = lambda: clickedBtn("R8"),height=3,width=6)
right8.grid(row=15,column=3,columnspan=2,rowspan=2)
```

На играчот му е дозволено кликнување на копчиња од редот за кој моментално треба да направи избор, па на почетокот на играта ги оневозможуваме сите останати копчиња (полиња) освен оние од првиот ред.

```
left2['state'] = 'disabled'
left3['state'] = 'disabled'
left4['state'] = 'disabled'
left5['state'] = 'disabled'
left6['state'] = 'disabled'
left7['state'] = 'disabled'
left8['state'] = 'disabled'

right2['state'] = 'disabled'
right3['state'] = 'disabled'
right4['state'] = 'disabled'
right5['state'] = 'disabled'
right6['state'] = 'disabled'
right7['state'] = 'disabled'
right8['state'] = 'disabled'
```

Креираме и две копчиња за да му овозможиме на играчот да ја затвори играта со Close, користејќи ја вградената функција destroy на Tkinter, и да игра одново со Play Again,

преку функцијата restart_program.

```
90  btnPlayAgain =tkinter.Button(master, text="Play Again", command=restart_program,height=2,width=14)
91  btnPlayAgain.grid(row=23,column=0, columnspan=2)
92
93  btnClose = tkinter.Button(master, text="Close", command = master.destroy,height=2,width=14)
94  btnClose.grid(row=23,column=4, columnspan=2)
95  # so close se zatvara client (i tkinter sekako kako del od klientot)
```

```
5  def restart_program():
6      python = sys.executable
7      os.execl(python, python, * sys.argv)
```

За полесно вртење низ for јамките на функцијата clickedBtn креираме две листи кои го содржат текстот на левите и соодветно десните копчиња.

```
119  lefts = ["L1", "L2", "L3", "L4", "L5", "L6", "L7", "L8"]
120  rights = ["R1", "R2", "R3", "R4", "R5", "R6", "R7", "R8"]
```

Функцијата clickedBtn проверува кое копче е кликнато. Соодветно на тоа ги овозможува копчињата од наредниот и ги оневозможува копчињата од претходниот ред.

```
122  def clickedBtn(btnText):
123      if btnText == "R1" or btnText == "L1":
124          left2['state'] = 'normal'
125          right2['state'] = 'normal'
126          left1['state'] = 'disabled'
127          right1['state'] = 'disabled'
128
129      if btnText == "R2" or btnText == "L2":
130          left3['state'] = 'normal'
131          right3['state'] = 'normal'
132          left2['state'] = 'disabled'
133          right2['state'] = 'disabled'
134
135      if btnText == "R3" or btnText == "L3":
136          left4['state'] = 'normal'
137          right4['state'] = 'normal'
138          left3['state'] = 'disabled'
139          right3['state'] = 'disabled'
140
141      if btnText == "R4" or btnText == "L4":
142          left5['state'] = 'normal'
143          right5['state'] = 'normal'
144          left4['state'] = 'disabled'
145          right4['state'] = 'disabled'
146
147      if btnText == "R5" or btnText == "L5":
148          left6['state'] = 'normal'
149          right6['state'] = 'normal'
150          left5['state'] = 'disabled'
151          right5['state'] = 'disabled'
```

```
    if btnText == "R6" or btnText == "L6":
        left7['state'] = 'normal'
        right7['state'] = 'normal'
        left6['state'] = 'disabled'
        right6['state'] = 'disabled'

    if btnText == "R7" or btnText == "L7":
        left8['state'] = 'normal'
        right8['state'] = 'normal'
        left7['state'] = 'disabled'
        right7['state'] = 'disabled'

    if btnText == "R8" or btnText == "L8":
        left8['state'] = 'disabled'
        right8['state'] = 'disabled'
```

Се прави проверка дали кликнатото копче е лево или десно поле и истата информација се праќа до серверот. Од серверот се прима порака која може да биде:

- "Good choice. The glass you stepped on was tempered."
- "You stepped on regular glass. Number of lives left: X"
- "No more lives left."
- "You have successfully reached the end."

