TEHNIČKA ŠKOLA RUĐERA BOŠKOVIĆA

GETALDIĆEVA 4, ZAGREB

ZAVRŠNI STRUČNI RAD:

Obrazovna igra

MENTORICA: UČENIK: Martin Matišić

Kristinka Maček Blažeka RAZRED: 4.C

Zagreb, travanj 2021.

Sadržaj

1. UVOD

Postoje različite vrste obrazovnih igara koje možemo razviti i proučavati. U ovom primjeru ćemo se konkretno fokusirati na igru tipa kviza, koja je inspirirana sličnim igrama kao: „Tko želi biti milijunaš“, te ćemo vidjeti kako se razvija taj tip igre u programiranju.

Ovaj tip igre je izabran zbog želje za razumijevanjem programskog jezika „Python“ i njegove primjene za izradu igara ovog tipa te kako najbolje i najjednostavnije izvesti interakciju između programa i korisnika tj. unos korisnika. Programski jezik Python je izabran zbog njegove široke primjene i jednostavnosti sintakse.

Cilj rada jest izraditi obrazovnu igru pomoću programskog jezika Python, koristeći module, funkcije i druge alate koje nam Python pruža.

2. TEORIJSKI DIO

U kvizu je igraču ponuđeno 3 opcije: da započne igru, da napravi svoj kviz i da izađe. Ako igrač započne igru, upisat će na koliko pitanja želi odgovoriti iz općeg znanja koja će se prikazivati jedno za drugim te će mu nakon svakog odgovora biti napisano da li je dao točan ili kriv odgovor. Kada igrač odgovori na broj pitanja koji je upisao, igra će mu prikazati sve njegove odgovore i sve točne odgovore na pitanja koja su mu postavljena. Igrača se zatim opet pita da odabere jednu od opcija. Druga opcija služi korisniku da napravi vlastiti kviz sa svojim pitanjima. Korisnik može: napisati, brisati i spremati pitanja za svoju igru. Ako se igrač odluči da završi igru, program se prekida. Ovo se izvelo pomoću idućih tehnologija: programskog jezika Python-a i tekst editora Sublime Text 3.

2.1. Python

Slika 2.1 Logo Python

Python je programski jezik kojeg je stvorio Guido van Rossum 1990. godine. Ime Python je uzeto iz televizijske serije „Monty Python's Flying Circus“. Python je poznat zbog svoje jednostavne sintakse i širokih mogućnosti te se često preporučuje kao prvi programski jezik za nove programere.

Python se koristi za programiranje: internet stranica, video igrica, za izradu internih alata i upravljanje podacima. Iako Python ima široku primjenu, nije poželjno s njime raditi u svim situacijama. Python je kod izvršavanja koda sporiji od drugih programskih jezika kao što su C ili C++ zato što koristi jednu procesorsku jezgru. Ako želimo napraviti igricu koja zahtjeva veliku brzinu, Python možda ne bi bio najbolji izbor, dok bi program C++ ili JavaScript bili bolji izbor za izradu igrice. To ne znači da se Python ne može koristiti u tu svrhu, ali bitno je pogledati karakteristike programskog jezika i gledajući njih odlučiti što je za nas najbolja opcija.

Python se najčešće uspoređuje sa programskim jezikom Java. Oboje su interpreterski jezici i imaju samo jednu procesorsku jeziku. No Java se najčešće koristi kod izrade mobilnih aplikacija i interaktivnog web sadržaja.

2.2. Sublime Text

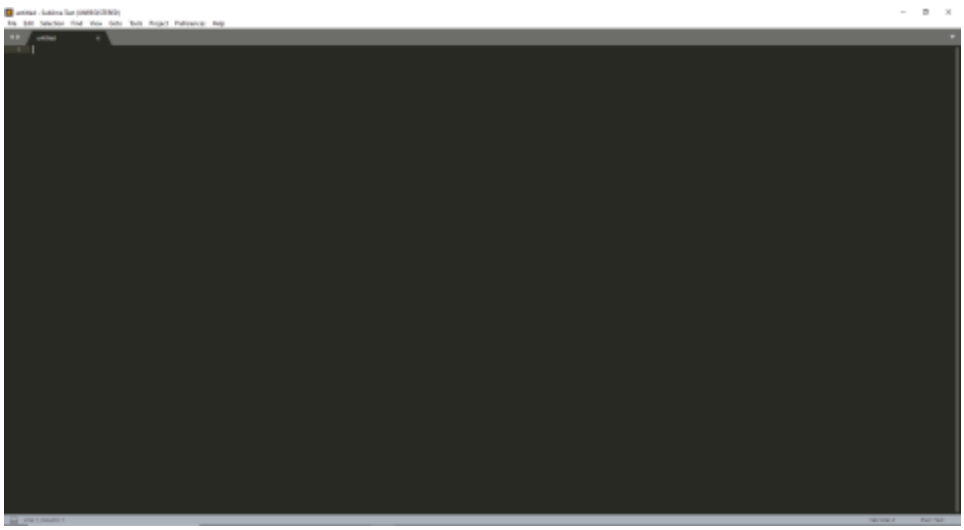
Sublime text je text editor (uređivač) za više platformi s Python programskim sučeljem. Bez dodataka podržava mnogo programskih jezika, a korisnici imaju opciju dodavati funkcije.

Neke od njegovih mogućnosti su:

1. „Goto anything“- za brzu navigaciju
2. „Command palette“- koristi podudaranje za brzo pozivanje proizvoljnih naredbi
3. Može istovremeno raditi promjene na dva različita mjesta.
4. Ima veliku prilagodljivost sa JSON

2.2.1 Sublime text 3

Verzija 3 se prvi put pojavila 2013. godine u beta verziji i bila je otvorena samo korisnicima koji su kupili Sublime text 2. Sublime Text 3 je postao dostupan svima 2017. godine, a verzija 3.2 je izašla 2019. g. Dva glavna dodatka koja su došla sa Sublime Text 3 je skeniranje datoteka za izradu indeksa kako bi se olakšalo značajkama „Goto defenition“ i „Goto symbol in Project“ i tzv. Pane Managment koji omogućuje korisnicima kretanje između prozora pomoću prečaca na tipkovnici.



