

VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS

FUNDAMENTINIŲ MOKSLŲ FAKULTETAS

INFORMACINIŲ SISTEMŲ KATEDRA

**Algoritmai ir duomenų struktūros**

Savarankiškas darbas Nr. 3

Atliko: PRIfs 19/4 grupės stud.

Ignas Dūdėnas

Priėmė: dėst. Nijolė Čeikienė

Vilnius, 2020

Turinys

[**Įterpimo (angl. Insertion sort) ir Šelo (Shell sort) rūšiavimo algoritmų tyrimas** 4](#_Toc58341193)

[**Užduotis:** 4](#_Toc58341194)

[**Algoritmai:** 4](#_Toc58341195)

[**Teoriniai įverčiai:** 4](#_Toc58341196)

[**Tyrimas:** 5](#_Toc58341197)

[**Sudėtingumo pagal sukeitimus ir palginimus tyrimas:** 5](#_Toc58341198)

[**Vidutinis atvejis** 5](#_Toc58341199)

[**Geriausias atvejis** 9](#_Toc58341200)

[**Blogiausias atvejis** 12](#_Toc58341201)

[**Sudėtingumo pagal laiką tyrimas:** 15](#_Toc58341202)

[**Vidutinis atvejis** 15](#_Toc58341203)

[**Geriausias atvejis** 17](#_Toc58341204)

[**Blogiausias atvejis** 19](#_Toc58341205)

[**Išvados:** 21](#_Toc58341206)

[**Bandymų paveiksliukai:** 22](#_Toc58341207)

[**Sudėtingumo pagal sukeitimus tyrimas:** 22](#_Toc58341208)

[**Vidutinis atvėjis:** 22](#_Toc58341209)

[**Geriausias atvėjis:** 24](#_Toc58341210)

[**Blogiausias atvejis:** 26](#_Toc58341211)

[**Sudėtingumo pagal laiką tyrimas:** 28](#_Toc58341212)

[**Vidutinis atvejis:** 28](#_Toc58341213)

[**Geriausias atvejis:** 30](#_Toc58341214)

[**Blogiausias atvejis:** 32](#_Toc58341215)

[**Programos kodas:** 34](#_Toc58341216)

**Lentelių ir paveikslėlių turinys**

[**lentelė 1** Įterpimo ir Šelo algoritmų teoriniai įverčiai 5](#_Toc58342086)

[**lentelė 2** Vidutinio atvėjo tyrimo rezultatai 6](#_Toc58342087)

[**pav. 1** Įterpimo algoritmo palyginimų skaičiaus kitimas didinant masyvo narių skaičių (vidutiniu atveju) 7](#_Toc58342088)

[**pav. 2** Šelo algoritmo palyginimų skaičiaus kitimas didinant maasyvo narių skaičių (vidutiniu atveju) 8](#_Toc58342089)

[**pav. 3** Įterpimo algoritmo sukeitimų skaičiaus kitimas didinant masyvo narių skaičių (vidutiniu atveju) 9](#_Toc58342091)

[**pav. 4** Šelo algoritmo sukeitimų skaičiaus kitimas didinant masyvo narių skaičių (vidutiniu atveju) 9](#_Toc58342092)

[**lentelė 3** Geriausio atvejo tyrimo rezultatai 10](#_Toc58342093)

[**pav. 5** Įterpimo algoritmo palyginimų skaičiaus kitimas didinant masyvo narių skaičių (geriausiu atveju) 11](#_Toc58342094)

[**pav. 6** Šelo algoritmo palyginimų skaičiaus kitimas didinant maasyvo narių skaičių (geriausiu atveju) 11](#_Toc58342095)

[**pav. 7** Įterpimo algoritmo sukeitimų skaičiaus kitimas didinant masyvo narių skaičių (geriausiu atveju) 12](#_Toc58342097)

[**pav. 8** Šelo algoritmo sukeitimų skaičiaus kitimas didinant masyvo narių skaičių (geriausiu atveju) 12](#_Toc58342098)

[**lentelė 4** Blogiausio atvėjo tyrimo rezultatai 13](#_Toc58342099)

[**pav. 9** šelo algoritmo palyginimų skaičiaus kitimas didinant masyvo narių skaičių (geriausiu atveju) 14](#_Toc58342100)

