Računarski praktikum 3, 2022/2023, 1. zadaća

Napomene:

- U zadaći se ne boduje efikasnost (složenost) rješenja.
- Svi rubni slučajevi koji nisu pokriveni tekstom zadaće ostavljeni su vama na izbor.
- Nije dopušten bilo koji oblik zajedničkog rada. U slučaju da koristite vanjske izvore (npr. Stack Overflow), navedite ih u obliku komentara na početku datoteke `Program.cs`.
- Predaja zadaće je na Merlinu.
- Prije predaje kopirajte čitav kod u `Program.cs` ako ste koristili više datoteka.

Jedan je kuhar uočio da često ima potrebu napisati recept koji je kombinacija nekih drugih recepata, ili recept koji je jednak nekom postojećem receptu ali s npr. 20% manje šećera.

U ovom zadatku pišemo program koji će pomoći Kuharu automatizirati ovakve poslove.

Recepte ćemo spremati u obliku JSON dokumenata. To je tekstualni format unutar kojeg se mogu zapisivati primjerice stringovi, brojevi, liste i rječnici (mape). Stringovi su omeđeni dvostrukim navodnicima, liste uglatim zagradama, a rječnici vitičastim zagradama. Korijen je JSON datoteke uvijek rječnik. Za jednostavnije čitanje i pisanje datoteka u JSON formatu unutar .NET Frameworka potrebno je dodati Microsoftov paket System.Text.Json kroz Visual Studio. Pogledajte primjer korištenja. Između ostalog, trebat će vam klase `JsonNode`, `JsonObject`, `JsonArray` i `JsonValue`. Koristite dokumentaciju kako biste provjerili koje funkcionalnosti ove klase nude i kako se koriste.

Primjer recepta, `slatka_sol.json`:

```
{
    "ime": "Slatka sol",
    "sastojci": [
        {
            "ime": "secer",
            "kolicina": 100
        },
        {
            "ime": "sol",
            "kolicina": 1000
        },
        {
            "ime": "Led (ovo je primjer grupe sastojaka)",
            "sastojci": [
```

```
                {
                    "ime": "usitnjen led",
                    "kolicina": 100
                },
                {
                    "ime": "kockice leda",
                    "kolicina": 100
                }
            ]
        }
    ],
    "koraci": [
        "Pripremite dvije posude",
        {
            "ime": "Vaganje",
            "koraci": [
                "Izvazite secer",
                "Izvazite sol"
            ]
        },
        "Pomijesajte sadrzaj posuda",
        "Dodajte jos secera po ukusu"
    ],
    "posluzivanje": [
        "Ukrasite ledom"
    ]
}
```

Ključevi "sastojci", "koraci" i "posluzivanje" mogu i ne moraju biti prisutni u receptu. Ako su prisutni, bit će liste. Liste sastojaka i koraka mogu biti ugniježđene kao u primjeru (dubina ugniježđenja nije ograničena). Pritom ugniježđeni sadržaj mora biti istog tipa (koraci ili sastojci), npr. unutar liste koraka možemo imati korak koji sadrži svoje korake, ali ne i svoje sastojke. Sve liste mogu biti i prazne. Svi rječnici u dokumentu mogu sadržavati ključ "ime" (iznimka su ref-rječnici koje ćemo uskoro opisati). Rječnik koji je korijen recepta, i samo taj rječnik, mora sadržavati ključ "ime". Ponovno, ref-rječnici su iznimka, no nakon procesa kojeg ćemo zvati dereferenciranje, rječnik koji je korijen također mora imati ime. Minimalan recept stoga je rječnik koji sadrži samo ključ "ime", primjerice `{"ime": "Prazan recept"}`.

Za rječnik koji je sadržan negdje unutar datoteke (uključujući rječnik koji je korijen) kažemo da je ime toga rječnika X ako on (neposredno) sadrži ključ "ime" i vrijednost za ključ "ime" jest X. Svako ime rječnika (za rječnike koji imaju ime) može sadržavati samo znakove `A-Z`, `a-z`, `0-9`, `_`, `-` i razmake.

