|  |
| --- |
| **KAUNO TECHNOLOGIJOS UNIVERSITETAS**  **INFORMATIKOS FAKULTETAS** |
| **Modulio P175B014 „Duomenų struktūros“**  Laboratorinio darbo aprašas (ataskaita)  **Antrasis laboratorinis darbas** |
| **Dėstytojas**  Giedrius Paulikas |
| **Studentas**  Nedas Liaudanskis |
| **KAUNAS, 2022** |

Turinys

[Tiriamieji metodai 2](#_Toc118845640)

[Greitaveikos testavimo metodika 3](#_Toc118845641)

[Kompiuterio parametrai 5](#_Toc118845642)

[Greitaveikos testavimo rezultatai 5](#_Toc118845643)

[Išvados 6](#_Toc118845644)

# Tiriamieji metodai

|  |  |  |
| --- | --- | --- |
| **9** | **Class AvlSet: contains()** | **Class java.util.TreeSet: contains()** |

Pirmiausia atkreipiau dėmesį į taip kaip abu metodai yra aprašyti. Class java.util.TreeSet: contains() metodas atrodo gan paprastas, tikrina ar yra tam tikras metodas sąraše su tam tikru atitikmeniu atsineštam Object o metodui. Jeigu metode atsineštas elementas yra null, tiesiog gražina null, o jeigu duoti duomenys yra tokie kurių lyginti negalimą išmeta ClassCastException ir nutraukia darbą.

Tuo tarpu Class AvlSet: contains() metodas pirmiausia išsiaiškina ar atsineštas E element nėra null, jeigu pasitaiko atvejis, jog jis yra null, yra išmetama IllegalArgumentException su tekstu „Element is null in contains(E element)“, kitu atveju metodas tęsia darbą ir iš duoto seto paima elementą E element, jeigu setas neturi tokio elemento tai gražinamas false, kitu atveju true.

Paveikslėlis, kuriame yra žinutė

Automatiškai sugeneruotas aprašymas

**pav. 1 Java TreeSet Contains() method**

Paveikslėlis, kuriame yra žinutė

Automatiškai sugeneruotas aprašymas

**pav. 2 AvlSet Contains() method**

# Greitaveikos testavimo metodika

Norint atlikti greitaveikos testus reikėjo apsirašyti kelis papildomus dydžius: AvlSet<Car> carSetAvl; TreeSet<Car> Javaset; Car element (Car element2 buvo naudotas kitiems testavimams). Kaip buvo sukurti šie dydžiai, reikėjo surasti elementą, kurį naudosime metode Contains() norint patikrinti ar tam tikras setas turi tą objektą Car element, taip pat užpildyti AvlSet<Car> ir TreeSet<Car> medžius. Užpildymas vyko gan paprastai, kiekvienas iteracijai buvo sugeneruotas tam tikras skaičius atsitiktinių elementų (kas 10000 didėjantis skaičius iki 80000, po kiekvieno testo), vienas iš tų elementų atsitiktine tvarka buvo išrinktas tapti tuo kurio ieškosime naudodami Contains() metodą. Kadangi elementas E element buvo randamas atsitiktine tvarka į greitaveikos testus galėjo papulti variacijų kai ieškomas elementas buvo rastas labai greitai, palyginus su kitais kartais, kai operaciją reikėjo atlikti einant per visą metį.

**Paveikslėlis, kuriame yra žinutė

Automatiškai sugeneruotas aprašymas**

**pav. 3 Greitaveikos pasiruošimas**

Paveikslėlis, kuriame yra žinutė

Automatiškai sugeneruotas aprašymas

**pav. 4 Greitaveikos metodai**

# Kompiuterio parametrai

Greitaveikos tyrimai buvo įvykdyti naudojant nešiojamą kompiuterį Legion 5 17ACH6H

Su tiesioginiu krovimu iš elektros lizdo.

* AMD Ryzen 7 5800H with Radeon Graphics 3.20 GHz
* RAM 16,0 GB
* Windows 10

# Greitaveikos testavimo rezultatai

|  |
| --- |
| Benchmark (elementCount) Mode Cnt Score Error Units  Benchmark.Contains 10000 avgt 5 1,126 ± 0,478 us/op  Benchmark.Contains 20000 avgt 5 2,527 ± 4,402 us/op  Benchmark.Contains 30000 avgt 5 3,750 ± 6,271 us/op  Benchmark.Contains 40000 avgt 5 5,556 ± 1,175 us/op  Benchmark.Contains 50000 avgt 5 7,280 ± 5,037 us/op  Benchmark.Contains 60000 avgt 5 9,536 ± 8,776 us/op  Benchmark.Contains 70000 avgt 5 9,504 ± 2,431 us/op  Benchmark.Contains 80000 avgt 5 9,944 ± 3,289 us/op  Benchmark.ContainsJava 10000 avgt 5 1,045 ± 1,722 us/op  Benchmark.ContainsJava 20000 avgt 5 1,540 ± 0,958 us/op  Benchmark.ContainsJava 30000 avgt 5 3,177 ± 5,613 us/op  Benchmark.ContainsJava 40000 avgt 5 6,217 ± 6,179 us/op  Benchmark.ContainsJava 50000 avgt 5 8,267 ± 8,310 us/op  Benchmark.ContainsJava 60000 avgt 5 8,874 ± 7,126 us/op  Benchmark.ContainsJava 70000 avgt 5 10,633 ± 5,042 us/op  Benchmark.ContainsJava 80000 avgt 5 9,604 ± 1,862 us/op |

# Išvados

Iš greitaveikos tyrimų rezultatų galiu teigti, jog metodų AvlSet: contains(), TreeSet<E> contains() greitaveika yra labai panaši, skyrėsi tik tuose dalyse, kur papuolė atsitiktinis skaičius esantis medžio pradžioje, tada metodas buvo atliktas greičiau. Mano manymu su greitaveikos metodu abu metodai susitvarkė gan gerai, tačiau Java.TreeSet<E> contains() metodas turi O(log (n) sudėtingumą, tuo tarpu AvlSet: contains() metodo sudėtingumas atitinka O(n). Taigi, dėl didelio atsitiktinių skaičių skaičiaus ir atsitiktinumo negalėjau pamatyti didelio greitaveikos skirtumo tarp abiejų metodų.