1. Napiši funkciju koja (na funkcionalni način) invertira rječnik, odnosno, sve vrijednosti rječnika neka postanu ključevi, a ključevi vrijednosti. Pretpostavi da će rječnik biti injektivno mapiranje, odnosno, sve vrijednosti bit će različite.

2. Napiši funkciju koja prima arbitraran broj pozicijskih argumenata, a vraća string u kojem su svi argumenti, u obrnutom poretku, u novom redu.
   Primjer: `rows_in_reverse('string', 2.4, 1)` → `"1\n2.4\nstring"`
   Bonus bodovi: obradi slučaj sa imenovanim argumentima

3. Raspiši sljedeće reducense preko `reduce()`:

   - `sum()`
   - `any()`
   - `all()`
   - `min()`
   - `max()`

4. Napiši funkciju koja prima prirodni broj, a vraća sumu znamenaka broja. No, umjesto onako kako smo radili na predavanju, napiši pomoćnu funkciju koja generira znamenke broja. Za dobivanje pojedinačnih znamenaka koristi operatore `//` te `%`, ili alternativno, ugrađenu funkciju `divmod()`. Za bonus bodove, testiraj performanse oba pristupa :))

5. (Malo teži ali ne puno) Razmisli i napiši funkciju koja u funkcionalnoj paradigmi računa binomni koeficijent $\binom{n}{k}$. Namjerno nisam opisao "standardni" zapis računanja binomnog koeficijenta (s faktorijelima) jer postoje drugi, efikasniji (čitaj: primjenjiviji) zapisi koje bi valjalo istražiti :D

6. Napiši program koji računa Eulerovu funkciju $\phi(n)$
   (`https://en.wikipedia.org/wiki/Euler's_totient_function#Euler's_product_formula`).
   Eulerova funkcija broji koliko je brojeva manjih od n relativno prosto s n:

   - Napiši funkciju koja provjerava je li neki broj prost. Broj 1 **nije prost**.
   - Iskoristi tu funkciju za dohvaćanje svih prostih brojeva manjih od n.
     (Alternativno, zamjeni ova dva koraka funkcionalnom implementacijom Eratostenovog sita)
   - Filtriraj sve te proste brojeve tako da ostaviš samo one koji cjelobrojno dijele n. Po definiciji, broj a cijelobrojno dijeli broj b ako pri djeljenju b/a nema ostatka.
   - Mapiraj te preostale prim brojeve prema izrazu u zapisu Eulerove funkcije
   - Reduciraj rezultatni generator umnoškom (ilitiga zdravoseljački, pomnoži ih sve)
   - Naposlijetku, izračunaj vrijednost Eulerove funkcije. Obrati pozornost na to kojeg tipa treba biti rezultat Eulerove funkcije - veličina skupa brojeva ne može nikako biti `float`!

   Da bi zadatak bio u potpunosti točno riješen, nigdje ne smije biti privremenih lista, sve mora ići preko generatora/generatorskih funkcija, i reducenasa koje iste konzumiraju. Priznajem i alternativne implementacije (imam jednu koju ću rado detaljno objasniti ako mi netko šibne mail), dokle god su čisto funkcionalne i kojima je povratni tip cijeli broj

7. Računanje statističkih momenata poput srednje vrijednosti i varijance vrlo je česta operacija u analizi podataka. Nažalost, naivna implementacija toga krcata je problemima: a) visoko je numerički nestabilna b) potrebna su dva prolaza (jedan za srednju vrijednost, drugi za varijancu). Srećom, može se bolje; a ti i budeš! B. P. Welford je 1962. našao bolji algoritam koji zahtjeva samo jedan prolaz po podacima! Detalje možeš naći na Wikipediji i na `https://www.johndcook.com/blog/standard_deviation`.
   Tvoj zadatak je implementirati generatorsku funkciju koja konzumira iterator i yielda trenutne procjene srednje vrijednosti i varijance (*ne devijacije*, varijance).
   Primjer: `list(welford([1, 2, 4]))` → `[(1.0, 0.0), (1.5, 0.25), (2.333, 1.556)]`
   Napomena: užasno je lagano naći gotove implementacije u Pythonu, ali probaj ipak napisati svoje. Ja neću znati je li tvoje rješenje kopirano ili nije jer nemam FER-ov detektor plagijata, no računam na tvoj entuzijazam.

Napomena: kod treba biti čitljiv djetetu iz trećeg osnovne sa osnovnim znanjem engleskog jezika. Tvoj kod možda, a možda i ne, bude provjeren s djetetom iz trećeg osnovne.