```
    if btnText in lefts:
        print("l")
        pole = "l"
        s.send(pole.encode())

    if btnText in rights:
        print("r")
        pole = "r"
        s.send(pole.encode())

    data = s.recv(1024).decode()
    print(data)
```

Зависно од тоа кое копче е кликнато, истото се обојува во темно црвена боја за играчот да може да ги следи своите избори.

```
183     if btnText == "L1":
184         left1['bg'] = '#B32727'
185     if btnText == "L2":
186         left2['bg'] = '#B32727'
187     if btnText == "L3":
188         left3['bg'] = '#B32727'
189     if btnText == "L4":
190         left4['bg'] = '#B32727'
191     if btnText == "L5":
192         left5['bg'] = '#B32727'
193     if btnText == "L6":
194         left6['bg'] = '#B32727'
195     if btnText == "L7":
196         left7['bg'] = '#B32727'
197     if btnText == "L8":
198         left8['bg'] = '#B32727'
199     if btnText == "R1":
200         right1['bg'] = '#B32727'
201     if btnText == "R2":
202         right2['bg'] = '#B32727'
203     if btnText == "R3":
204         right3['bg'] = '#B32727'
205     if btnText == "R4":
206         right4['bg'] = '#B32727'
207     if btnText == "R5":
208         right5['bg'] = '#B32727'
209     if btnText == "R6":
210         right6['bg'] = '#B32727'
211     if btnText == "R7":
212         right7['bg'] = '#B32727'
213     if btnText == "R8":
214         right8['bg'] = '#B32727'
```

Доколку пораката која играчот ќе ја прими од серверот е "No more lives left.", треба да се оневозможат сите полиња. Нема потреба од проверка дали пораката е "You have successfully reached the end." зашто ако играчот успешно стигнал до крајот, сите полиња ќе се оневозможени.

```
216     if data == "No more lives left.":
217         left1['state'] = 'disabled'
218         left2['state'] = 'disabled'
219         left3['state'] = 'disabled'
220         left4['state'] = 'disabled'
221         left5['state'] = 'disabled'
222         left6['state'] = 'disabled'
223         left7['state'] = 'disabled'
224         left8['state'] = 'disabled'
225         right1['state'] = 'disabled'
226         right2['state'] = 'disabled'
227         right3['state'] = 'disabled'
228         right4['state'] = 'disabled'
229         right5['state'] = 'disabled'
230         right6['state'] = 'disabled'
231         right7['state'] = 'disabled'
232         right8['state'] = 'disabled'
```

Сите пораки кои играчот ги добива од серверот се испишуваат како текст за играчот да

води сметка за текот на својата игра. Text box-от функционира на тој начин што мора да му се менува состојбата од normal во disabled и обратно при секоја промена на текстот во него.