[**pav. 10** Šelo algoritmo palyginimų skaičiaus kitimas didinant maasyvo narių skaičių (geriausiu atveju) 14](#_Toc58342101)

[**pav. 11** Įterpimo algoritmo sukeitimų skaičiaus kitimas didinant masyvo narių skaičių (geriausiu atveju) 15](#_Toc58342103)

[**pav. 12** Šelo algoritmo sukeitimų skaičiaus kitimas didinant masyvo narių skaičių (geriausiu atveju) 15](#_Toc58342104)

[**lentelė 5** Vidutinio atvėjo tyrimo rezultatai 16](#_Toc58342105)

[**pav. 13** Įterpimo algoritmo skaičiavimo laiko kitimas didinant masyvo narių skaičių 17](#_Toc58342106)

[**pav. 14** Šelo algoritmo skaičiavimo laiko kitimas didinant masyvo narių skaičių 17](#_Toc58342107)

[**lentelė 6** Geriausio atvėjo tyrimo rezultatai 18](#_Toc58342108)

[**pav. 15** Įterpimo ir šelo algoritmų skaičiavimo laiko kitimas didinant masyvo narių skaičių 19](#_Toc58342109)

[**lentelė 7** Blogiausio atvėjo tyrimo rezultatai 20](#_Toc58342110)

[**pav. 16** Įterpimo algoritmo skaičiavimo laiko kitimas didinant masyvo narių skaičių 21](#_Toc58342111)

[**pav. 17** Šelo algoritmo skaičiavimo laiko kitimas didinant masyvo narių skaičių 21](#_Toc58342112)

# **Įterpimo (angl. Insertion sort) ir Šelo (Shell sort) rūšiavimo algoritmų tyrimas**

## **Užduotis:**

**Variantas 5**

1. Realizuoti duomenų rūšiavimą partasto įterpimo (Insertion sort) ir Šelo (Shell sort) rūšiavimo algoritmais. Algoritmai duomenis turi rūšiuoti didėjimo tvarka. Studentas turi naudoti orginalią atstumų seką šelo algoritmui.
2. Atlikti realizuotų algoritmų analizę:
   * Nustatyti palyginimų ir sukeitimų kiekį, vykdymo greitį geriausiu, blogiausiu ir vidutiniu atvėjais. Nagrinėti bent 4 skirtingus duomenų kiekius.
   * Atlikti testavimo rezultatų analizę, palyginti ezultatus su teoriniais įverčiais.
   * Testavimo rezultatus pateikti lentele ir pavaizduoti grafiškai.
3. Pateikti išvadas.

## **Algoritmai:**

Įterpimo algoritmas

Pirmasis masyvo elementas yra praleidžiamas kaip jau surušiuotas elementas. Tuomet imamas sekantis elementas ir išsaugomas kaip laikinas, tuomet lyginamas su praeitu elementu, jei jis mažesnis į jo vietą įrašomas prieš jį esantis elementas ir tai daroma tol kol pasiekiamas už jį didesnis arba lygus elementas, tada elementas išsaugotas kaip laikinasis įrašomas atgal į masyvą. Taip algoritmas yra kartojamas kol visi elementai atsiduria savo vietoje.

Šelo algoritmas

Tarp lyginamų elementų yra fiksuotas atstumas („gap“). Pradinis atstumas dažniausiai imamas lygus pusei masyvo elementų (n/2). Tuomet yra lyginami masyvo elementai atskirtu tokiu atstumu. Kai su einamuoju atstumu nebeatliekama daugiau pakeitimų, jis mažinamas dar per pusę. Šis procesas tęsiamas, kol atstumas pasidaro 1 ir nebepakeičiama nė viena pora elementų.

## **Teoriniai įverčiai:**

**lentelė 1** Įterpimo ir Šelo algoritmų teoriniai įverčiai

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Algoritmai | Laiko sudėtingumas | | | Vietos sud. |
| Geriausias | Vidutinis | Blogiausias | Blogiausias |
| Įterpimo | O(n) - palyginimų  O(1) - sukeitimų | O(n2) - palyginimų  O(n2) - sukeitimų | O(n2) - palyginimų  O(n2) - sukeitimų | O(1) |
| Šelo | O(n\*log n) - palyginimų  O(n\* log2 n) - sukeitimų | O(n4/3) - palyginimų  O(n1,25) - sukeitimų | O(n2) - palyginimų  O(n1,25) - sukeitimų | O(1) |

# **Tyrimas:**

Buvo tirti trys atvėjai:

* Vidutinis atvejis: visimasyvo elementai sugeneruoti atsitiktiniu būdu,
* Blogiausias atvejis: masyvo elementai surašyti atvirkščia tvarka nei reikia surūšiuoti,
* Geriausias atvejis: kai visi masyvo elementai surūšiuoti išskyrus vieną.