Slika 2.2 Izgled Sublime Text 3 prozora

3. TEHNIČKO-TEHNOLOŠKI DIO

U ovom ćemo poglavlju pogledati kod obrazovne igre. Vidjet ćemo kako su organizirana: pitanja, odgovori i točni odgovori, kako se izvlači i postavlja pitanje i opcije koje korisnik ima na početku igre te kako korisnik može izraditi vlastitu igru.

```
1   #Ovi su modeli importani iz drugih dokumenata!
2   from Odgovori_i_pitanja import pitanja, odgovori, t_odgovori
3   from Costum_odgovori_i_pitanja import c_pitanja, c_odgovori, c_t_odgovori
4   import random
5
6   #Varijable koje se koriste kroz program!
7   odgovor = []
8   odgovoreno = []
9   tocan_odgovor = []
10  pogađanja = []
11  tocno = []
12  c_pitanje = []
13  c_odgovor = []
14  c_t_odgovor = []
15  bodovi = 0
16  i = 0
17  k = 0
18
19  # Ova funkcija ce generirati pitanja i odgovore!
20  def odabir_pitanja(pitanja, odgovori, t_odgovori): ...
43
44  # Ova funkicija ispisuje pitanje!
45  def ispis_pitanja(pitanja, odgovori): ...
63
64  # Ova funkcija ce provjeriti tocnost!
65  def provjera_tocnosti(odgovor, tocan_odgovor): ...
75
76  # Ovo je glavna igra!
77  def igra(pitanja, odgovori, t_odgovori): ...
125
126 # Ovo je funkcija za ispis i brisanje costum quiz-a!
127 def upravljanje_c_quiz(): ...
146
147 # Ova funkcija omogućuje da se napravi costum quiz!
148 def costum_quiz(): ...
211
212 # Ovo je menu igre!
213 def menu(): ...
235
236 menu()
```

Slika 3.1 Ovo je kod igre u kojem vidimo importane module i datoteke, varijable koje se koriste kroz program i funkcije koje su međusobno povezane

3.2. Moduli

```
# Ovi su moduli importani iz drugih dokumenata!
from Odgovori_i_pitanja import pitanja, odgovori, t_odgovori
from Costum_odgovori_i_pitanja import c_pitanja, c_odgovori, c_t_odgovori
import random
import os
import pickle
```

Slika 3.2 Importani moduli i datoteke

Općenito, modul je datoteka koja sadrži kod određenog programskog jezika. Python ima mnogo modula koji su dostupni korisniku odmah nakon preuzimanja programskog jezika. Neki primjeri Python modula: PyGame, Tkinter, Random itd. Za ovaj program smo koristili module: Random, Pickle i OS.

3.2.1. Modul Random

Modul Random generira nasumični broj ili znak (float) u određenom dometu. U kodu za obrazovnu igru, Random će generirati nasumično pitanje iz liste „pitanja“. Pomoću ovoga također možemo ispisati ponuđene odgovore i točan odgovor koji je povezan sa nasumično odabranim pitanjem.

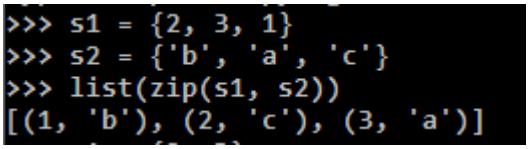
```
import random

print(random.randint(0,9))
```

Slika 3.3 Primjer Random modula, izlaz će biti broj između 0 i 9

3.2.2. Zip funkcija

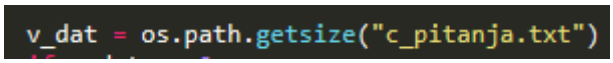
Zip funkciju koristimo kada želimo uzeti više sekvenca i spojiti ih (npr. dvije ili više lista). Spaja ih na način da uzima n-ti element jedne sekvence i pripadajući n-ti dio druge sekvence i spoji ih u „tuple“. Ako jedna od sekvenca nema jednaki broj argumenata kao druga, zip će ispisati tuple sa broj elemenata jednak broju elemenata u kraćoj sekvenci. Ako sekvence nemaju elemenata, zip će vratiti prazni iterator. Funkciji zip je nebitno kojeg tipa podataka su argumenti koje spaja (int, char, float). Ono što moramo paziti kada koristimo zip funkciju jest redoslijed argumenata. Ako argumenti nisu poredani ispravno zip će ih spojiti nasumično. U programu smo zip koristili kako bi liste sa: pitanjima, odgovorima i točnim odgovorima spojili u jednu cjelinu radi lakšeg ispisa.



Slika 3.4 Primjer zip funkcije, rezultat se sastoji od nasumično spojenih argumenata

3.2.3. OS modul

OS modul je modul koji nam daje pristup funkcijama za upravljanje datotekama, direktorijima i općenito operativnim sustavom te je jedan od modula koji je ugrađen u python. Za naš program je bitno razumjeti funkciju „os.path.getsize“ koju smo koristili za pronalaženje tekstualne datoteke na C disku i iščitavanje dužine liste u datoteci. Prvi dio „os.path“ pronalazi put do određene datoteke bez da mi moramo upisati cijeli put, a „getsize“ provjerava veličinu navedenog puta te vraća veličinu u bajtovima.