```
20  text_box = tkinter.Text(master, width = 99, height = 3)
21  text_box.grid(row = 17, column = 1, columnspan = 4)
22
23  text_box.insert("end-1c", data)
24  text_box.configure(state='disabled', bg = "#B32727")
```

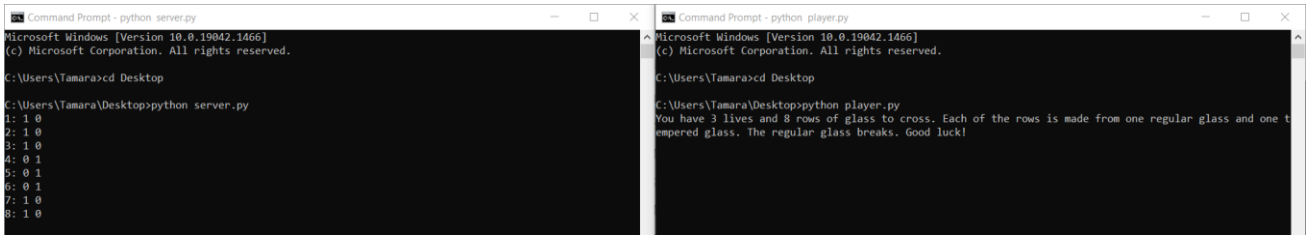
```
234         text_box.configure(state='normal')
235         text_box.delete(1.0, "end-1c")
236         text_box.insert("end-1c", data)
237         text_box.configure(state='disabled')
```

Овој дел од кодот е од естетска природа. Се додава padding на копчињата и на текстот.

