

**KAUNAS UNIVERSITY OF TECHNOLOGY**

**FACULTY OF INFORMATICS**

**COMPUTER DEPARTMENT**

### Lygiagretaus programavimo Inžinerinis projektas

**Darbą atliko:**

IFF 6/8 grupės studentas

Tadas Laurinaitis

**Darbą vertino**:

Doc. Romas Marcinkevičius

### Darbo užduoties analizė ir sprendimo metodas

Darbo užduotis: Parallel Quicksort algoritmo greičio tyrimas lyginant su sequencial Quicksort algoritmu naudojant skirtingus duomenų ir gijų kiekius.

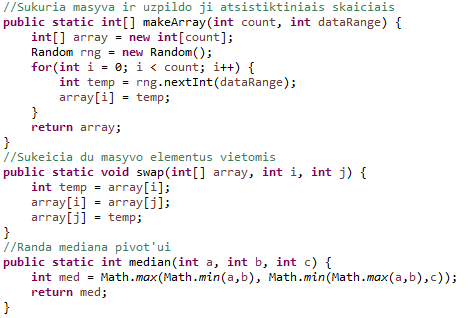
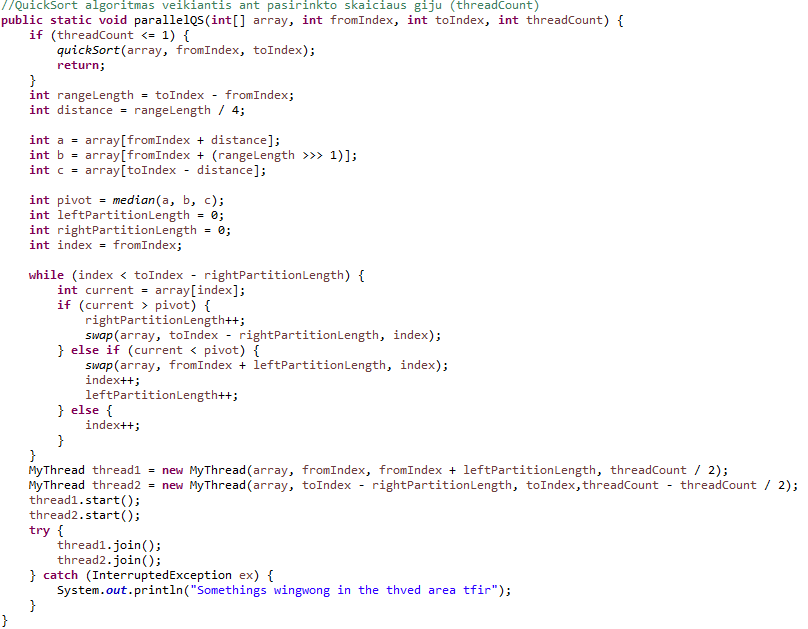
Programinė darbo dalis buvo atlikta su Java programavimo kalba. Rikiuotų duomenų tipas – integer, kiekvienas skaičius buvo

Quicksort algoritmas yra efektyvus rikiavimo algoritmas, paremtas rikiuojamų dalykų lyginimu. Efektyviai implementavus, jis gali būti 2-3 kartus greitesnis negu pagrindiniai jo priešininkai – merge sort ir heapsort.

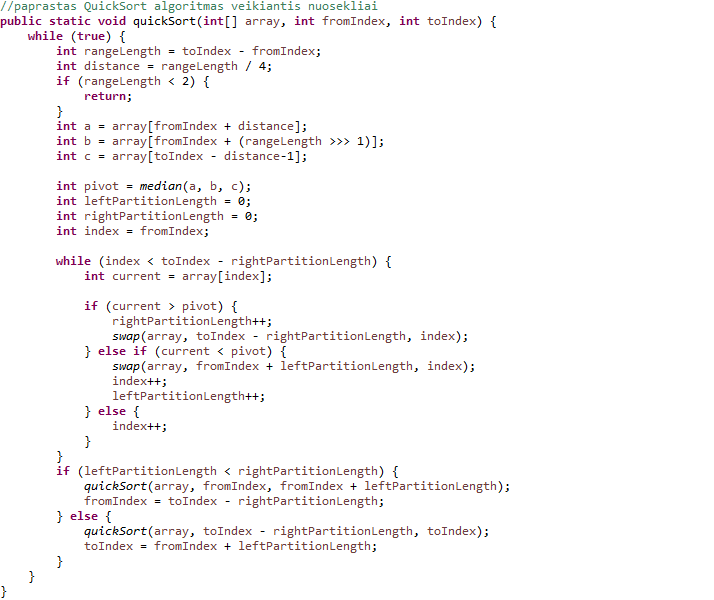
Visu pirma susiradau Quicksort veikimo principus ir pseudo kodą. Po to ieškojau kuom skiriasi lygiagrečiai veikiantis Quicksort nuo nuosekliai veikiančio, bei kaip iš nuosekliai veikiančio algoritmo padaryti lygiagrečiai veikiantį. Susiradęs reikiamą informaciją, bei apytiksliai žinodamas ką reikia padaryti susikūriau naują Java projektą. Tada susikūriau metodą kuris sudarytų masyvą užpildytą atsitiktiniais skaičiais bei metodą kuris rastų trijų skaičių medijaną. Po to padariau nuosekliai veikiantį Quicksort algoritmą, kad turėčiau su kuo lyginti lygiagrečiai veikiantį algoritmą. Po to padariau lygiagrečiai veikiantį Quicksort algoritmą, kurio nemažą dalį kodo radau internete. Galiausiai parašęs visas likusias programos dalis, lyginau gautus programus rezultatus naudojant įvairius duomenų kiekius ir viską surašiau į lentelę. Po to pagal šią lentelę sudariau grafikus.