Referencama zovemo stringove poput sljedećeg:

```
slatka_sol.json#Slatka sol/sastojci/3/sastojci/2
```

Ova je referenca kratak zapis za "dio datoteke slatka_sol.json, unutar rječnika čije je ime 'Slatka sol', unutar vrijednosti pod ključem 'sastojci', 3. vrijednost u listi (indeksiranje kreće jedinicom jer Kuhar nije navikao na indeksiranje koje kreće nulom), unutar vrijednosti pod ključem 'sastojci', 2. vrijednost u listi". Uz raniji primjer recepta, ta referenca referira na:

```
{
    "ime": "kockice leda",
    "kolicina": "100"
}
```

Općenito, svaka referenca mora početi imenom datoteke, simbolom `#`, i potom imenom rječnika. Može se dogoditi da se neko ime rječnika javlja više puta u istoj datoteci, tj. više rječnika može imati isto ime. U tom slučaju referenca referira na bilo koji rječnik takvog imena. Ostatak je reference (možda prazan) niz oblika `/x/y/z...` (npr. `/sastojci/3/sastojci/2`) gdje je svaki segment ili indeks (indeksiranje kreće jedinicom) nekog elementa liste (ako dotadašnji dio reference referira na listu) ili ključ rječnika (ako dotadašnji dio reference referira na rječnik).

Kuhar koristi reference za sastojke i korake koji se ponavljaju na više mjesta (u jednom ili više recepata), kako bi u slučaju trajne promjene sastojka morao promijeniti samo jedno mjesto u svojim receptima. Na bilo kojem mjestu u datoteci (uključujući na mjestu rječnika koji je korijen), Kuhar može napisati rječnik poput sljedećeg:

```
{
    "ref": "slatka_sol.json#Slatka sol/sastojci/3/sastojci/2"
}
```

Gornji primjer znači da na to mjesto (umjesto danog ref-rječnika) želimo da piše sljedeće:

```
{
    "ime": "kockice leda",
    "kolicina": "100"
}
```

Nazovimo proces zamjene referenci onime na što referiraju *dereferenciranjem*. U slučaju da se ime rječnika u referenci nalazi na više mjesta u zatraženoj datoteci, svaki je izbor jednako dobar.

Reference mogu pokazivati na objekte koji i sami ovise o nekim referencama (koje i same mogu pokazivati na objekte koji ovise o referencama itd.). Recept nakon dereferenciranja više ne sadrži reference, tj. prilikom dereferenciranja treba zamijeniti reference na svakoj dubini. Ref-rječnici osim ključa "ref" ne smiju sadržavati druge ključeve, uključujući ključ "ime" (tj. ako rječnik sadrži ključ "ref", nema drugih ključeva). Primjerice, dereferenciranjem sljedećeg recepta spremljenog kao `a.json`:

```
{
    "ime": "Recept A",
    "sastojci": [
        {
            "ref": "a.json#X/sastojci/1"
        },
        {
            "ime": "X",
            "sastojci": [
                {
                    "ref": "a.json#Y/sastojci/1"
                }
            ]
        },
        {
            "ime": "Y",
            "sastojci": [
                {
                    "ref": "a.json#Sastojak koji ce se ponoviti"
                }
            ]
        },
        {
            "sastojci": [
                {
                    "ime": "Sastojak koji ce se ponoviti",
                    "kolicina": 1
                }
            ]
        }
    ]
}
```


Dobivamo sljedeći rječnik:


```
{
    "ime": "Recept A",
    "sastojci": [
        {
            "ime": "Sastojak koji ce se ponoviti",
            "kolicina": 1
        },
        {
            "ime": "X",
            "sastojci": [
                {
                    "ime": "Sastojak koji ce se ponoviti",
```

```
                "kolicina": 1
            }
        ]
    },
    {
        "ime": "Y",
        "sastojci": [
            {
                "ime": "Sastojak koji ce se ponoviti",
                "kolicina": 1
            }
        ]
    },
    {
        "sastojci": [
            {
                "ime": "Sastojak koji ce se ponoviti",
                "kolicina": 1
            }
        ]
    }
]
}
```

Zbog jednostavnosti ne dopuštamo da je vrijednost ključa "ref" neki drugi ref-rječnik. Primjerice, ne dopuštamo sljedeći ref-rječnik: `{"ref": {"ref": "a.json#neka/referenca"}}`. Osim toga, ne dopuštamo da je vrijednost bilo kojeg ključa "ime" ref-rječnik, već očekujemo da je vrijednost ključa "ime" uvijek string. Ne trebate provjeravati jesu li ova svojstva ispunjena. Ako nisu ispunjena, rješenje može imati proizvoljno ponašanje.

