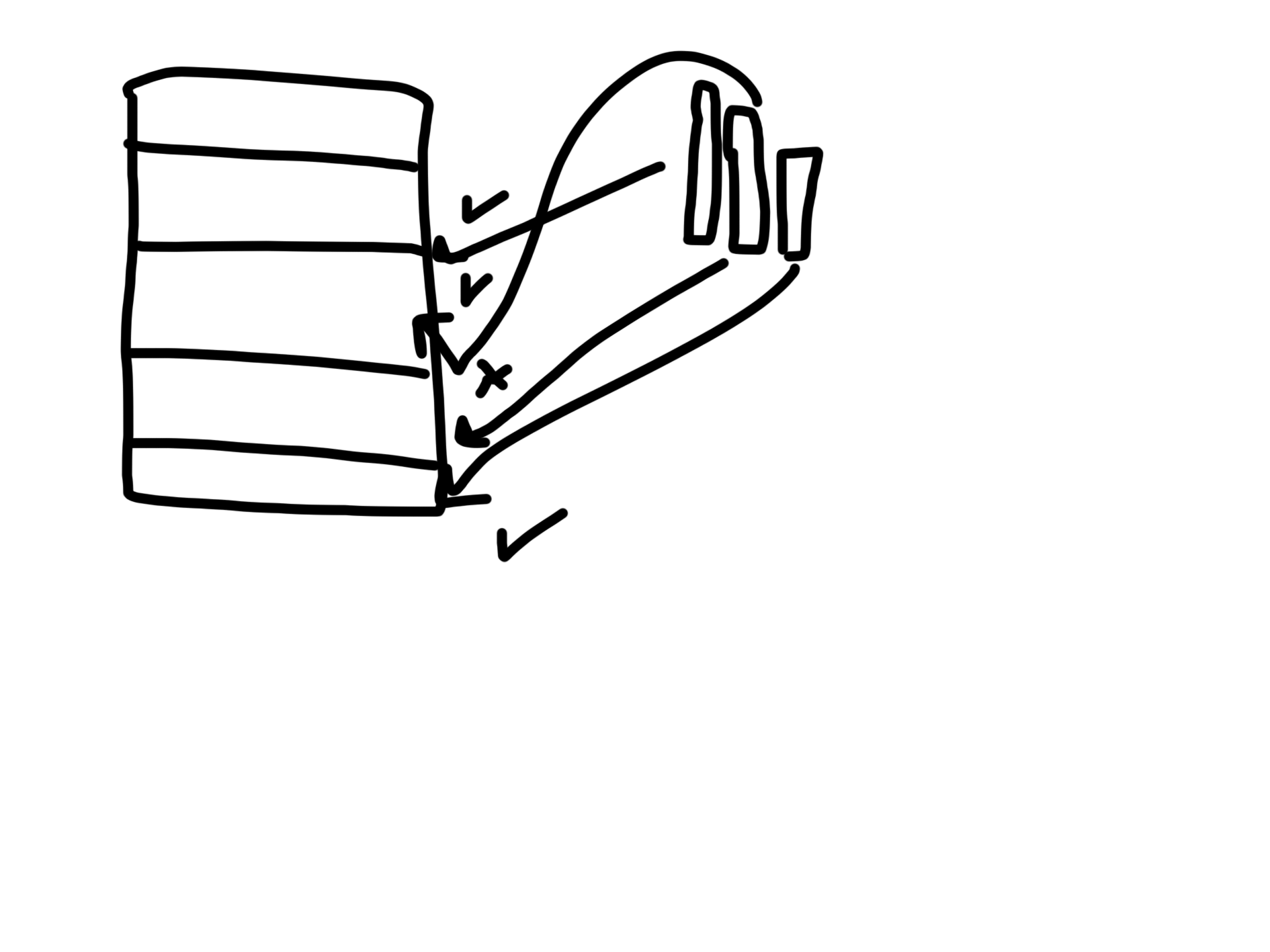
**1. Raamatute järjestamise algoritm**

Tuleb alustada tühja riiuliga. Tuleb luua raamatute loend ja sortida need pikkuse alusel kahanevas järjekorras, kõrgemad raamatud eespool. Tuleb luua muutuja “hetkene riiulikõrgus” ja selle value 0. Seejärel tuleks alustada ükshaaval raamatute läbi vaatamist/töötlemist, tuleb kontrollida, kas riiulikõrgus pluss raamatu kõrgus on väiksem või võrdne riuuli maksimaalse kõrgusega. Kui raamatu lisamine ületaks maksimaalse kõrguse siis “start new shelf” ja resetime “hetkese riiulikõrguse” nulliks.

Kordame seda protsessi kuniks kõik raamatud on sorteeritud. Lõpetame algoritmi.



**2. LIFO/FIFO andmestruktuur**

LIFOStack push ja pop. push lisab elemendi ja pop eemaldab ja tagastab pealmise elemendi. Kirjutasin pythonis selle ilma sisseehitatud teeke kasutades. Saab kasutada järjekordade haldamisel ja algoritmides. See on sarnane hunniku taldrikutega kus sa paned juurde ja võtad ära pealt poolt. Kood eraldi python failis.

