Prg3 readme

Käytetyt tietorakenteet

- std::unordered_map<TownID, TownDataMap>
    - Valitsin unordered_mapin pitämään jokaisen kaupungin kaikkea tietoa, jotta etenkin kaikki ”get” metodit, jota käytetään ohjelmassa useasti, olisivat nopeita (1). Toinen suuri syy oli se, että lähes jokaisen funktion ensimmäinen tehtävä oli tarkastaa, löytyykö kaupunkia ollenkaan, ja siinä map on nopea. Mapin sisälle muodostui puu rakenne, jonka avulla pystyi seuraamaan kaupunkien verotusreitteijä.
    - struct TownDataMap
        - Pitää sisällään kaupungin id:n, nimen, koordinaatit, veron, setin osoittimista kaupungin vassaleihin, vektorin joka sisältää kaikkien kaupungin vassalien id:t, osoittimen kaupungin verottajaan sekä boolean arvon siitä, onko kaupunki juuri(juurella ei ole verottajaa).
        - Kolmanne harkkatyön lisäyksenä tänne säilöttiin myös boolean arvot iswhite ja isgrey, joilla tarkkailtiin solmujen tilaa. Samoin lisättiin vektori osoittimista kaupunkeihin johon kulkee tie.
        - vassalien id:t oli tavallaan ”ylimääräinen” tänne, mutta se helpotti huomattavasti vassalien palautusta get_vassals funktiossa.
- std::vector<TownDataVector>
    - Sisältää kaupungin id:n, nimen ja koordinaatit.
    - Tämä tietorakenne on olemassa ainoastaan siksi, että kun kaupunkeja järjestetään joko aakkos tai etäisyysjärjestykseen, voidaan ensin laittaa tämä tietorakenne järjestykseen, ja sen jälkeen laittaa palautusvektori järjestykseen. Toinen vaihtoehto olisi ollut sortata palautusvektoria ja liittää siihen comparefunktio joka tarkastelisi mapin dataa, mutta tällöin funktion nopeudeksi olisi tullut nlogn*logn.
- std::vector<TownID>
    - Tässä tietorakenteessa säilytettiin kaikkien kaupunkien ID:t nopeaa ja yksinkertaista palautusta varten. boolean arvot alphasorted ja distsorted seuraavat tämän tietorakenteen tilaa ja kertovat ohjelmalle, missä järjestyksessä vektori on.