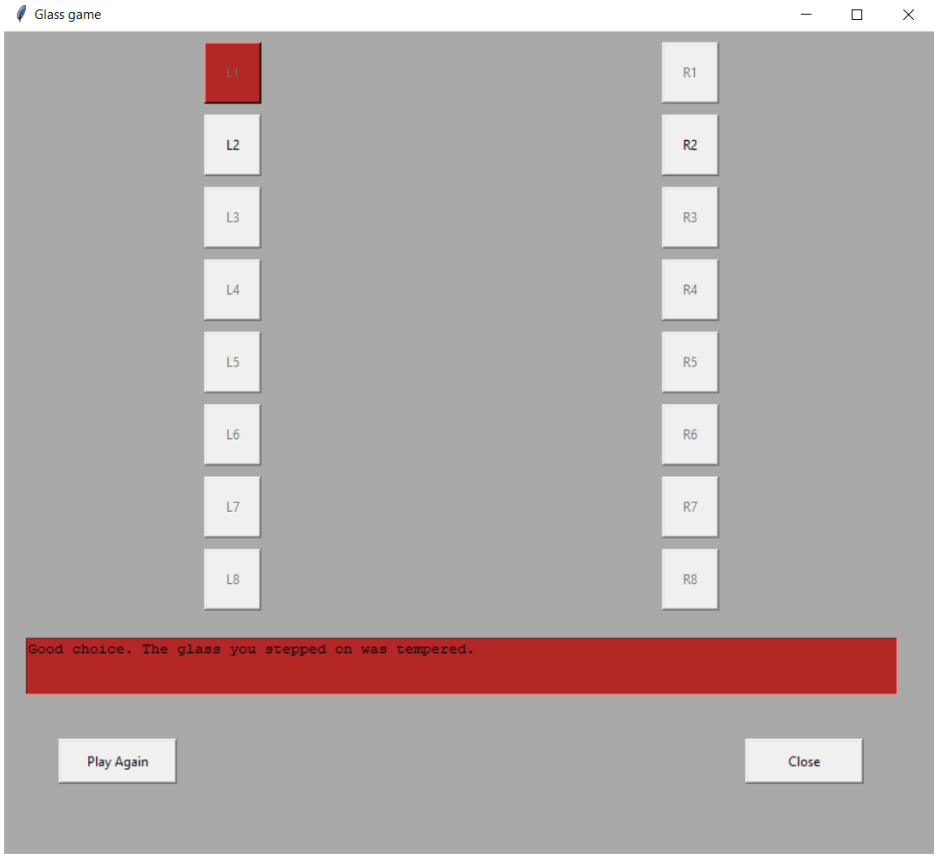
```
 96
 97  left1.grid(pady=(10,5))
 98  left2.grid(pady=(5,5))
 99  left3.grid(pady=(5,5))
100  left4.grid(pady=(5,5))
101  left5.grid(pady=(5,5))
102  left6.grid(pady=(5,5))
103  left7.grid(pady=(5,5))
104  left8.grid(pady=(5,5))
105
106  right1.grid(pady=(10,5))
107  right2.grid(pady=(5,5))
108  right3.grid(pady=(5,5))
109  right4.grid(pady=(5,5))
110  right5.grid(pady=(5,5))
111  right6.grid(pady=(5,5))
112  right7.grid(pady=(5,5))
113  right8.grid(pady=(5,5))
114
115  btnPlayAgain.grid(padx=(20,20),pady=(20,20))