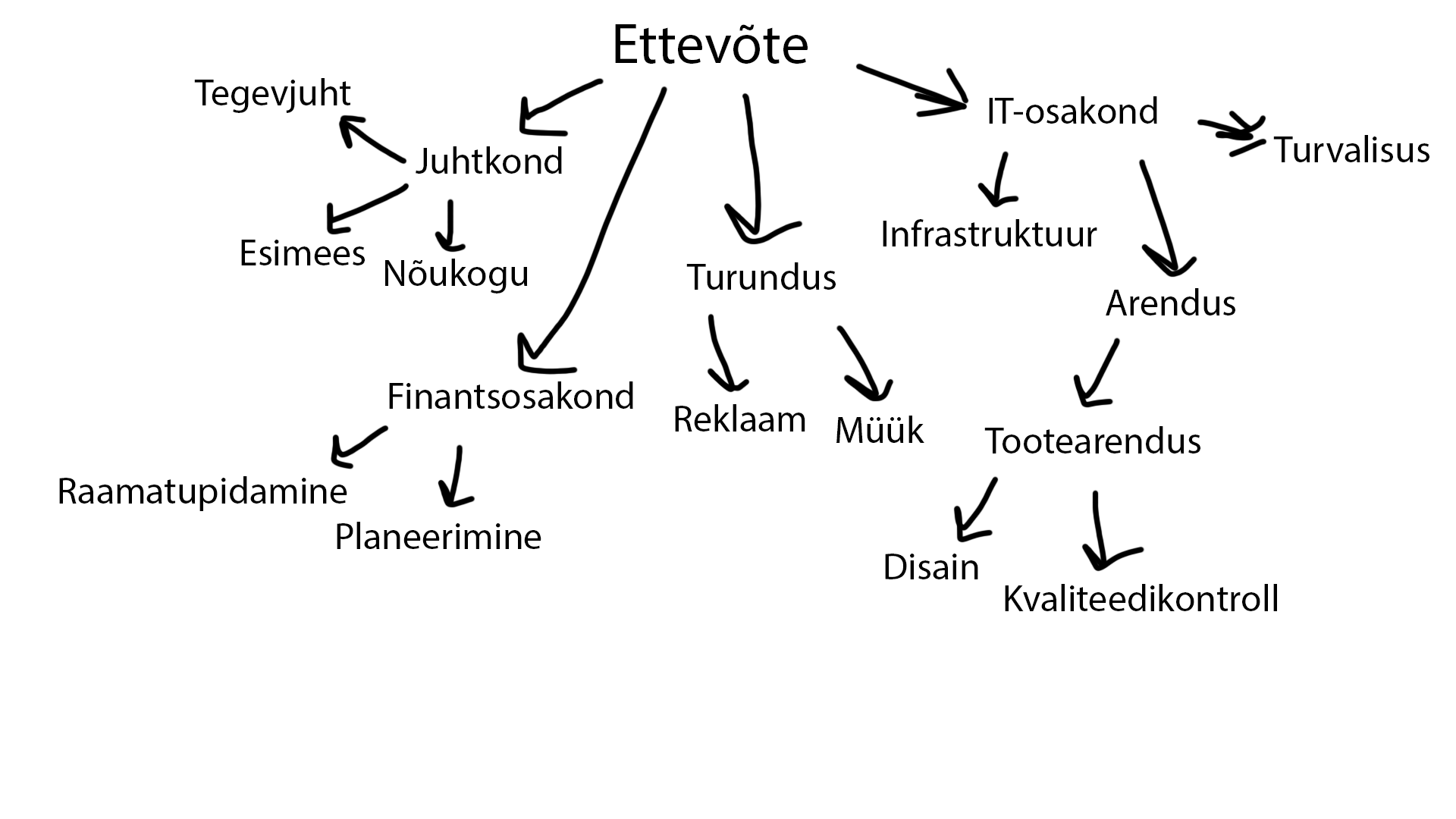
**3. Rekursiivne Fibonacci funktsioon**

Python failis on selgitus.

**4. Binaarne otsingualgoritm**

Loend jagatakse pidevalt kaheks ja võrreldakes “target” väärtusega. Algortimi eeltingimuseks on siseloendi sorteerimine. Põhimõtteliselt funktsioon otsib täisarvu “target” . Leiab siis tagastab indeksi kui ei leia siis tagastab “täisarvu ei leitud”. Kood python failis.

**5. Puu joonistamine**



Nii puude kui graafide puhul on tegemsit matemaatiliste struktuuridega. Puu puhul on oluline hierarhia. Graaf võib sisaldada suvalisi suhteid sõltumata hierarhiast. Graafide abil saab luua keerukaid seoseid. Tee kaardid on hea näide graafide puhul kus erinevad linnad on ühendatud teedega.