Kiekvienu atveju buvo atlikti 5 bandymai, elementai buvo rūšiuojami didėjimo tvarka.

## **Sudėtingumo pagal sukeitimus ir palginimus tyrimas:**

### **Vidutinis atvejis**

Iš 1 lentelės matyti, kad įterpimo algoritmo teoriniai įverčiai vidutiniu atveju yra O(n2) palyginimų ir O(n2) sukeitimų, o šelo algoritmo - O(n4/3) palyginimų ir O(n1,25) sukeitimų.

Vidutinio atvejo rezultatai, kai buvo atlikti bandymai su n = 50, n = 200, n = 500, n = 2000 pateikiami 2 lentelėje.

**lentelė 2** Vidutinio atvėjo tyrimo rezultatai

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Algoritmai** | **Bandymas, kai n = 50.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 634 – lyg.  49 – keit. | 734 – lyg.  49 – keit. | 594 – lyg.  49 – keit. | 599 – lyg.  49 – keit. | 549 – lyg.  49 – keit. | 622 – lyg.  49 – keit. |
| **Šelo** | 396 – lyg.  193 – keit. | 337 – lyg.  134 – keit. | 341 – lyg.  138 – keit. | 400 – lyg.  197 – keit. | 394 – lyg.  191 – keit. | 373 – lyg.  170 – keit. |
|  | **Bandymas, kai n = 200.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 9161 – lyg.  199 – keit. | 9953 – lyg.  199 – keit. | 10053 – lyg.  199 – keit. | 10512 – lyg.  199 – keit. | 9791 – lyg.  199 – keit. | 9894 – lyg.  199 – keit. |
| **Šelo** | 2402 – lyg.  1199 – keit. | 2127 – lyg.  924 – keit. | 2398 – lyg.  1195 – keit. | 2348 – lyg.  1145 – keit. | 2305 – lyg.  1102 – keit. | 2316 – lyg.  1113 – keit. |
|  | **Bandymas, kai n = 500.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 64601 – lyg.  499 – keit. | 62300 – lyg.  499 – keit. | 62787 – lyg.  499 – keit. | 60716 – lyg.  499 – keit. | 64273 – lyg.  499 – keit. | 62935 – lyg.  499 – keit. |
| **Šelo** | 6997 – lyg.  3491 – keit. | 6330 – lyg.  2824 – keit. | 6785 – lyg.  3279 – keit. | 6383 – lyg.  2877 – keit. | 6595 – lyg.  3089 – keit. | 6618 – lyg.  3112 – keit. |
|  | **Bandymas, kai n = 2000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 1002584 – lyg.  1999 – keit. | 1025972 – lyg.  1999 – keit. | 1008068 – lyg.  1999 – keit. | 993659 – lyg.  1999 – keit. | 990741 – lyg.  1999 – keit. | 1004204 – lyg.  1999 – keit. |
| **Šelo** | 35901 – lyg.  17895 – keit. | 37921 – lyg.  19915 – keit. | 36984 – lyg.  18978 – keit. | 36434 – lyg.  18428 – keit. | 36769 – lyg.  18763 – keit. | 36801 – lyg.  18795 – keit. |

**Rezultatų analizė:**

Įterpimo algoritmas yra stabilus, nes sukeitimu sukeitimu skaičius nesiskiria skirtinguose bandymuose, nors palyginimu skaičiai visuomet skiriasi, tačiau nedaug. Šelo algoritmas nėra stabilus, kadangi skaičius visuomet skiriasi. Pavyzdziui, kai n = 50, tai sukeitimų skaičius buvo nuo 134 iki 197, skirtumas – 63, tai yra daugiau neigu trekšdalis didziausio skaičiaus.

Įterpimo ir Šelo algoritmų palyginimų rezultatai labai skiriasi, kadangi teoriniai įverčiai skiriasi. Įterpimo yra O(n2) – palyginimų, o Šelo yra O(n4/3) – palyginimų.