116  btnClose.grid(padx=(20,20),pady=(20,20))
117  text_box.grid(padx=(20,20),pady=(20,20))
```

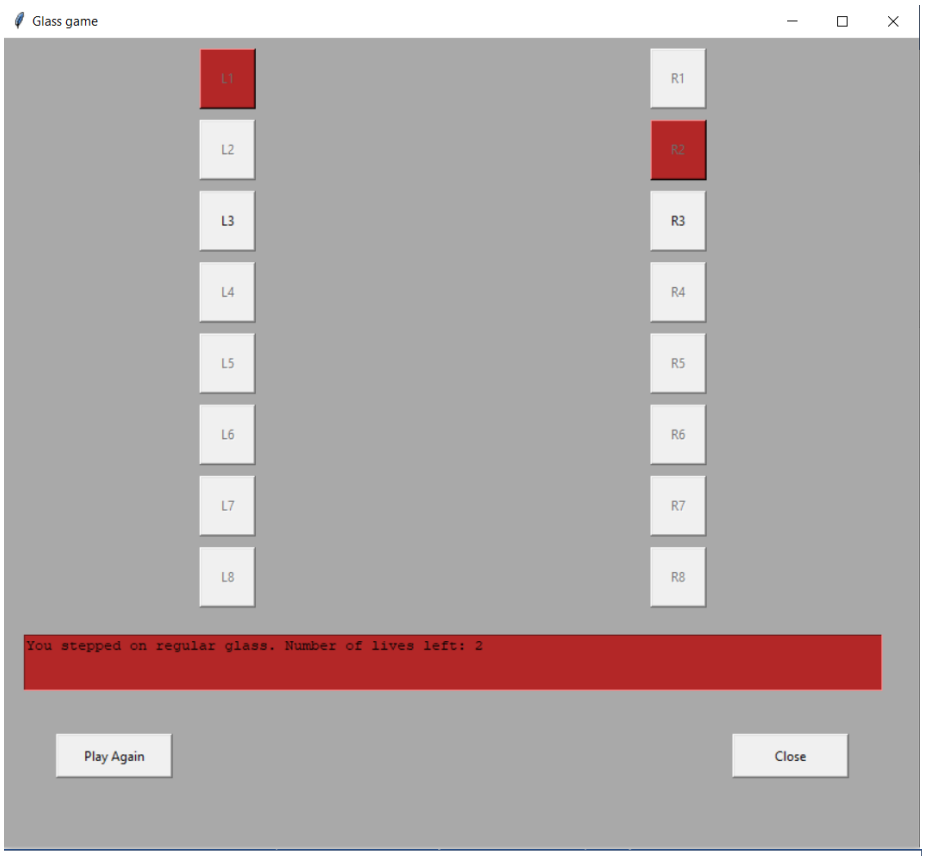
Демонстрација на играта

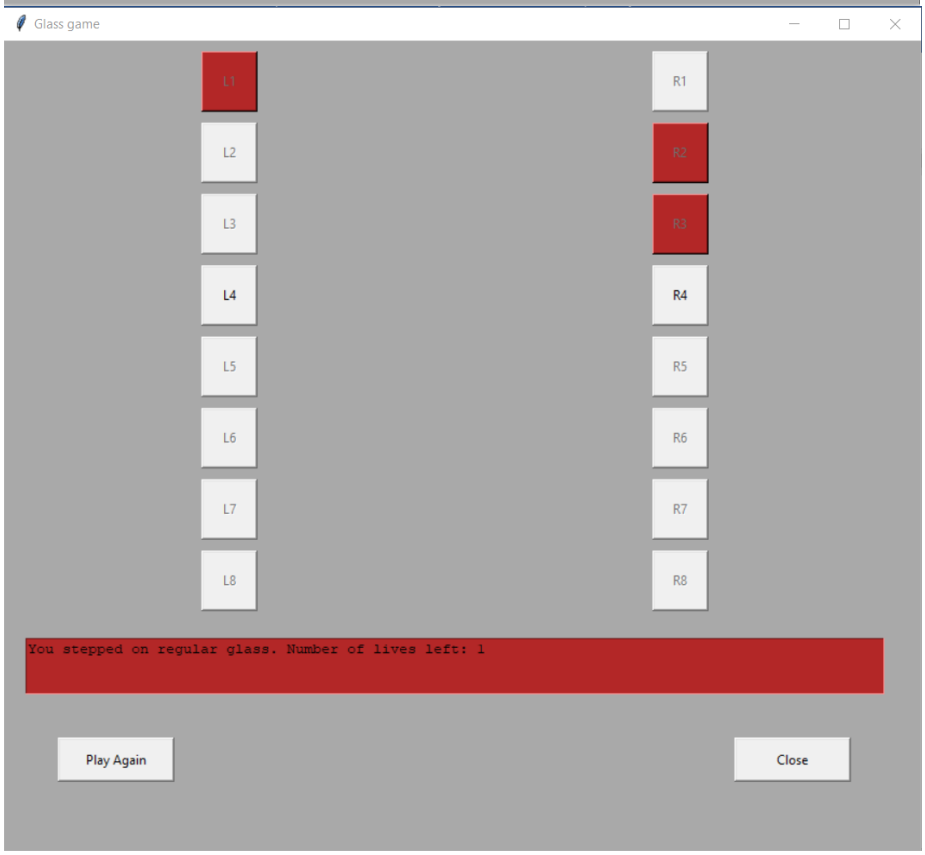
Се започнуваат два командни прозорци: на едниот се стартува серверот, а потоа на другиот клиентот.

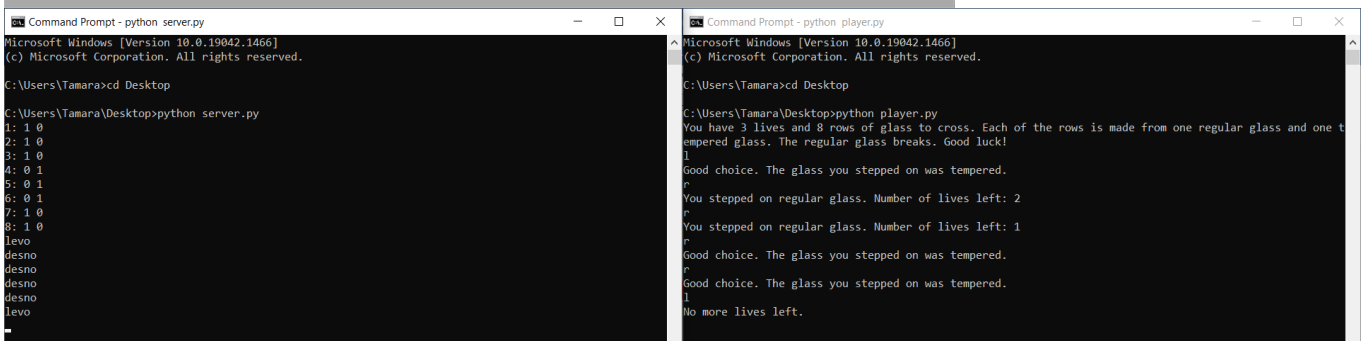



Glass game
L1
L2
L3
L4
L5
L6
L7
L8
R1
R2
R3
R4
R5
R6
R7
R8
You have 3 lives and 8 rows of glass to cross. Each of the rows is made from one regular glass and one tempered glass. The regular glass breaks. Good luck!