Slika 3.5 Kako izgleda os funkcija

3.2.4. Modul Pickle

Modul Pickle je jedan od načina za serijalizaciju i deserijalizaciju objekata u python-u. Proces serijalizacije je kada kompleksni objekt prebacujemo u seriju bajtova koju spremamo na disk ili šaljemo preko mreže. Ovaj proces se također zove „marshalling“. Proces suprotan ovome je deserijalizacija, u kojem se niz bajtova pretvara natrag u strukturu podataka. Pickle se često uspoređuje sa modulom Json. Json modul također sprema podatke, ali ih sprema na način koji čovjek može razumjeti, dok Pickle sprema na način koji čovjek ne može pročitati. Prednost Pickle modula jest da je brži te može spremiti puno više vrsti podataka iz python-a (npr. liste, varijable, korisnikove objekte itd.). Nedostatak Pickle modula jest njegov manjak tj. nedostatak sigurnosti. U našem programu koristimo Pickle kako bi spremili pitanja koja si je korisnik napisao za vlastitiu igru. Primjer kako izgleda pitanje koje spremimo pomoću Pickle modula: „€——————————————]q X Ĺ to pruÄŤava hidrologija?qa“.

3.3. Datoteke

Na početku koda, osim modula, također smo importali datoteke. Prvu datoteku koju smo importali je datoteka „Odgovori_i_pitanja.py“. Ova datoteka sadrži sva pitanja, odgovore i točne odgovore koja će se ispisivati u igri. Svako pitanje, odgovore i točne odgovore smo napisali kao zasebnu varijablu i na kraju smo ih sve stavili u zasebne liste. Druga datoteka sadrži prazne liste u koju korisnik može upisivati vlastita pitanja, odgovore i točne odgovore kroz jednu od opcija. Bitno je primijetiti da su se pitanja, ponuđeni odgovori i točni odgovori napisali na način da im se u listi njihovi indeksi podudaraju.

```
#ODGOVORI!!!
a1 = 'A. vodu \nB. zrak \nC. zemlju \nD. kamenje'
a2 = 'A. prije svega \nB. post skriptum \nC. post skrantum \nD. posljednje slovo'
a3 = 'A. Jugurta \nB. Hanibal \nC. Atila \nD. Arpad'
a4 = 'A. mornarici \nB. pješadija \nC. inžinjeri \nD. zrakoplovstvu'
a5 = 'A. slalomu \nB. spustu \nC. kombinaciji \nD. veleslalomu'

a6 = 'A. Madamme Butterfly \nB. Madamme Tussaud \nC. Maddame Bovary \nD. Maddame Micoud'
a7 = 'A. maler \nB. kelner \nC. šnajder \nD. maher'
a8 = 'A. gaza \nB. štof \nC. frotir \nD. jeans'
a9 = 'A. George Custer \nB. Robert Lee \nC. George Patton \nD. Ulysses Grant'
a10 = 'A. C \nB. D \nC. L \nD. V'

a11 = 'A. roman \nB. novelu \nC. drama \nD. tragedija'
a12 = 'A. Newtonu \nB. Kepleru \nC. Halleyju \nD. Saganu'
a13 = 'A. Alf \nB. E.T. \nC. Pikachu \nD. Dexter'
a14 = 'A. boje \nB. mirise \nC. zvukove \nD. okuse'
a15 = 'A. Ezstahija Bržić \nB. Brzi Gonzales \nC. Garfield \nD. Gargamel'

a16 = 'A. Kina \nB. Rusija \nC. Japan \nD. SAD'
a17 = 'A. Gonzo \nB. Pancho \nC. Kermit \nD. Torro'
a18 = 'A. Farskih \nB. Falklandskih \nC. Djevičanskih \nD. Cookovih'
a19 = 'A. strijelac \nB. jarac \nC. vodenjak \nD. škorpion'
a20 = 'A. Londonu \nB. Havani \nC. Šangaj \nD. Pisi'
```

Slika 3.6 Odgovori

```
#PITANJA!!!
q1 = 'Što proučava hidrologija?'
q2 = 'Što znači p.s. na kraju pisma?'
q3 = 'Kako se zvao hunski vojskovođa kojemu su pridjenuli nadimak "bič božji"?'
q4 = 'Pripadnost kojem rodu vojske, u kratici SAS, označava slovo A?'
q5 = 'U kojoj disciplini Janica Kostelić osvojila prvo hrvatsko zlato na ZOI?'

q6 = 'O kojoj je od navedenih gospođa Puccini napisao operu?'
q7 = 'Koji germanizam označava krojača?'
q8 = 'Koja se tkanina najčešće upotrebljava za izradu traperica?'
q9 = 'Tko je stradao kod Little Big Horna?'
q10 = 'Kako se rimskim brojevima piše broj 50?'

q11 = 'Koja se književna vrsta našla u naslovu Držićeve drame o Stancu?'
q12 = 'Po kojem je astronomu nazvan komet koji se vraća svakih 76 godina?'
q13 = 'Kako se zove biće s Melmaka s devet želudaca?'
q14 = 'Što ne raspoznaju daltonisti?'
q15 = 'Kako se zove najbrža animirana zmija na svijetu?'

q16 = 'Koja je najmnogoljudnija svjetska država?'
q17 = 'Kako se zove žabac iz "Muppet Showa" u kojeg je zaljubljena Miss Piggy?'
q18 = 'Zbog kojih su otoka zaratili Velika Britanija i Argentina 1982. godine?'
q19 = 'Koji je horoskopski znak čovjeka rođen na Božić?'
q20 = 'U kojem se gradu nalazi Kosi toranj?'
```

Slika 3.7 Pitanja

```
#TOČNI ODGOVORI!!!
ca1 = 'A'
ca2 = 'B'
ca3 = 'C'
ca4 = 'D'
ca5 = 'A'

ca6 = 'A'
ca7 = 'C'
ca8 = 'C'
ca9 = 'A'
ca10 = 'C'

ca11 = 'B'
ca12 = 'D'
ca13 = 'B'
ca14 = 'A'
ca15 = 'B'

ca16 = 'A'
ca17 = 'C'
ca18 = 'B'
ca19 = 'D'
ca20 = 'D'
```

