|  |
| --- |
| f=open("fakultet.txt", "r") |
|  | studenti=f.read() |
|  | f.close() |
|  | print(studenti) |
|  |  |
|  | with open("fakultet.txt", "r") as f: |
|  | studenti=f.read() |
|  | print(studenti) |
|  |  |
|  | with open("fakultet1.txt", "w") as f1: |
|  | f1.write("Ivan Ivanic\n") |
|  |  |
|  | with open("fakultet.txt", "r") as f: |
|  | studenti=f.readline() |
|  | print(studenti) |
|  |  |
|  | with open("fakultet.txt", "r") as f: |
|  | line=f.readline() |
|  | while line: |
|  | print("--", line) |
|  | line = f.readline() |
|  |  |
|  | with open("fakultet.txt", "r") as f: |
|  | line=f.readline() |
|  | while line: |
|  | print("--", repr(line)) |
|  | line = f.readline() |
|  |  |
|  | with open("fakultet.txt", "r") as f: |
|  | for line in f: |
|  | print(line) |
|  |  |
|  | with open("fakultet.txt", "r") as f: |
|  | lines=f.readlines() |
|  | print(lines) |
|  |  |
|  |  |
|  |  |
|  | import json |
|  |  |
|  | '''student = {"student": {"ime": "Ante", |
|  | "prezime": "Antić", |
|  | "dob": 21}} |
|  |  |
|  | with open("fakultet.json", "w") as f: |
|  | json.dump(student, f)''' |
|  |  |
|  | with open("fakultet.json", "r") as f: |
|  | student=json.load(f) |
|  | print(student) |
|  |  |
|  |  |
|  |  |
|  | import sqlite3 |
|  |  |
|  | conn=sqlite3.connect("fakultet.db") |
|  | cur=conn.cursor() |
|  |  |
|  | '''cur.executescript(""" |
|  | CREATE TABLE IF NOT EXISTS grupe ( |
|  | id text PRIMARY KEY, |
|  | naziv text NOT NULL); |
|  |  |
|  | CREATE TABLE IF NOT EXISTS studenti ( |
|  | id integer PRIMARY KEY, |
|  | ime text NOT NULL, |
|  | prezime text NOT NULL, |
|  | grupa\_id text DEFAULT NULL, |
|  | FOREIGN KEY (grupa\_id) REFERENCES grupe(id)); |
|  |  |
|  | CREATE TABLE IF NOT EXISTS profesori ( |
|  | id text PRIMARY KEY, |
|  | titula text NOT NULL, |
|  | ime text NOT NULL, |
|  | prezime text NOT NULL); |
|  |  |
|  | CREATE TABLE IF NOT EXISTS kolegiji ( |
|  | id integer PRIMARY KEY, |
|  | naziv text NOT NULL, |
|  | profesor\_id integer NOT NULL, |
|  | FOREIGN KEY (profesor\_id) REFERENCES profesori (id)); |
|  |  |
|  | CREATE TABLE IF NOT EXISTS grupe\_kolegiji( |
|  | grupa\_id integer not NULL, |
|  | kolegij\_id integer not NULL, |
|  | PRIMARY KEY (grupa\_id, kolegij\_id) |
|  | FOREIGN KEY (grupa\_id) REFERENCES grupe (id), |
|  | FOREIGN KEY (kolegij\_id) REFERENCES kolegiji (id));""")''' |
|  |  |
|  | '''#cur.execute("INSERT INTO grupe (id, naziv) VALUES (?, ?)", ("MAT", "Matematika")) |
|  | #conn.commit() |
|  | cur.execute("SELECT \* FROM grupe") |
|  | print(cur.fetchall())''' |
|  |  |
|  | '''grupe = [("INF", "Informatika"), |
|  | ("BIO", "Biologija"), |
|  | ("KEM", "Kemija")] |
|  | cur.executemany("INSERT INTO grupe (id, naziv) VALUES (?,?)", grupe) |
|  |  |
|  | cur.execute("SELECT \* FROM grupe") |
|  | print(cur.fetchall())''' |
|  |  |
|  | '''cur.execute("INSERT INTO studenti (ime, prezime, grupa\_id) VALUES (?,?,?)", |
|  | ("Ante", "Antic", "MAT")) |
|  | print(cur.lastrowid) |
|  | conn.commit() |
|  |  |
|  | cur.execute("SELECT \* FROM studenti") |
|  | print(cur.fetchall())''' |
|  |  |
|  | '''studenti = [("Dora", "Dorić", "MAT"), |
|  | ("Filip", "Filipić", "INF"), |
|  | ("Marija", "Marijić", "BIO"), |
|  | ("Pero", "Perić", "KEM")] |
|  | cur.executemany("""INSERT INTO studenti (ime, prezime, grupa\_id) VALUES (?, ?, ?)""", |
|  | studenti) |
|  | conn.commit() |
|  | cur.execute("SELECT \* FROM studenti") |
|  | print(cur.fetchall())''' |
|  |  |
|  | '''cur.execute("UPDATE studenti SET grupa\_id=? WHERE id=?", ("INF",2)) |
|  | conn.commit() |
|  | cur.execute("SELECT \* FROM studenti") |
|  | print(cur.fetchall())''' |
|  |  |
|  | '''cur.execute("DELETE FROM studenti WHERE id=?", (3, )) |
|  | conn.commit() |
|  | cur.execute("SELECT \* FROM studenti") |
|  | print(cur.fetchall())''' |
|  |  |
|  | cur.execute(""" |
|  | SELECT |
|  | studenti.ime, |
|  | studenti.prezime, |
|  | grupe.naziv |
|  | FROM studenti |
|  | JOIN grupe ON studenti.grupa\_id=grupe.id |
|  | """) |
|  |  |
|  | print(cur.fetchone()) |
|  | print(cur.fetchone()) |
|  | print(cur.fetchone()) |