2-oji užduotis: Supaprastintos blokų grandinės (*Blockchain*) kūrimas

Tikslas

Sukurti **supaprastintą blokų grandinę**, imituojančią jos veikimą realiomis sąlygomis.
Blokų grandinė yra nuoseklus blokų sąrašas, kuriame kiekvienas blokas susietas su ankstesnio bloko maišos reikšme (*hash*).
Kiekvienas blokas turi dvi dalis: **antraštę** (*Header*) ir **turinį** (*Body*).

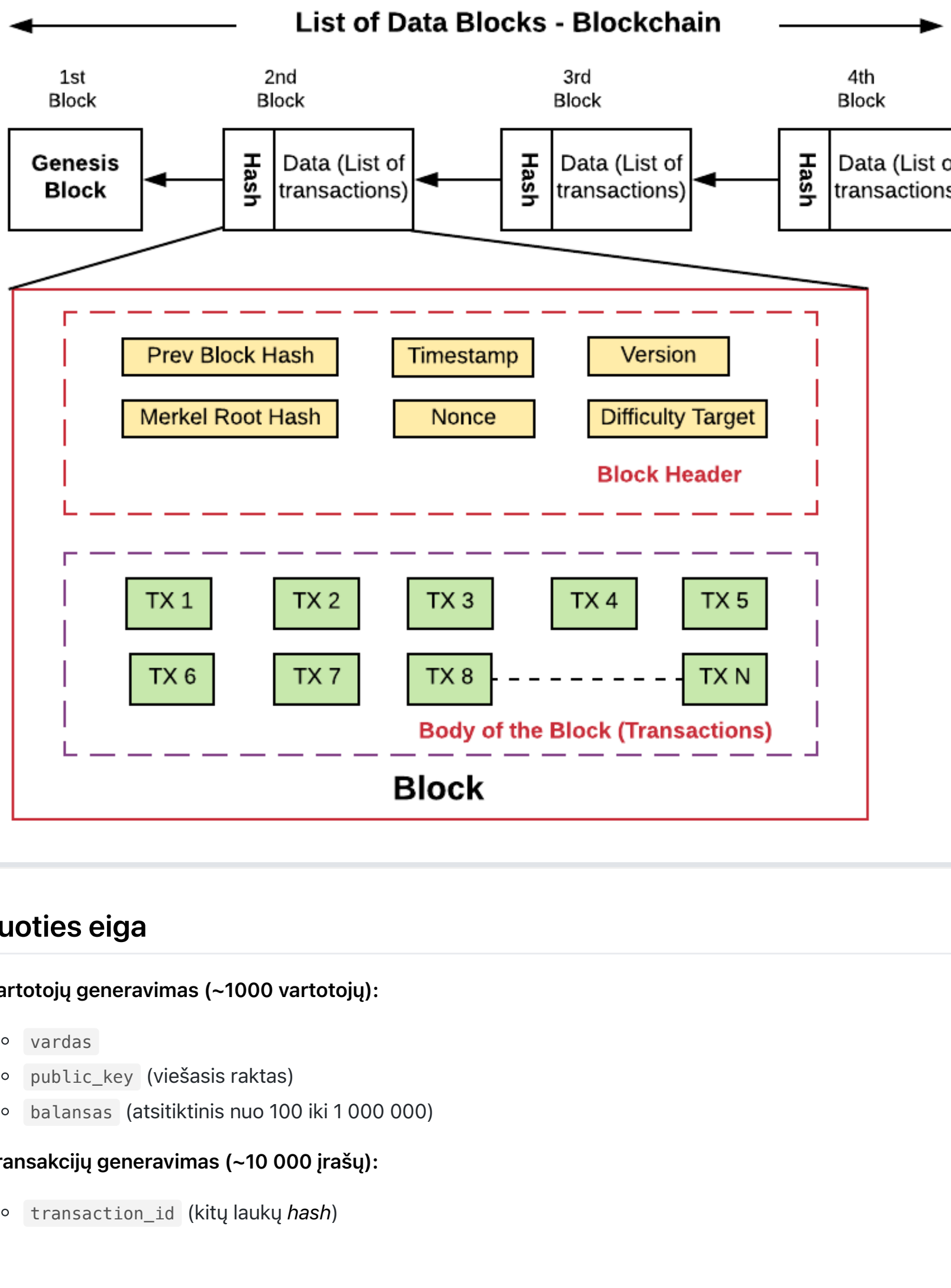
Bloko struktūra

Antraštė (Header):

- Ankstesnio bloko maišos reikšmė (`Prev Block Hash`)
- Laiko žyma (`Timestamp`)
- Duomenų struktūros versija (`Version`)
- Visų transakcijų maišos reikšmė (pvz., *Merkle Root Hash*)
- Atsitiktinis skaičius – `Nonce` , naudojamas *Proof-of-Work* procese
- Sudėtingumo lygis (`Difficulty Target`)

Turinys (Body):

- Transakcijų sąrašas.