Play Again
Close

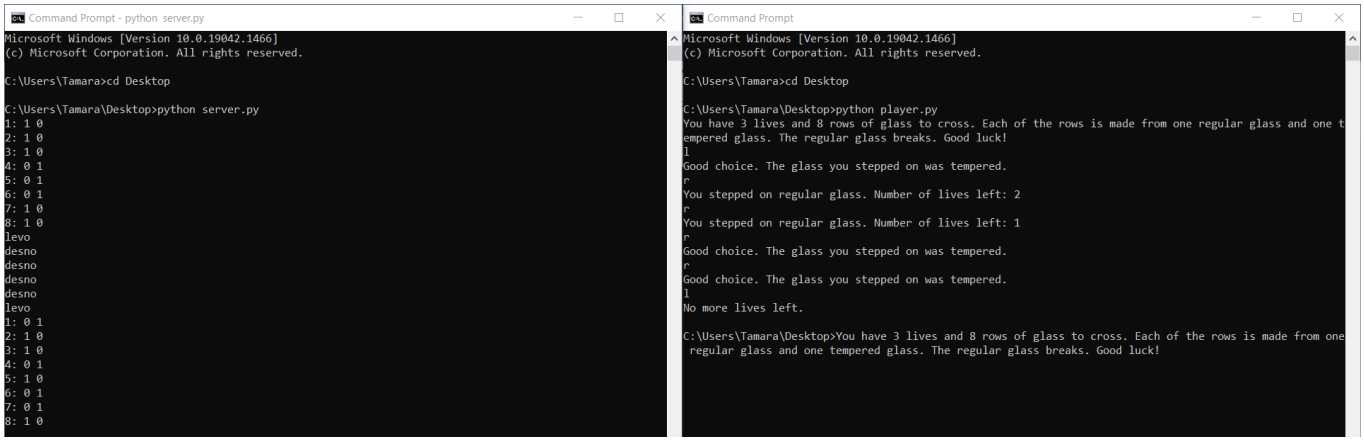

Glass game
L1
L2
L3
L4
L5
L6
L7
L8
R1
R2
R3
R4
R5
R6
R7
R8
Good choice. The glass you stepped on was tempered.
Play Again
Close


Glass game
L1
L2
L3
L4
L5
L6
L7
L8
R1
R2
R3
R4
R5
R6
R7
R8
You stepped on regular glass. Number of lives left: 2
Play Again
Close

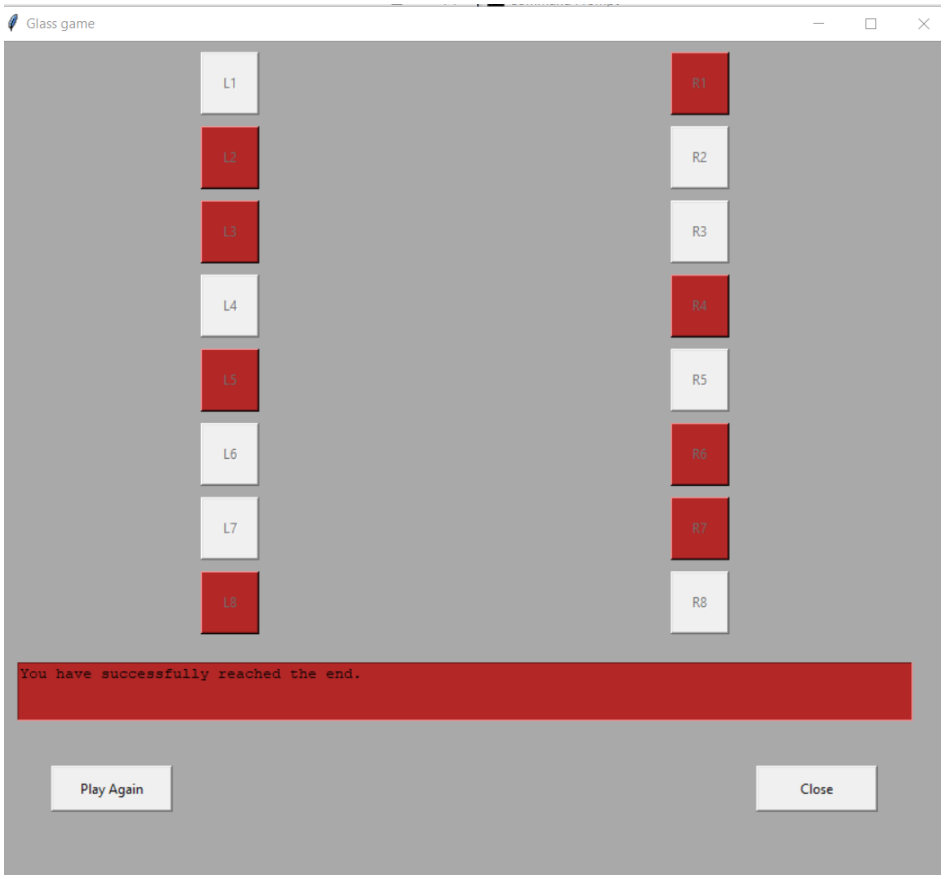

Glass game
L1
L2
L3
L4
L5
L6
L7
L8
R1
R2
R3
R4
R5
R6
R7
R8
You stepped on regular glass. Number of lives left: 1
Play Again
Close



По клик на копчето Play Again, можеме да почнеме повторно со играње.



