VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS

FUNDAMENTINIŲ MOKSLŲ FAKULTETAS

INFORMACINIŲ TECHNOLOGIJŲ KATEDRA

**LABORATORINIS DARBAS 1**

Atliko: DGTfm 22 stud.

Matas Malickas

Priėmė: doc. dr. Andrej Bugajev

Vilnius, 2023

Darbo eiga

Užduoties numerio apskaičiavimas

Pagal formulę ((n − 1) mod 4) + 1, priskyrus n = 8, buvo apskaičiuotas užduoties skaičius: 4.

Map funkcionalumo programavimas

Map funkcionalumo įgyvendinimui, buvo sukurtas map.py failas, kuris geba išrinkti reikalingus duomenis tolimesniam rūšiavimui ir reduce operacijai. Šis failas turi pagrindini ciklą, kuris iteruoja per visas pilnas duomenų failo eilutes, panaikina nereikalingus tarpus, paima eilutės duomenis be pradinių ir galinių duomenų atskirimo ženklų ir suskirsto esamą eilutę į atskirus duomenų objektų įrašus.

for line in sys.stdin:

line = line.strip()

line = line[2:len(line)-2]

stripped\_string = line.split('}}{{')

Tada esamame cikle yra kuriamas dar vienas ciklas ir iteruojama per išskirstytus pilnus duomenų objektus. Visi duomenų įrašai yra atskiriami per skiriamuosius ženklus, paruošiami kintamieji, kurie turės būt ištraukti iš duomenų įrašo.

for data\_entry in stripped\_string:

parsed\_strings = data\_entry.split('}{')

map\_key = None

map\_value\_1 = None

map\_value\_2 = None

map\_value\_3 = None

Sekantis ciklas yra skirtas išrinkti pačias raktų ir verčių poras, jei jos egzistuoja objekte. Kai baigiama iteruoti per visas vieno įrašo reikšmes, dabartinis ciklas pasibaigia ir duomenys yra rašomi į standartinę išvestį (failą), jei visos vertės egzistuoja.

for parsed\_string in parsed\_strings:

(key, value) = parsed\_string.split('=')

# Skip iteration if value is empty

if value == '':

continue

if key == main\_field:

map\_key = int(value)

if key == second\_field:

map\_value\_1 = float(value)

if key == third\_field:

map\_value\_2 = int(value)

if key == fourth\_field:

map\_value\_3 = value

if(map\_key != None and map\_value\_1 != None and map\_value\_2 != None and map\_value\_3 != None):

print(f'{map\_key}\t{map\_value\_1}\t{map\_value\_2}\t{map\_value\_3}')

Rūšiavimo funkcionalumo įgyvendinimas

Sekanti programa yra skirta duomenų rūšiavimui. Ji skaito standartinę išvestį (nustatytas failas) bei deda įrašus į atmintyje esantį masyvą. Tada masyvas yra išrūšiuojamas pagal raktą bei visi įrašai yra surašomi į standartinę išvestį (failą).

for parsed\_string in parsed\_strings:

(key, value) = parsed\_string.split('=')

# Skip iteration if value is empty

if value == '':

continue

if key == main\_field:

map\_key = int(value)

if key == second\_field:

map\_value\_1 = float(value)

if key == third\_field:

map\_value\_2 = int(value)

if key == fourth\_field:

map\_value\_3 = value

if(map\_key != None and map\_value\_1 != None and map\_value\_2 != None and map\_value\_3 != None):

print(f'{map\_key}\t{map\_value\_1}\t{map\_value\_2}\t{map\_value\_3}')

Šioje būsenoje duomenys yra paruošti reduce operacijai.

Reduce

Siekiant apskaičiuoti reikiamą statistiką, t.y. svorio, siuntų skaičiaus bei geografinės zonos statistikas, buvo sukurtas red.py failas, kuriame yra atliekama reduce operacija. Ši failas skaito standartinę įvestį (išrūšiuotą faila) ir sudedinėja vertes svorio bei siuntų skaičiaus vertes tol, kol pasikeičia raktas. Kadangi visa įvestis yra išrūšiuota pagal raktus, tai visad užtikrina, jog pasikeitus raktui, jis toliau niekad nepasikartos ir galima išvesti rezultatą.

Kol raktas yra nepasikeitęs, taip pat yra sekamas geografinių zonų dažnis tam tikram maršrutui (raktui). Tai veikia dedant zonas kaip raktus į dictionary ir jų vertes laikant pasikartojimų skaičiumi. Kai raktas pasikeičia, tada yra išvedamos reikšmės, kurios yra: maršrutas, svorių suma, siuntų skaičiaus suma bei geografinės zonos su jų pasikartojimo dažniais.

for parsed\_string in parsed\_strings:

(key, value) = parsed\_string.split('=')

# Skip iteration if value is empty

if value == '':

continue

if key == main\_field:

map\_key = int(value)

if key == second\_field:

map\_value\_1 = float(value)

if key == third\_field:

map\_value\_2 = int(value)

if key == fourth\_field:

map\_value\_3 = value

if(map\_key != None and map\_value\_1 != None and map\_value\_2 != None and map\_value\_3 != None):

print(f'{map\_key}\t{map\_value\_1}\t{map\_value\_2}\t{map\_value\_3}')

Gautas rezultatas su nepilnais duomenimis

Atlikus map, sort ir reduce funkcijas, rezultatas su daliniais duomenimis buvo gautas:

102 20969.199999999993 1883 Z1 311 Z3 5

103 6556.700000000001 179 Z1 32 Z3 34

104 629.4100000000001 123 Z1 53