Slika 3.8 Točni odgovori

```
pitanja = [q1, q2, q3, q4, q5, q6, q7, q8, q9, q10, q11, q12, q13, q14, q15, q16, q17, q18, q19, q20]
odgovori = [a1, a2, a3, a4, a5, a6, a7, a8, a9, a10, a11, a12, a13, a14, a15, a16, a17, a18, a19, a20]
t_odgovori = [ca1, ca2, ca3, ca4, ca5, ca6, ca7, ca8, ca9, ca10, ca11, ca12, ca13, ca14, ca15, ca16, ca17, ca18, ca19, ca20]
```

Slika 3.9 Liste koje se koriste u glavnom kodu

```
# Ovo su pitanja koja korisnik izmisli!

c_pitanja = []
c_odgovori = []
c_t_odgovori = []
```

Slika 3.10 Liste koje korisnik nadopunjava

3.4. Funkcije

Funkcija je izdvojena programska cjelina koja uzima argumente te pomoću tih argumenata izvodi svoj kod. Na kraju funkcije dobivamo rezultat funkcije. Funkciju možemo deklarirati bez argumenata. Za program je bitno reći da se funkcija može pozvati u drugoj funkciji. U kodu smo koristili funkcije za specifične dijelove koji zajedno čine igru.

```
# Ova funkcija ce generirati pitanja i odgovore!
def odabir_pitanja(pitanja, odgovori, t_odgovori): ...

# Ova funkicija ispisuje pitanje!
def ispis_pitanja(pitanja, odgovori): ...

# Ova funkcija ce provjeriti tocnost!
def provjera_tocnosti(odgovor, tocan_odgovor): ...

# Ovo je glavna igra!
def igra(pitanja, odgovori, t_odgovori): ...

# Ovo je funkcija za ispis i brisanje costum quiz-a!
def upravljanje_c_quiz(): ...

# Ova funkcija omogućuje da se napravi costum quiz!
def costum_quiz(): ...

# Ovo je menu igre!
def menu(): ...

menu()
```

Slika 3.11 Funkcije koje čine igru

3.4.1. Odabir pitanja

```
# Ova funkcija ce generirati pitanja i odgovore!
def odabir_pitanja(pitanja, odgovori, t_odgovori):
    global tocan_odgovor
    global odgovoreno
    global tocno
    global i
    global r_pitanje

    # Ovaj for loop ispisuje pitanje i ponudene odgovore!
    for p,o,t in zip(pitanja, odgovori, t_odgovori):
        if r_pitanje == p:
            print(f"\n{i + 1}. {p}")
            print(f"{o}")
            print("-------------------------------------------------------------------------")
            tocan_odgovor = t
            odgovoreno.append(r_pitanje)
            tocno.append(tocan_odgovor)
            i += 1

    return(tocno)
    return(i)
    return(odgovoreno)
    return(odgovori)
    return(r_pitanje)
```

Slika 3.12 Funkcija „odabir_pitanja()“

Ova funkcija je zaslužna za podudaranje nasumično dobivenog pitanja i pitanja iz postojeće liste pomoću koje onda ispiše pitanje i odgovore. To radi pomoću zip funkcije. Funkcija se sastoji od for() petlje koja će pretražiti iterator u kojem smo svaki element nazvali „p“ za pitanje, „o“ za odgovore i „t“ za točan odgovor. Nakon toga ispitujemo da li je nasumično pitanje koje smo dobili jednak jednom od pitanja u iteratoru. Ako je, funkcija će ispisati pitanje i moguće odgovore. Nasumično pitanje spremamo u listu „odgovoreno“, a točan odgovor spremamo u listu „tocno“ koju će program kasnije ispisati da pokaže korisniku koji su bili točni odgovori. Na kraju se varijabla „i“ poveća za jedan.

3.4.2. Provjera točnosti

```
# Ova funkcija ce provjeriti tocnost!
def provjera_tocnosti(odgovor, tocan_odgovor):
    global bodovi

    if odgovor == tocan_odgovor:
        print("Točno\n")
        bodovi += 1
    if odgovor != tocan_odgovor:
        if odgovor == 'A' or odgovor == 'B' or odgovor == 'C' or odgovor == 'D':
            print("Krivo\n")
    if odgovor != 'A' and odgovor != 'B' and odgovor != 'C' and odgovor != 'D':
        pass
```

3.13 Funkcija „provjera_tocnosti()“

Ova funkcija provjerava točnost odgovora. To radi tako da uzme odgovor koji je korisnik dao i onda ga usporedi sa točnim odgovorom. Ako je odgovor točan onda će program ispisati „Točno“ i povećati varijablu bodovi za 1. Ako je odgovor kriv, program ispisuje „Krivo“. Ako je korisnik slučajno upisao nešto osim: A, B, C ili D, funkcija neće ispisati riječ „Krivo“ i neće izvesti ostatak funkcije.

3.4.3. Ispis pitanja

```
# Ova funkicija ispisuje pitanje!
def ispis_pitanja(pitanja, odgovori):
    global pogađanja
    global tocno
    global r_pitanje

    # Ovaj dio koda usporeduje korisnikov odgovor sa tocnim odgovorm!
    odgovor = input("Odgovorite sa A / B / C / D: \n").upper()
    pogađanja.append(odgovor)
    provjera_tocnosti(odgovor, tocan_odgovor)
    while odgovor != 'A' and odgovor != 'B' and odgovor != 'C' and odgovor != 'D':
        pogađanja.remove(odgovor)
        print("\nMorate napisati A / B / C / D!")
        odgovor = input("Odgovorite sa A / B / C / D: \n").upper()
        provjera_tocnosti(odgovor, tocan_odgovor)
        pogađanja.append(odgovor)

    return(tocno)
    return(pogađanja)
```

Slika 3.14 Funkcija „ispis_pitanja(pitanja, odgovori)“

Ova funkcija uzima korisnikov odgovor i sprema ga u listu „pogađanja“. Zatim poziva funkciju „provjera_tocnosti()“ koja ispituje točnost odgovora. Petlja služi u slučaju da korisnik ne upiše: A, B, C ili D. Ako korisnik ne upiše jedno od ovih slova (npr. upiše slovo „p“ ili znak „?“), petlja će maknuti odgovor iz liste „pogađanja“ i obavijestit će korisnika da mora ponovno upisati slovo. Ova petlja će se ponavljati dok god korisnik ne upiše jedno od traženih slova. Kada korisnik upiše jedno od tih slova, ispitat će točnost kroz funkciju „provjera_tocnosti()“ i odgovor spremiti u listu „pogađanja“.