Доколку успешно стигнеме до крајот:



Дополнително успеавме да ја run-уваме на два компјутери врзани на иста мрежа. Единствената разлика што ја направивме е наместо loopback, во кодот ја додадовме ip адресата на мрежата на која се врзани двата компјутери. Серверот е стратуван на еден компјутер, а клиентот се поврзува од друг компјутер и тој може да ја игра играта, со тоа што пристигнуваат логовите кај серверот за секој чекор на клиентот.

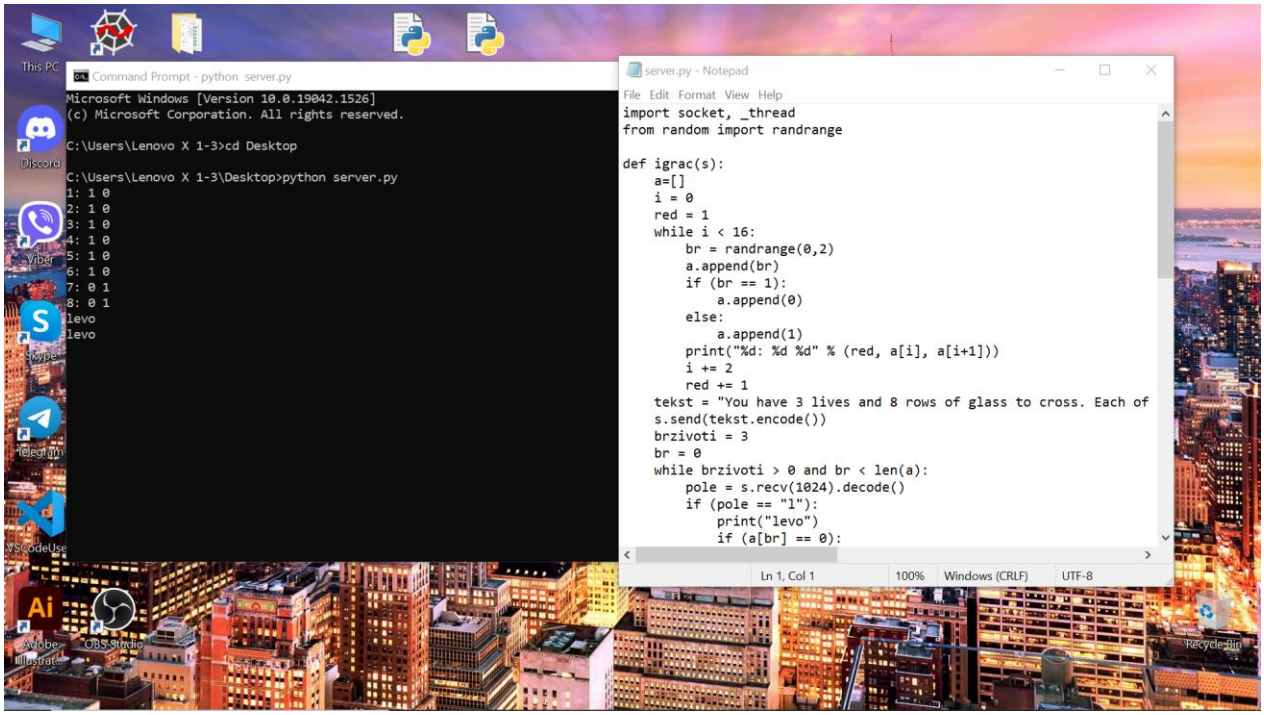
Промената кај серверот е во bind.