Masyvo narių skaičius kyla nuo 50 -> 200, tai 50\*4, 200->500, tai 200\*2,5, 500->2000, tai 500\*4.

Pagal teorinį modelį įterpimo metodu O(n2) – palyginimų skaičius kyla kvadratu.

622 \* 42 = 9952 ~ 9894, 9894 \* 2,52 = 61838 ~ 62935, 62935 \*42 = 1006960 ~ 1004204.

**pav. 1** Įterpimo algoritmo palyginimų skaičiaus kitimas didinant masyvo narių skaičių (vidutiniu atveju)

Pagal teorinį modelį šelo metodu O(n4/3) – palyginimų skaičius kyla n4/3.

373 \* 44/3 = 2368 ~ 2316, 2316 \* 2,54/3 = 7858 ~ 6618, 6618 \*44/3 = 42021 ~ 36801.

**pav. 2** Šelo algoritmo palyginimų skaičiaus kitimas didinant maasyvo narių skaičių (vidutiniu atveju)

Abiejų algoritmų sukeitimų rezultatai atitinka teorinius modelius: įterpimo O(n2) , šelo O(n1,25).Įterpimo algoritmo sukeitimų kiekis kito tiesiškai – tačiau didinant masyvo narių skaičių, keitimų skaičius nekilo kvadratiškai (49 → 199 → 499 → 1999). Šelo algoritmo rezultatai kito tolygiai pagal n1.25 formule didinant masyvo narių skaičių taip pat didėjo ir keitimų skaičius (170 → 1113 → 3112 → 18795).

**pav. 3** Įterpimo algoritmo sukeitimų skaičiaus kitimas didinant masyvo narių skaičių (vidutiniu atveju)

**pav. 4** Šelo algoritmo sukeitimų skaičiaus kitimas didinant masyvo narių skaičių (vidutiniu atveju)

### **Geriausias atvejis**

Iš 1 lentelės matyti, kad įterpimo algoritmo teoriniai įverčiai geriausiu atveju yra O(n) palyginimų ir O(1) sukeitimų, o šelo algoritmo - O(n\*log n) palyginimų ir O(n1,25) sukeitimų.

Geriausio atvejo rezultatai, kai buvo atlikti bandymai su n = 50, n = 200, n = 500, n = 2000 pateikiami 3 lentelėje.

**lentelė 3** Geriausio atvejo tyrimo rezultatai

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Algoritmai** | **Bandymas, kai n = 50.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 87 – lyg.  49 – keit. | 67 – lyg.  49 – keit. | 70 – lyg.  49 – keit. | 64 – lyg.  49 – keit. | 69 – lyg.  49 – keit. | 71 – lyg.  49 – keit. |
| **Šelo** | 242 – lyg.  39 – keit. | 221 – lyg.  18 – keit. | 224 – lyg.  21 – keit. | 225 – lyg.  22 – keit. | 249 – lyg.  16 – keit. | 226 – lyg.  23 – keit. |
|  | **Bandymas, kai n = 200.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 299 – lyg.  199 – keit. | 247 – lyg.  199 – keit. | 257 – lyg.  199 – keit. | 257 – lyg.  199 – keit. | 255 – lyg.  199 – keit. | 261 – lyg.  199 – keit. |
| **Šelo** | 1293 – lyg.  90 – keit. | 1204 – lyg.  1 – keit. | 1220 – lyg.  17 – keit. | 1229 – lyg.  26 – keit. | 1222 – lyg.  19 – keit. | 1233 – lyg.  30 – keit. |
|  | **Bandymas, kai n = 500.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 601 – lyg.  499 – keit. | 544 – lyg.  499 – keit. | 522 – lyg.  499 – keit. | 538 – lyg.  499 – keit. | 546 – lyg.  499 – keit. | 550 – lyg.  499 – keit. |
| **Šelo** | 3592 – lyg.  86 – keit. | 3591 – lyg.  85 – keit. | 3597 – lyg.  91 – keit. | 3596 – lyg.  90 – keit. | 3599 – lyg.  93 – keit. | 3595 – lyg.  89 – keit. |
|  | **Bandymas, kai n = 2000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 3132 – lyg.  1999 – keit. | 2795 – lyg.  1999 – keit. | 2807 – lyg.  1999 – keit. | 2815 – lyg.  1999 – keit. | 2805 – lyg.  1999 – keit. | 2870 – lyg.  1999 – keit. |
| **Šelo** | 19334 – lyg.  1328 – keit. | 18726 – lyg.  720 – keit. | 18766 – lyg.  760 – keit. | 18763 – lyg.  757 – keit. | 18733 – lyg.  727 – keit. | 18864 – lyg.  858 – keit. |