3.4.4. Igra

```
# Ovo je glavna igra!
def igra(pitanja, odgovori, t_odgovori):
    global i
    global pogađanja
    global odgovoreno
    global tocno
    global bodovi
    global r_pitanje
    global k

    k = int(input
        ("Na koliko pitanja želite odgovoriti?\n"
         "PAZITE DA NE UPIŠETE SLOVO I DA NE UPIŠETE VIŠE PITANJA NEGO ŠTO IMA (U SLUČAJU ZADANOG QUIZ-A 20)!\n"
         "\nUpišite '0' ako ne želite započeti igru.\n"))
    if k == 0:
        menu()
    k = k - 1

    # Ovaj while loop osigurava da se pitanja ne ponavljaju!
    while i <= k:
        r_pitanje = random.choice(list(pitanja))
        if r_pitanje in odgovoreno:
            while r_pitanje in odgovoreno:
                r_pitanje = random.choice(list(pitanja))
                if r_pitanje not in odgovoreno:
                    odabir_pitanja(pitanja, odgovori, t_odgovori)
                    ispis_pitanja(pitanja, odgovori)
                    break
                elif r_pitanje in odgovoreno:
                    r_pitanje = random.choice(list(pitanja))

        elif r_pitanje not in odgovoreno:
            odabir_pitanja(pitanja, odgovori, t_odgovori)
            ispis_pitanja(pitanja, odgovori)

    # Ovaj dio koda printa podatke kviza i resetira igru!
    print(f"Vaši bodovi: {bodovi}/{k + 1}")
    print(f"\nVaši odgovori: {pogađanja}")
    print(f"Točni odgovori: {tocno}")
    pogađanja.clear()
    tocno.clear()
    odgovoreno.clear()
    i = 0
    bodovi = 0
    k = 0

    return(tocno)
    return(odgvoreno)
    return(pogađanja)
    return(i)
    return(bodovi)
    return (k)
```

Slika 3.15 Funkcija „igra()“

Ova funkcija je sama igra. Prvo pita korisnika na koliko pitanja želi odgovoriti, što predstavlja varijabla „k“. Zatim oduzmemo od varijable „k“ 1 zato što brojanje pitanja počinje od 0.

Nakon toga slijedi petlja koja će se ponavljati dok god je „i“ (brojač) manji ili jednak varijabli „k“. Petlja uzima nasumično pitanje iz liste „pitanja“ pomoću modula Random.

Prvo program pogleda da li je nasumično pitanje u listi „odgovoreno“. Ako je, program će opet uzeti nasumično pitanje i opet provjeriti da li je u „odgovoreno“ i to će ponavljati sve dok ne dobije pitanje koje nije u listi. Kada nađe pitanje koje nije u listi „odgovoreno“, funkcija poziva „odabir_pitanja()“ za ispis pitanja i ponuđenih odgovora i „ispis_pitanja()“ za uzimanje korisnikovog odgovora te prekida petlju. Ako je nasumično pitanje i dalje u listi „odgovoreno“ program će opet odabrati novo nasumično pitanje.

Ako nasumično pitanje nije u listi „odgovoreno“ iz prve, onda program odmah poziva funkcije „odabir_pitanja()“ i „ispis_pitanja()“.

Kada varijabla „i“ postane veća od varijable „k“, funkcija ispisuje: bodove korisnika, odgovore koje je korisnik upisao i točne odgovore od pitanja kako bi ih mogao usporediti. Na kraju se liste: „pogađanja“, „tocno“ i „odgovoreno“ čiste tj. sve u njima se briše, te se varijable: „bodovi“, „i“ i „k“ smanji vrijednost na 0.

3.4.5. Izbornik

```
# Ovo je menu igre!
def menu():
    print("\n**************OBRAZOVNA IGRA**************\n")
    start = input(
        "Odaberite jednu od ovih opcija:"
        "\n*********************************"
        "\n1. START"
        "\n*********************************"
        "\n2. NAPRAVITE QUIZ"
        "\n*********************************"
        "\n3. KRAJ\n"
        ).upper()

    if start == "START" or start == "1" or start == "1.":
        igra(pitanja, odgovori, t_odgovori)
        menu()

    elif start == "NAPRAVITE QUIZ" or start == "2" or start == "2.":
        costum_quiz()

    elif start == "KRAJ" or start == "3" or start == "3.":
        print("Hvala na igranju! :)\n")
        quit()

    while start != 'start' and start != 'kraj':
        menu()

menu()
```

Slika 3.16 Funkcija „menu()“

Ova funkcija je izbornik igre. Kada se pokrene program, prvo što se ispiše korisniku jest ovaj izbornik. Korisnik je pitan da odabere jednu od ovih opcija. Ako korisnik upiše „start“, „1“ ili „1.“, pokrene se igra tj. pokrene se funkcija „menu()“. Ako korisnik odabere opciju „napravite quiz“, „2“ ili „2.“, korisnik ulazi u „costum mode“. Treća opcija prekida program ako korisnik upiše: „kraj“, „3“ ili „3.“. Dok korisnik ne upiše jednu od ovih opcija, konstantno će se ispisivati izbornik.