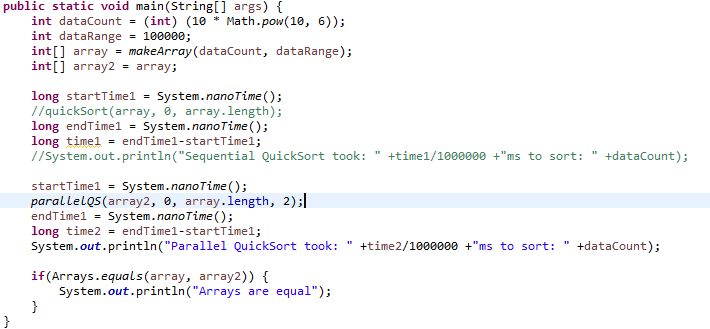
### Programos dalių tekstai ir jų trumpi aprašymai

Pav. Nr. 1 – Gijos klasė, kurią paleidus bus vykdomas parallel Quicksort algortimas ant priskirtos masyvo dalies

Pav. Nr. 2 – Pagalbiniai metodai naudoti sukurti masyvą, apkeisti masyvo elemtus vietomis ir apskaičiuoti Quicksort algoritmo pivot’ą

Pav. Nr. 3 – Lygiagretus Quicksort algoritmas, veikiantis pagal nustatytą gijų skaičių

 Pav. Nr. 4 – Nuosekliai veikiantis Quicksort algoritmas

Pav. Nr. 5 – main metodas kuriame atliekami visi veiksmai

### Vykdymo laiko kitimo tyrimas

Buvo ištirtas programos vykdymo laikas naudojant duomenų kiekius nuo 1 milijono iki 300 milijonų. Svarbu paminėti, kad didesni laiko skirtumai tarp algoritmų vykdytų su skirting skaičiumi gijų galėjo atsirasti dėl skirtingos kompiuterio apkrovos vykdymo metu.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Duomenų kiekis | Laikų vidurkiai(ms) | | | |
| Nuoseklus QS | Lygiagretus QS su 2 gijom | Lygiagretus QS su 4 gijom | Lygiagretus QS su 6 gijom |
| 1000000 | 100 | 25 | 95 | 80 |
| 5000000 | 480 | 100 | 236 | 280 |
| 10000000 | 980 | 190 | 450 | 496 |
| 15000000 | 1410 | 284 | 640 | 680 |
| 20000000 | 1960 | 370 | 830 | 896 |
| 25000000 | 2370 | 465 | 1130 | 1080 |
| 30000000 | 2950 | 560 | 1290 | 1310 |
| 35000000 | 3220 | 620 | 1570 | 1630 |
| 40000000 | 3815 | 759 | 1664 | 1710 |
| 45000000 | 4115 | 800 | 1881 | 1950 |
| 50000000 | 4530 | 900 | 2242 | 2321 |
| 100000000 | 8930 | 1686 | 3930 | 3877 |
| 150000000 | 14863 | 2889 | 5915 | 5733 |
| 200000000 | 19808 | 3955 | 8735 | 7789 |
| 250000000 | 24843 | 5180 | 12454 | 11077 |
| 300000000 | 29429 | 5822 | 16394 | 14875 |

Lentelė Nr. 1 Kiekvieno algoritmo su atitinkamu skaičiumi gijų vidutinis veikimo laikas esant tam tikram skaičiui duomenų

Grafikas Nr. 1 Nuoseklaus QS vykdymo laiko palyginimas su lygiagrečiu QS naudojančiu 2 gijas

Grafikas Nr. 2 Nuoseklaus QS vykdymo laiko palyginimas su lygiagrečiu QS naudojančiu 4 gijas

Gafikas Nr. 3 Nuoseklaus QS vykdymo laiko palyginimas su lygiagrečiu QS naudojančiu 6 gijas

Grafikas Nr. 4 Nuoseklaus QS vykdymo laiko palyginimas su kiekvienu lygiagrečiu QS naudojančiu nuo 2 iki 6 gijų

Grafikas Nr. 5 Nuoseklas QS laiko palyginimas su kiekvienu lygiagrečiu QS, naudojant labai didelius duomenų kiekius (50-300milijonų)

### Išvados ir literatūros sąrašas

Išvados: Programa lėčiausiai veikė nenaudojant lygiagretaus Quicksort algoritmo, o greičiausiai veikė naudojant lygiagretų Quicksort algoritmą ir 2 gijas. Veikimo laiko skirtumas, tarp lygiagretaus ir nelygiagretaus algoritmo didėjo, didinant duomenų kiekį ir buvo didžiausias naudojant 2 gijas, bei mažiausias naudojant 4 gijas. Naudojant 6 gijas laiko skirtumas buvo kiek mažesnis nei naudojant 4 gijas.

Literatūros sąrašas:

<https://en.wikipedia.org/wiki/Quicksort>

<https://stackoverflow.com/questions/3425126/java-parallelizing-quick-sort-via-multi-threading>

<https://codereview.stackexchange.com/questions/121996/parallel-integer-quicksort-in-java>

<https://www.youtube.com/watch?v=dD4ls9cLnMk>

<https://www.youtube.com/watch?v=Jluv6uMwv3Q>