**Kodutöö nr 4**

**Liisbeth Tatter**

**03.12.2023**

## Ülesanne 1: Räsimine

1. Kirjuta selgitus räsimise (*hashing*) kontseptsioonist - põhiidee ja eesmärk.

Räsimine on ühesuunaline funktsioon, mis muudab mistahes suurusega andmed fikseeritud suuruseks, et andmeid saaks kiiremini kätte.

Andmed salvestatakse võti-väärtus paaridena, kus võtmele rakendatakse räsifunktsioon.

2. Kirjelda hea räsifunktsiooni omadusi ja selgita, miks need on olulised.

Hea räsifunktsioon tagastab sama sisendi korral sama räsi väärtuse.
Hea räsifunktsioon genereerib erinevate sisendite puhul erinevaid räsi väärtusi.
Hea räsifunktsioon jaotab räsi väärtusi ühtlaselt räsitabelis.

3. Selgita kokkupõrgete lahendamise tehnikaid, eriti eraldi aheldamist (separate chaining) ja avatud aadressimist (open addressing).

Kokkupõrge tekib siis, kui kaks võtit genereerivad sama räsi väärtuse, õige tehnika valik sõltub konkreetsest rakendusest ja andmetest.
**Separate chaining:** iga pilu räsitabelis on linkedlist, mis võimaldab mitmel võtmel eksisteerida samas asukohas räsitabelis
See käsitleb põrkeid, hoides andmestruktuuri igas räsitabeli pesas. Seda on lihtne rakendada ja lubab piiramatul hulgal elemente, kuid nõuab lisamälu viidete jaoks ja jõudlus halveneb, kui põrgete arv suureneb.
**Open addressing:** kõik elemendid on salvestatud räsitabelis endas, sekundaarset andmestruktuuri pole. See säästab ruumi ja väldib viite ülekoormust.
See käsitleb põrkeid, leides järgmise pilu räsitabelis, tehakse nii kaua, kuni leitakse tühi koht. Lisamälu ei ole vaja ning ajafunktsioonid on konstantsed, kuid piiratud räsitabeli suurusega ja jõudlus halveneb, kui räsitabel täitub.

**Ülesanne 3: *Separate Chaining* Kokkupõrgete Lahendamiseks**

1. Rakenda *separate chaningut* kasutades *linked-liste*.
2. Võrdle *separate chaningu* efektiivsust *open addressing* meetodiga ajalise ja ruumilise kompleksuse mõttes.
3. Aruta *separate chaning* kasutamise plusse ja miinuseid räsitabelites.

Separate changing plussid on:

- Lihtne rakendada, ei ole vaja muretseda räsitabeli suuruse pärast
- Lubab piiramatul hulgal elemente, elementide arv ei ole piiratud räsitabeli suurusega
- Sisestamine, kustutamine ja otsingutoimingud on efektiivsed
- Suudab hakkama saada kõrge koormusteguriga

Separate changing miinused on:

- Nõuab lisamälu viidete jaoks LinkedListis või massiivides
- Jõudlus halveneb, kui põrgete arv suureneb
- Toimingute tõhusus sõltub räsifunktsiooni kvaliteedist
- Dünaamilise mälu eraldamine

**Ülesanne 4: *Open Addressing* Tehnikate Uurimine**

1. Kirjuta lühike ülevaade avatud aadressimise meetodist kokkupõrgete lahendamisel räsimises.

Avatud aadressimise meetodis on võtme-väärtuse paar, kus iga võti on seotud unikaalse väärtusega. Räsifunktsiooni kasutatakse võtmete kaardistamiseks massiivide aadressidele. Kui tekib põrge, leitakse võtmele uus aadress, seda tehakse nii kaua kuni leitakse tühi koht.

2. Võrdle (teooria) kolme tehnikat: lineaarne otsing (linear probing), ruuduline otsing (quadratic probing) ja topelträsimine (*double hashing*).

| **Lineaarne otsing** | **Ruuduline otsing** | **Topelträsimine** |
|---|---|---|
| Osa avatud aadressimise meetodist | Avatud aadresseerimise variatsioon | Avatud aadresseerimise variatsioon |
| Kui tekib põrge, otsitakse järgmist saadaolevat kohta | Kasutab ruutfunktsiooni tühja koha leidmiseks põrke korral | |
| Määrab iga võtme jaoks algse indeksi | Määrab iga võtme jaoks algse indeksi | Esimene räsifunktsioon määrab iga võtme jaoks algse indeksi |
| Kui tekib põrge, liigub funktsioon järgmisele indeksile, kuni leitakse tühi koht | Kui tekib põrge, liigub funktsioon indeksile, mis on väärtuse ruudu kaugusel | Kui tekib põrge, määrab teine räsifunktsioon võtme jaoks uue indeksi, kuni leitakse tühi koht |

3. Aruta, millistes olukordades iga tehnika oleks kõige efektiivsem.

**Boonusülesanne: Koormustegur ja *Rehashing (Double hashing vs Rehashing)***

1. Selgita, mis on räsitabeli koormustegur ja miks see on oluline.

Koormustegur on räsitabelis salvestatud elementide arvu ja räsitabeli kogumahu suhe, mis määrab millal räsitabelit suurendada.
Tõhususe säilitamiseks heaks koormustegurks peetakse umbes 0.7

2. Rakenda lihtsat ***Rehashingu*** protsessi ja aruta, kuidas see aitab säilitada efektiivset räsitabelit.

Ümberhajutamine on räsitabeli suuruse muutmise protsess, kus kõigile olemasolevatele võtmetele rakendatakse uut räsifunktsiooni. See vähendab koormustegurit ja parandab efektiivsust.

Rehashingu rakendamise sammud on järgmised:

- Määra, millal koormustegur ületab lävendi
- Loo suurema suurusega räsitabel
- Sisesta kõik võtmed vanast räsitabelist uude

3. Analüüsi ***Rehashingu*** mõju räsitabeli jõudlusele.