**Rezultatų analizė:**

Įterpimo algoritmas yra stabilus, nes sukeitimu sukeitimu skaičius nesiskiria skirtinguose bandymuose, nors palyginimu skaičiai visuomet skiriasi, tačiau nedaug. Šelo algoritmas nėra stabilus, kadangi skaičius visuomet skiriasi. Pavyzdziui, kai n = 2000, tai sukeitimų skaičius buvo nuo 720 iki 1328, skirtumas – 608, tai yra beveik pusė didziausio skaičiaus.

Įterpimo ir Šelo algoritmų palyginimų rezultatai labai skiriasi, kadangi teoriniai įverčiai skiriasi. Įterpimo yra O(n) – palyginimų, o Šelo yra O(n\*log n) – palyginimų.

Masyvo narių skaičius kyla nuo 50 -> 200, tai 50\*4, 200->500, tai 200\*2,5, 500->2000, tai 500\*4.

Pagal teorinį modelį įterpimo metodu O(n) – palyginimų skaičius kyla tolygiai.

71 \* 4 = 284 ~ 261, 261 \* 2,5 = 652 ~ 550, 550 \*4 = 2200 ~ 2870.

**pav. 5** Įterpimo algoritmo palyginimų skaičiaus kitimas didinant masyvo narių skaičių (geriausiu atveju)

**pav. 6** Šelo algoritmo palyginimų skaičiaus kitimas didinant maasyvo narių skaičių (geriausiu atveju)

Abiejų algoritmų sukeitimų rezultatai atitinka teorinius modelius: įterpimo O(1) , šelo O(n\* log2 n).Įterpimo algoritmo sukeitimų kiekis kito tiesiškai – didinant masyvo narių skaičių taip pat didėjo ir keitimų skaičius (49 → 199 → 499 → 1999). Šelo algoritmo rezultatai kito netolygiai pagal n\* log2 n formule didinant masyvo narių skaičių taip pat didėjo ir keitimų skaičius (23 → 30 → 89 → 858).

**pav. 7** Įterpimo algoritmo sukeitimų skaičiaus kitimas didinant masyvo narių skaičių (geriausiu atveju)

**pav. 8** Šelo algoritmo sukeitimų skaičiaus kitimas didinant masyvo narių skaičių (geriausiu atveju)

### **Blogiausias atvejis**

Iš 1 lentelės matyti, kad įterpimo algoritmo teoriniai įverčiai blogiausiu atveju yra O(n2) palyginimų ir O(n2) sukeitimų, o šelo algoritmo - O(n2) palyginimų ir O(n1,25) sukeitimų.

Blogiausio atvejo rezultatai, kai buvo atlikti bandymai su n = 50, n = 200, n = 500, n = 2000 pateikiami 4 lentelėje.