3.4.6. Korisnikova igra

```
# Ova funkcija omogućuje da se napravi costum quiz!
def costum_quiz():
    global c_pitanje
    global c_odgovor
    global c_t_odgovor
    global c_pitanja
    global c_odgovori
    global c_t_odgovori

    print("\n**************COSTUM QUIZ**************\n")
    opcija = input(
        "Odaberite jednu od ovih opcija:"
        "\n---------------------------------"
        "\n1. NAPIŠITE PITANJE"
        "\n---------------------------------"
        "\n2. ISPROBAJTE SVOJ QUIZ"
        "\n---------------------------------"
        "\n3. ISPIS/BRISANJE PITANJA"
        "\n---------------------------------"
        "\n4. IZAĐITE\n")

    if opcija == "NAPIŠITE PITANJE" or opcija == "1" or opcija == "1.":
        print("Ako želite izaći iz costum quiz-a, za vrijeme pisanja napišite 'quit'.\n")

        # Ovo je dodavanje pitanja, odgovora i tocnog odgovora!
        c_pitanje = input("\nOvdje upišite pitanje koje bi ste htijeli dodati u quiz:\n")
        if c_pitanje == 'quit':
            costum_quiz()
        else:
            c_pitanja.append(c_pitanje)

        print("\nOvdje upišite moguće odgovore na pitanje.(UPIŠITE SAMO ODGOVOR) \n")

        a = input("Pod A:\n")
        if a == 'quit':
            del a
            c_pitanja.remove(c_pitanje)
            costum_quiz()
        b = input("Pod B:\n")
        if b == 'quit':
            del a
            del b
            c_pitanja.remove(c_pitanje)
            costum_quiz()
        c = input("Pod C:\n")
        if c == 'quit':
            del a
            del b
            del c
            c_pitanja.remove(c_pitanje)
            costum_quiz()
        d = input("Pod D:\n")
        if d == 'quit':
            del a
            del b
            del c
            del d
            c_pitanja.remove(c_pitanje)
            costum_quiz()
```

Slika 3.17 funkcija „costum_quiz()“

```
        c_odgovor = (f"A.{a}" + f"\nB.{b}" + f"\nC.{c}" + f"\nD.{d}")
        c_odgovori.append(c_odgovor)

        c_t_odgovor = input("\nOvdje upišite samo slovo točnog odgovora pitanja: \n").upper()
        if c_t_odgovor == 'A' or c_t_odgovor == 'B' or c_t_odgovor == 'C' or c_t_odgovor == 'D':
            c_t_odgovori.append(c_t_odgovor)
            costum_quiz()
        elif c_t_odgovor == 'QUIT':
            c_pitanja.remove(c_pitanje)
            c_odgovori.remove(c_odgovor)
            costum_quiz()
        else:
            while c_t_odgovor != 'A' and c_t_odgovor != 'B' and c_t_odgovor != 'C' and c_t_odgovor != 'D':
                print("\nMorate napisati A / B / C / D!")
                c_t_odgovor = input("\nOvdje upišite samo slovo točnog odgovora pitanja: \n").upper()
                if c_t_odgovor == 'A' or c_t_odgovor == 'B' or c_t_odgovor == 'C' or c_t_odgovor == 'D':
                    c_t_odgovori.append(c_t_odgovor)
                    costum_quiz()
                if c_t_odgovor == 'QUIT':
                    c_pitanja.remove(c_pitanje)
                    c_odgovori.remove(c_odgovor)
                    costum_quiz()

    # Ovo pokreće costum quiz!
    if opcija == "ISPORBAJTE SVOJ QUIZ" or opcija == "2" or opcija == "2.":
        if not c_pitanja:
            print("Nemate pitanja za quiz!")
            costum_quiz()
        igra(c_pitanja, c_odgovori, c_t_odgovori)
        costum_quiz()

    # Ovo ulazi u upravljanje costum quiz-a!
    if opcija == "ISPIS/BRISANJE PITANJA" or opcija == "3" or opcija == "3.":
        upravljanje_c_quiz()

    # Ovo izlazi iz costum quiz-a!
    if opcija == "IZAĐITE" or opcija == "4" or opcija == "4.":
        menu()

    while opcija != '1':
        costum_quiz()

    return(c_pitanja)
    return(c_odgovori)
    return(c_t_odgovori)
```

Slika 3.18 funkcija „costum_quiz()“

Ova funkcija je izbornik za „costum mode“ u kojem korisnik može napraviti vlastitu obrazovnu igru te njome upravljati na više načina. Ova opcija može poslužiti nekom ako želi ispitati svoje znanje. U ovaj izbornik korisnik ulazi tako da u glavnom izborniku upiše broj 2. Zatim će se korisniku ponuditi 4 opcije.

Odabirom prve opcije će korisnik napisati svoje pitanje, moguće odgovore i jedan točan odgovor. Prvo se ispiše da za vrijeme pisanja bilo kojeg dijela pitanja može upisati „quit“ u slučaju da se predomislio i ne želi napisati pitanje. Zatim se pita korisnika da upiše svoje pitanje koje smo pridodali varijabli „c_pitanje“ te ako „c_pitanje“ nije riječ „quit“ program nastavlja. Drugo ga pita da upiše moguće/ponuđene odgovore. Upisuje ih jedan za drugim, te je svaki odgovor zasebna varijabla; odgovor pod A je pridružen varijabli „a“, odgovor pod B je pridružen varijabli „b“ itd. Ako za vrijeme pisanja odgovora upiše „quit“, ovisno o tome na kojem je odgovoru, brišu se sve prijašnje varijable pomoću „del“ i varijabla „c_pitanje“ se ukloni iz liste „c_pitanja“ i vraća se na izbornik. Ako se ispiše sva 4 odgovora, svaki od njih se pridodaju varijabli „c_odgovor“ na način da će se ispisivati jedan ispod drugog kada

korisnik pokrene svoju igru („\n“ u python-u znači da krene u novi red) te se sprema u listu „c_odgovori“. Na kraju se korisnika pita da napiše samo slovo točnog odgovora: a, b, c ili d koje se sprema kao „c_t_odgovor“ varijabla. Ako korisnik upiše jedno od tih slova, varijabla „c_t_odgovor“ će se spremiti u listu „c_t_odgovori“ i vratiti će se na izbornik. Ako je korisnik upisao „QUIT“ onda će se iz liste „c_pitanja“ izbriše varijabla „c_pitanje“ i iz liste „c_odgovori“ briše se varijabla „c_odgovor“ te se vraća na izbornik. Ako korisnik nije napisao ništa od navedenog (npr. upisao je slovo L ili broj 8) onda se korisnika opet pita da upiše: a, b, c ili d. Ova petlja će se ponavljati sve dok korisnik ne upiše jednu od tih opcija ili „quit“. Ako korisnik upiše a, b, c ili d onda se prekida petlja i sprema se vraijabla i vraća se na izbornik. Ako korisnik upiše „quit“ onda se brišu vraijable i vraća se na izbornik.