```
59
60  s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
61  s.bind(('192.168.100.6', 1245))
62  s.listen(5)
```

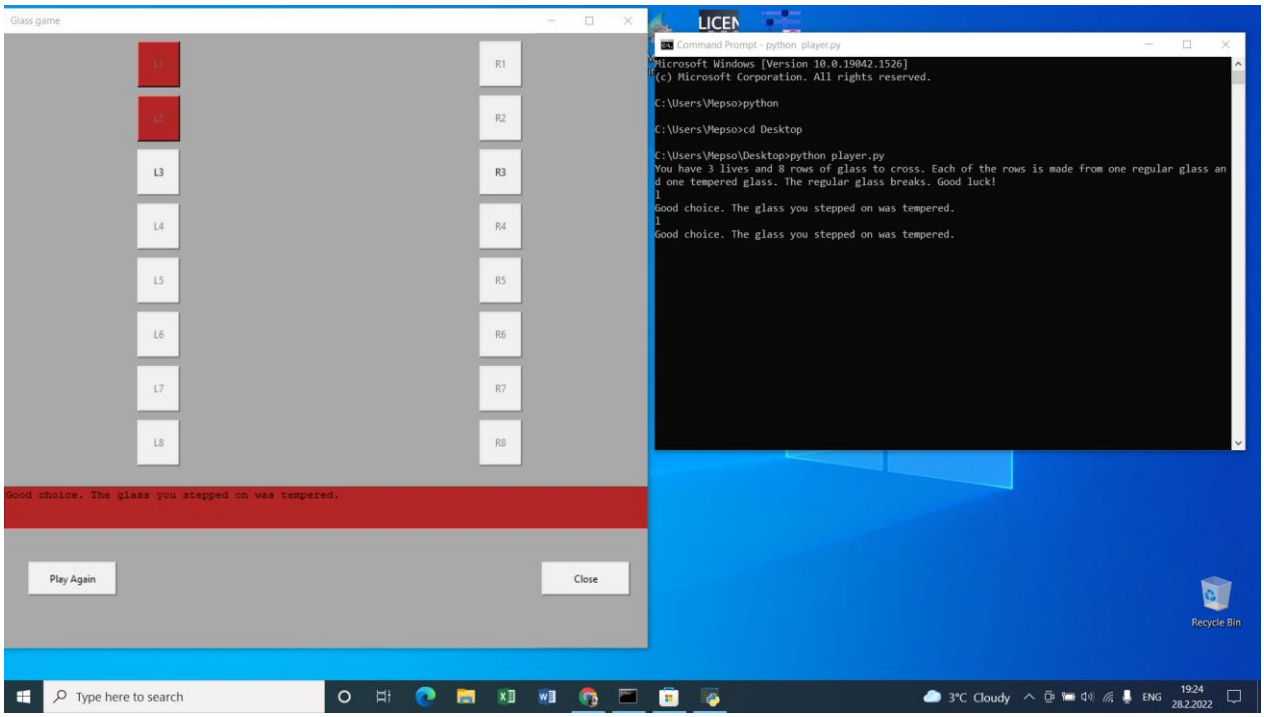
Промената кај клиентот е во connect.

```
 8
 9  s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10  s.connect(('192.168.100.6', 1245))
```

На првата машина е претставен серверот.



На друга машина е претставен клиентот.



Заклучок и можни подобрувања

За крај, важно е да напоменеме дека ова е едноставна игра која секако може да се подобри. Еден од начините би било потенцијално овозможување на повеќе играчи да се натпреваруваат во исто време, секој со по еден живот, како што е оригиналната игра од серијата од која ја добивме инспирацијата. Во тој случај, би било можно креирање на лоби во кое играчите би се приклучувале сè додека бројот не стигне до 16. Некои помали подобрувања во однос на веќе постоечката игра би се однесувале на самиот интерфејс: пр. полињата да се со слики од стакло наместо ознаки и сл.