U slučaju da Kuhar nije pažljiv, može se dogoditi da napravi kružnu referencu. Primjer je sljedeća datoteka `a.json`:

```
{
    "ime": "Recept A",
    "sastojci": [
        {
            "ref": "a.json#Recept A/sastojci/1"
        }
    ]
}
```

Ovakve recepte (ispunjavaju formu recepta, ali nemoguće ih je dereferencirati) zovemo *kružnima*. Naravno, mogući su i manje očiti primjeri *kružnih* recepata. Posebno, da je gornja referenca glasila "a.json#Recept A/sastojci/1/12345/x/y/z/999", to bi također bila kružna referenca jer je nemoguće dereferencirati već njen početak: "a.json#Recept A/sastojci/1".

Jedan primjer ispravnih ali neobičnih recepata (koji se na prvi pogled mogu činiti kružnima) `a.json` i `b.json`. U datoteci `a.json` možemo imati:

```
{
    "ime": "Recept A",
    "koraci": [
        "Korak 1",
        {
            "ref": "b.json#Recept B/koraci/3"
        },
        {
            "ref": "b.json#Recept B/koraci/2"
        },
        {
            "ref": "b.json#Recept B/koraci/4"
        }
    ]
}
```

U datoteci `b.json` možemo imati:

```
{
    "ime": "Recept B",
    "koraci": [
        {
            "ref": "a.json#Recept A/koraci/4"
        },
        "Korak 2",
        {
            "ref": "a.json#Recept A/koraci/3"
        },
        {
            "ref": "a.json#Recept A/koraci/1"
        }
    ]
}
```

Ove dvije datoteke ovise jedna o drugoj kroz nekoliko lanaca referenci koji prolaze kroz obje datoteke, ali možemo ih dereferencirati, tj. nema kružnih referenci.

Moguće je da neka referenca sadrži ime rječnika koji ne postoji eksplicitno u traženoj datoteci, ali će se pojaviti u procesu dereferenciranja. Dakle, nije dovoljno pročitati potpuno neobrađenu datoteku kako bi se provjerilo postoji li rječnik s nekim imenom u njoj ili ne.

Iste polazne datoteke mogu rezultirati različitim dereferenciranim verzijama jer imena rječnika ne moraju biti jedinstvena (pa time reference nisu uvijek jednoznačne), i poredak ključeva rječnika može biti proizvoljan. Sve te verzije smatraju se jednako ispravnima i dovoljno je pronaći jednu.

Zadatak je napisati komandolinijski program koji kao argument (`argv[0]`) prima ime datoteke i potom:

- ispisuje na ekran dereferenciranu verziju ulazne datoteke (gdje su sve reference zamijenjene sadržajem na kojeg pokazuju);
- ako postoje kružne reference ili postoji bilo kakav drugi problem (npr. postoji referenca na nepostojeću datoteku ili nepostojeći dio dokumenta), program može imati proizvoljno ponašanje.

U nekim situacijama kružne reference nisu neizbježan problem, npr. ako postoje dva rječnika imena `R`, te drugi sadrži referencu na `R`. Tada referenca u drugom rječniku nije nužno kružna jer ju možemo shvatiti kao referencu na prvi rječnik. Rješenje ne mora uzimati ovakve slučajeve u obzir, tj. rješenje može imati proizvoljno ponašanje čak i ako postoji referenca čije je barem jedno moguće dereferenciranje kružno.

Pretpostavite da je ulazna datoteka, kao i sve datoteke na koje ona izravno ili neizravno referira, spremljena u istom direktoriju u kojem je izvršna datoteka (`.exe`), i iz tog direktorija testirajte program. Ovisno o konfiguraciji projekta, to je vjerojatno direktorij `direktorij_projekta/bin/Debug/`.

Primjeri:

- Primjer 1
- Primjer 2
- Primjer 3
- Primjer 4