Opcija broj dva služi za pokretanje igre sa korisnikovim pitanjima. Funkcija prvo pogleda da li je lista „c_pitanja“ prazna pomoću „not c_pitanja“. Ako je prazna, igra se ne pokreće i ispisuje se poruka koja obavještava da korisnik nema pitanja za igru. Ako ima pitanja onda će uzeti liste „c_pitanja“, „c_odgovori“ i „c_t_odgovori“ i postaviti ih kao argumente za funkciju „igra()“. Program se vraća na izbornik nakon igre.

Treća opcija u izborniku ulazi u dodatne opcije za upravljanje korisnikove igre, a te opcije se nalaze u funkciji „upravljanje_c_quiz()“. Četvrta opcija vraća korisnika na početni izbornik.

3.4.7. Dodatne opcije za korisnikovu igru

```
# Ovo je funkcija za ispis i brisanje costum quiz-a!
def upravljanje_c_quiz():
    global c_pitanja
    global c_odgovori
    global c_t_odgovori

    upravljanje = input(
        "\nOdaberite jednu od ovih opcija:"
        "\n---------------------------------"
        "\n1. ISPIS QUIZ-A"
        "\n---------------------------------"
        "\n2. UČITAVANJE QUIZ-A"
        "\n---------------------------------"
        "\n3. SPREMANJE QUIZ-A"
        "\n---------------------------------"
        "\n4. BRISANJE QUIZ-A"
        "\n---------------------------------"
        "\n5. BRISANJE PITANJA"
        "\n---------------------------------"
        "\n6. IZAĐITE\n")

    if upravljanje == "ISPIS QUIZ-A" or upravljanje == "1" or upravljanje == "1.":
        print(f"Pitanja:{c_pitanja} \nOdgovori:{c_odgovori} \nTočni odgovori:{c_t_odgovori}")
        upravljanje_c_quiz()

    if upravljanje == "UČITAVANJE QUIZ-A" or upravljanje == "2" or upravljanje == "2.":

        v_dat = os.path.getsize("c_pitanja.txt")
        if v_dat == 0:
            print("\nNemate pitanja za učitavanje!")
            upravljanje_c_quiz()

        c_p = open("c_pitanja.txt", "rb")
        while True:
            try:
                c_pitanja.extend(pickle.load(c_p))
            except EOFError:
                break

        c_o = open("c_odgovori.txt", "rb")
        while True:
            try:
                c_odgovori.extend(pickle.load(c_o))
            except EOFError:
                break

        c_t_o = open("c_t_odgovori.txt", "rb")
        while True:
            try:
                c_t_odgovori.extend(pickle.load(c_t_o))
            except EOFError:
                break
```

Slika 3.19 funkcija „upravljanje_c_quiz()“

```
if upravljanje == "SPREMANJE QUIZ-A" or upravljanje == "3" or upravljanje == "3.":
    if not c_pitanja:
        print("Nemate pitanja koja možete spremiti!")
        upravljanje_c_quiz()

    pickle.dump(c_pitanja, open("c_pitanja.txt", "wb"))
    pickle.dump(c_odgovori, open("c_odgovori.txt", "wb"))
    pickle.dump(c_t_odgovori, open("c_t_odgovori.txt", "wb"))

if upravljanje == "BRISANJE QUIZ-A" or upravljanje == "4" or upravljanje == "4.":
    c_pitanja.clear()
    c_odgovori.clear()
    c_t_odgovori.clear()

    x = input(
        "\nDa li želite izbrisati i spremljeni quiz (D / N)?\n"
        "\nAko izbrišete spremljeni quiz, sva pitanja koja ste spremili će se izbrisati!\n").upper()
    prazna_lista = []
    if x == "D" or x == "DA":
        c_p = open("c_pitanja.txt", "wb")
        pickle.dump(prazna_lista, open("c_pitanja.txt", "wb"))
        c_p.close()

        c_o = open("c_odgovori.txt", "wb")
        pickle.dump(prazna_lista, open("c_odgovori.txt", "wb"))
        c_o.close()

        c_t_o = open("c_t_odgovori.txt", "wb")
        pickle.dump(prazna_lista, open("c_t_odgovori.txt", "wb"))
        c_t_o.close()
        upravljanje_c_quiz()

    if x == "N" or x == "NE":
        upravljanje_c_quiz()

if upravljanje == "BRISANJE PITANJA" or upravljanje == "5" or upravljanje == "5.":
    if not c_pitanja:
        print("Nemate pitanja!")
        upravljanje_c_quiz()
    print(f"Pitanja:{c_pitanja} \nOdgovori:{c_odgovori} \nTočni odgovori:{c_t_odgovori}")
    y = int(input("Napišite broj pitanja iz liste koje bi ste htijeli izbrisati:\n"))
    y = y - 1
    c_pitanja.pop(y)
    c_odgovori.pop(y)
    c_t_odgovori.pop(y)
    upravljanje_c_quiz()

if upravljanje == "IZAĐITE" or upravljanje == "6" or upravljanje == "6.":
    costum_quiz()

while upravljanje != '1':
    upravljanje_c_quiz()
```

Slika 3.20 funkcija „upravljanje_c_quiz()“

Kada korisnik odabere 3. opciju kod izbornika „COSTUM QUIZ“, program ga premjesti u ovaj izbornik sa dodatnim opcijama. Ove opcije sluše za modificiranje korisničke igre.