**lentelė 4** Blogiausio atvėjo tyrimo rezultatai

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Algoritmai** | **Bandymas, kai n = 50.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 1269 – lyg.  49 – keit. | 1274 – lyg.  49 – keit. | 1274 – lyg.  49 – keit. | 1274 – lyg.  49 – keit. | 1274 – lyg.  49 – keit. | 1273 – lyg.  49 – keit. |
| **Šelo** | 308 – lyg.  105 – keit. | 308 – lyg.  105 – keit. | 308 – lyg.  105 – keit. | 308 – lyg.  105 – keit.. | 308 – lyg.  105 – keit.. | 308 – lyg.  105 – keit. |
|  | **Bandymas, kai n = 200.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 20095 – lyg.  199 – keit. | 20097 – lyg.  199 – keit. | 20098 – lyg.  199 – keit. | 20095 – lyg.  199 – keit. | 20097 – lyg.  199 – keit. | 20096 – lyg.  199 – keit. |
| **Šelo** | 1822 – lyg.  619 – keit. | 1823 – lyg.  619 – keit.. | 1822 – lyg.  619 – keit. | 1822 – lyg.  619 – keit.. | 1822 – lyg.  620 – keit. | 1822 – lyg.  619 – keit.. |
|  | **Bandymas, kai n = 500.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 125232 – lyg.  499 – keit. | 125236 – lyg.  499 – keit. | 125236 – lyg.  499 – keit. | 125238 – lyg.  499 – keit. | 125240 – lyg.  499 – keit. | 125236 – lyg.  499 – keit. |
| **Šelo** | 5588 – lyg.  2082 – keit. | 5601 – lyg.  2095 – keit. | 5597 – lyg.  2091 – keit. | 5596 – lyg.  2090 – keit. | 5596 – lyg.  2090 – keit. | 5595 – lyg.  2089 – keit. |
|  | **Bandymas, kai n = 2000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 2000761 – lyg.  1999 – keit. | 2000806 – lyg.  1999 – keit. | 2000799 – lyg.  1999 – keit. | 2000814 – lyg.  1999 – keit. | 2000810 – lyg.  1999 – keit. | 2000798 – lyg.  1999 – keit. |
| **Šelo** | 28193 – lyg.  10187 – keit. | 28324 – lyg.  10318 – keit. | 28310 – lyg.  10304 – keit. | 28312 – lyg.  10306 – keit. | 28316 – lyg.  10310 – keit. | 28291 – lyg.  10285 – keit. |

**Rezultatų analizė:**

Įterpimo algoritmas yra stabilus, nes sukeitimu sukeitimu skaičius nesiskiria skirtinguose bandymuose, nors palyginimu skaičiai visuomet skiriasi, tačiau nedaug.Šį kart, Šelo algoritmas yra stabilus, kadangi vos per kelias reikšmes. Pavyzdziui, kai n = 500, tai sukeitimų skaičius buvo nuo 5588 iki 5601, skirtumas – 13, tai yra 0,2% didziausio skaičiaus.

Įterpimo ir Šelo algoritmų palyginimų rezultatai labai skiriasi, kadangi teoriniai įverčiai skiriasi. Įterpimo yra O(n2) – palyginimų, o Šelo yra O(n2) – palyginimų.

Masyvo narių skaičius kyla nuo 50 -> 200, tai 50\*4, 200->500, tai 200\*2,5, 500->2000, tai 500\*4.

Pagal teorinį modelį įterpimo metodu O(n2) – palyginimų skaičius kyla tolygiai.

1273 \* 42 = 20368 ~ 20096, 20096 \* 2,52 = 125600 ~ 125236, 125236 \*42 = 2003776 ~ 2000798.

**pav. 9** šelo algoritmo palyginimų skaičiaus kitimas didinant masyvo narių skaičių (geriausiu atveju)

Pagal teorinį modelį šelo metodu O(n2) – palyginimų skaičius nesutampa

308 \* 42 = 4928 ≠ 1822, 1822 \* 2,52 = 11387 ≠ 5595, 5595 \*42 = 89520 ≠ 28291.

**pav. 10** Šelo algoritmo palyginimų skaičiaus kitimas didinant maasyvo narių skaičių (geriausiu atveju)

Abiejų algoritmų sukeitimų rezultatai atitinka teorinius modelius: įterpimo O(n2) , šelo O(n1,25).Įterpimo algoritmo sukeitimų kiekis kito tiesiškai – tačiau didinant masyvo narių skaičių, keitimų skaičius nekilo kvadratiškai (49 → 199 → 499 → 1999). Šelo algoritmo rezultatai kito tolygiai pagal n1.25 formule didinant masyvo narių skaičių taip pat didėjo ir keitimų skaičius (105 → 619 → 2089 → 10285).

**pav. 11** Įterpimo algoritmo sukeitimų skaičiaus kitimas didinant masyvo narių skaičių (geriausiu atveju)

**pav. 12** Šelo algoritmo sukeitimų skaičiaus kitimas didinant masyvo narių skaičių (geriausiu atveju)

## **Sudėtingumo pagal laiką tyrimas:**

### **Vidutinis atvejis**

Vidutiniu atveju įterpimo algoritmo laiko sudėtingumas O(n2), šelo algoritmo laiko sudėtingumas O(n4/3).

Vidutinio atvejo tyrimo pagal laiką rezultatai yra pateikiami 5 lentelėje.