> 💡 *Struktūra vaizduojama žemiau:*
>
> 
>

Užduoties eiga

1. **Vartotojų generavimas (~1000 vartotojų):**
   - `vardas`
   - `public_key` (viešasis raktas)
   - `balansas` (atsitiktinis nuo 100 iki 1 000 000)
2. **Transakcijų generavimas (~10 000 įrašų):**
   - `transaction_id` (kitų laukų *hash*)

- `sender` (siuntėjo raktas)
- `receiver` (gavėjo raktas)
- `amount` (siunčiama suma)

> Galite naudoti **sąskaitos modelį** (*account model*) arba **UTXO modelį** (papildomi balai).

3. **Naujo bloko formavimas:**
   - Atsitiktinai pasirinkite 100 transakcijų iš sąrašo.
   - Paruoškite jas įtraukimui į naują bloką.
4. **Bloko kasimas (*Proof-of-Work*):**
   - Tikslas – hash'uojant 6 pagrindinius bloko antraštės (angl. block header) elementus rasti bloko *hash*, prasidedantį bent trimis nuliais (`000...`).
   - Naudokite savo maišos funkciją (iš 1-os užduoties). Jei ji netinkama šiam tikslui – patobulinkite ją, kad tiktų.
5. **Bloko patvirtinimas ir įtraukimas:**
   - Pašalinkite į bloką įtrauktas transakcijas iš sąrašo.
   - Atnaujinkite vartotojų balansus.
   - Pridėkite naują bloką prie grandinės.
6. **Procesų kartojimas:**
   - Kartokite 3–5 žingsnius, kol neliks neįtrauktų transakcijų.

Versijos reikalavimai

🔷 Versija `v0.1` (iki 2025-10-29)

- Sukurta **centralizuota** blokų grandinė.
- Naudojama jūsų sukurta maišos funkcija (modifikuota, jei reikia, kad sugeneruotų `000...` pradžią).
- Transakcijų ir blokų kūrimo procesas **turi būti matomas** (išvedamas į konsolę).
  > Išvedimo kokybė ir vizualumas turės įtakos balui.
  > Pvz. Bitcoin Block Explorer – Block #1
- Vietoje *Merkle Tree* galima naudoti paprastą visų transakcijų ID maišą.
- Paruoškite `README` failą aprašant jūsų blockchain versijos `v0.1` realizaciją, specifiką ar įdomesnius sprendimus (jei yra), taip pat pridedant ekranvaizdžius bei naudojimosi instrukciją.

> ⚙️ Naudokite gerąsias OOP praktikas:
> Enkapsuliacija, konstruktoriai, RAII idiomos taikymas ir aiški klasės struktūra.

🔷 Versija `v0.2` (iki 2025-11-05)

- Įgyvendinkite **Merkle Tree** ir tikrą `Merkle Root Hash`.
- Realizuokite **transakcijų verifikaciją**:
  - ✅ Balanso tikrinimas (siuntėjas negali siųsti daugiau, nei turi)
  - ✅ Transakcijos ID tikrinimas (maišos reikšmės teisingumas)
- Patobulinkite **kasimo procesą**:
  - Sugeneruokite 5 kandidatinius blokus (~100 transakcijų kiekviename)
  - Bandykite juos „kasti“ ribotą laiką (pvz., 5 s) arba iki riboto bandymų skaičiaus
  - Jei nė vienas neiškastas – padidinkite laiką / bandymus ir pakartokite
  - Taip imituojamas **decentralizuotas kasimas**
- Papildykite `README` failą. Taip pat jame aiškiai (geriausiai atskirame skyrelyje) išskirkite kuriems tikslams/žingsniams buvo pasitelkta AI pagalba.

📊 Vertinimas

Individualiai:

- Iki **2.0 balų** – už pagrindinę užduotį.

Vertinimo kriterijai:

- projekto raida (*commit'ai*, *release'ai* atlikti laiku);
- kodo struktūra ir objektinio programavimo principų taikymas (enkapsuliacija, klasės, konstruktoriai ir pan.);

- blokų ir transakcijų kūrimo, kasimo ir validavimo logikos teisingumas;
- tinkamas *Proof-of-Work* mechanizmo veikimas (maišos sudėtingumas, `nonce` iteracijos ir t. t.);
- rezultatų aiškumas ir pateikimas `README.md` faile (paaiškinimai, pavyzdžiai, vizualizacijos);
- projekto veikimo demonstracija (konsolės išvestis, bloko / transakcijos pavyzdžiai);
- plagijavimo (tarp studentų) nebuvimas.

Poromis:

- Iki **2.0 balų** – už pagrindinę užduotį.

Vertinimo kriterijai:

- abiejų narių įnašas matomas per *commit'ų* istoriją;
- projektas išbaigtas ir veikia pagal `v0.1` ir `v0.2` reikalavimus;
- kasimo ir transakcijų validavimo logikos pasidalijimas tarp narių (pvz., vienas realizuoja kasimą, kitas – verifikaciją ar Merkle medį);
- aiškus projekto aprašymas bendrame `README.md` faile (su pavyzdžiais ir išvedimais);
- analogiškai ir papildomų užduočių (UTXO modelis, lygiagretus kasimas) – pasidalijimas.

Papildomi balai

Užduotis	Papildomi balai
UTXO modelio realizavimas	+0.5
Lygiagretus kasimo procesas `v0.2` versijoje	+0.5