Prva opcija služi za ispis korisničkog quiz-a. Ako korisnik odabere opciju, ispisat će sva pitanja , odgovore i točne odgovore koja se trenutačno nalaze u listama: „c_pitanja“, „c_odgovori“ i „c_t_odgovori“ te će ga vratiti na izbornik.

Druga opcija služi za učitavanje korisnikovih pitanja iz tekstualnih datoteka pomoću modula Pickle. Prvo program provjerava jesu li datoteke prazne. To radi pomoću linije: “v_dat = os.path.getsize(„c_pitanja.txt“)“ te ako je rezultat jednak 0, korisniku će se ispisati poruka da nema spremljena pitanja i vraća ga na izbornik. Ako se u datotekama nalaze pitanja, program

će otvoriti tekstualne datoteke samo sa listom pitanja. Nakon što ga otvori, program će proširiti postojeću listu „c_pitanja“ sa listom „c_p“ koja je zapravo lista iz tekstualne datoteke koju smo učitali. Proces učitavanja liste u program i proširivanje liste „c_pitanja“ se napravi u istoj liniji. Ovaj kod se nalazi u petlji koja će se ponavljati sve dok ne isčita cijelu listu. Općenito, kada čitač dođe do kraja datoteke javlja se tzv. „EOFError“. Ako se on pojavi za vrijeme izvršavanja programa, javlja se greška i program se naglo prekida. Stoga u programu uzimamo u obzir tu pogrešku te kada se ona pojavi petlja se prekida („break“). Ovaj proces se ponavlja za datoteke sa odgovorima i točnim odgovorima.

Treća opcija u izborniku jest za spremanje korisnikove igre u datoteke: „c_pitanja“, „c_odgovori“ i „c_t_odgovori“. Prvo program provjerava ima li lista „c_pitanja“ pitanja koja se mogu spremiti. Ako nema, program ispisuje da je lista prazna te se vraća na izbornik. Ako postoje pitanja u listi, program koristi jednu od funkcija koju nudi modul Pickle. Koristi funkciju „dump()“ koja uzima trenutačne liste koje su u programu te ih enkapsulira i sprema u datoteke. Bitno je naglasiti da ako ne postoje navedene datoteke, funkcija „open()“ će ih stvoriti u istom direktoriju gdje se nalazi kod. Također je bitno da ako se nešto nalazi u datotekama i korisnik spremi trenutne liste bez da učita sadržaj datoteka, trenutni sadržaj datoteka se briše i liste u programu se stavljaju u predodređene datoteke.

Četvrta opcija briše trenutna pitanja, odgovore i točne odgovore u listama programa ili, ako korisnik želi, može izbrisati pitanja u datotekama. Liste u programu se brišu pomoću funkcije „clear()“. Ako korisnik želi izbrisati sadržaj datoteka, program radi sljedeće. Prvo postavimo praznu listu te otvorimo datoteku. Zatim praznu listu spremimo u datoteku pomoću funkcije „dump()“. Ako pogledamo u datoteke vidjet ćemo znak „□煸ᒻ“, koji označava da je datoteka prazna tj. nema objekata.

Peta opcija se koristi za brisanje individualnog pitanja i njegovih pripadnih odgovora i točnog odgovora iz lista u programu. Prvo se provjerava je li lista „c_pitanja“ prazna. Ako je, korisnika se upozorava i vraća se na izbornik. Ako u listi postoje pitanja, program će korisniku ispisati liste: „c_pitanja“, „c_odgovori“ i „c_t_odgovori“. Zatim će ga pitati da upiše broj pitanja iz lista koje želi izbrisati. Varijabla „y“ predstavlja broj pitanja i prije nego što izbrišemo pitanje, od varijable moramo oduzeti 1 jer liste počinju od indeksa 0. Kada korisnik upiše broj, program pomoću funkcije „pop()“ briše pitanje indeksa „y“. Šesta opcija vraća korisnika na izbornik „COSTUM QUIZ“.

4. ZAKLJUČAK

Za izradu ove igre smo je bilo potrebno na pregledan i jednostavan način organizirati podatke kako bi kod bio što kraći i što jednostavniji za razumjeti.

Kako bi pojednostavili kod, koristili smo funkcije i liste. Pomoću funkcija smo segmentirali rad što nam olakšava programiranje. Svaka funkcija ima svoju svrhu koja je čisto definirana da uvijek imamo ideju što određena funkcija treba raditi i da lakše možemo testirati ispravnost koda tijekom pisanja te implementiranjem jedne funkcije u drugu smanjujemo veličinu koda.

Liste smo koristili da možemo jer je s njima jednostavno raditi. Pitanja koja su već implementirana u kod su stavljena u liste. Ovaj način korištenja lista nam daje veću fleksibilnost kod izmjenjivanja pitanja, ali zauzima više mjesta kod spremanja. Za razliku od toga, korištenje modula Pickle nam je omogućilo da napravimo vlastita pitanja, koja možemo spremati. Iako je mijenjanje pitanja otežano, pošto svaki put kada želimo nešto promijeniti moramo učitati liste iz tekstualnih dokumenata, pitanja koja su spremljena na ovaj način zauzimaju manje prostora.

Opcija za spremanje lista u datoteke može biti korisna na više načina sama po sebi. Korisnik pomoću te opcije može ispitati svoje znanje u određenom polju znanja, ili se kod može proširiti i staviti na Internet stranicu na kojoj korisnici mogu podijeliti svoje igre.

5. LITERATURA

Python (programming language). URL: https://en.wikipedia.org/wiki/Python_(programming_language)

Sublime text. URL: https://en.wikipedia.org/wiki/Sublime_Text

Real Python URL: https://en.wikipedia.org/wiki/Sublime_Text

stackoverflow. URL: https://stackoverflow.com/

Dan Bader URL: https://dbader.org/

Kite URL: https://www.kite.com/