**lentelė 5** Vidutinio atvėjo tyrimo rezultatai

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Algoritmai** | **Bandymas, kai n = 15 000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 2.944 s | 2.953 s | 2.940 s | 2.974 s | 2.949 s | 2.952 s |
| **Šelo** | 0.025 s | 0.023 s | 0.024 s | 0.024 s | 0.023 s | 0.024 s |
|  | **Bandymas, kai n = 30 000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 11.815 s | 11.866 s | 11.780 s | 11.850 s | 11.870 s | 11.836 s |
| **Šelo** | 0.059 s | 0.053 s | 0.051 s | 0.053 s | 0.052 s | 0.054 s |
|  | **Bandymas, kai n = 60 000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 47.971 s | 47.524 s | 47.655 s | 47.711 s | 48.035 s | 47.779 s |
| **Šelo** | 0.115 s | 0.129 s | 0.114 s | 0.120 s | 0.116 s | 0.119 s |
|  | **Bandymas, kai n = 120 000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 192.257 s | 190.813 s | 192.291 s | 192.279 s | 189.671 s | 191.462 s |
| **Šelo** | 0.277 s | 0.263 s | 0.263 s | 0.270 s | 0.266 s | 0.268 s |

**Rezultatų analizė:**

Įterpimo algoritmo laiko sudėtingumas O(n2). Bandymo rezultatai tai patvirtina, narių skaičius kylant, surūšiavimo laikas didejo 4 kartus. Šelo algoritmo laiko sudėtingumas O(n4/3). Bandymo rezultatai tai patvirtina, surūšiavimo laikas didejo apie 2 kartus.

**pav. 13** Įterpimo algoritmo skaičiavimo laiko kitimas didinant masyvo narių skaičių

**pav. 14** Šelo algoritmo skaičiavimo laiko kitimas didinant masyvo narių skaičių

### **Geriausias atvejis**

Geriausiu atveju įterpimo algoritmo laiko sudėtingumas O(n), šelo algoritmo laiko sudėtingumas O(n\*log n).

Geriausio atvejo tyrimo pagal laiką rezultatai yra pateikiami 6 lentelėje.

**lentelė 6** Geriausio atvėjo tyrimo rezultatai

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Algoritmai** | **Bandymas, kai n = 15 000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 0.001 | 0.001 | 0.002 | 0.002 | 0.002 | 0.002 |
| **Šelo** | 0.010 | 0.011 | 0.010 | 0.009 | 0.010 | 0.010 |
|  | **Bandymas, kai n = 30 000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 0.002 | 0.002 | 0.002 | 0.002 | 0.002 | 0.002 |
| **Šelo** | 0.021 | 0.020 | 0.021 | 0.022 | 0.021 | 0.021 |
|  | **Bandymas, kai n = 60 000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 0.004 | 0.003 | 0.004 | 0.003 | 0.004 | 0.004 |
| **Šelo** | 0.045 | 0.048 | 0.045 | 0.046 | 0.045 | 0.046 |
|  | **Bandymas, kai n = 120 000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 0.007 | 0.006 | 0.007 | 0.006 | 0.006 | 0.006 |
| **Šelo** | 0.095 | 0.095 | 0.097 | 0.095 | 0.096 | 0.096 |

**Rezultatų analizė:**

Įterpimo algoritmo laiko sudėtingumas O(n). Bandymo rezultatai tai patvirtina, narių skaičius kylant, surūšiavimo laikas didejo 2 kartus. Šelo algoritmo laiko sudėtingumas O(n\*log n).Bandymo rezultatai tai patvirtina, surūšiavimo laikas didejo apie 2 kartus.

**pav. 15** Įterpimo ir šelo algoritmų skaičiavimo laiko kitimas didinant masyvo narių skaičių

### **Blogiausias atvejis**

Blogiausiu atveju įterpimo algoritmo laiko sudėtingumas O(n2), šelo algoritmo laiko sudėtingumas O(n2).

Blogiausio atvejo tyrimo pagal laiką rezultatai yra pateikiami 7 lentelėje.

**lentelė 7** Blogiausio atvėjo tyrimo rezultatai

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Algoritmai** | **Bandymas, kai n = 15 000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 5.907 s | 5.958 s | 5.908 s | 5.896 s | 5.906 s | 5.915 s |
| **Šelo** | 0.014 s | 0.017 s | 0.015 s | 0.014 s | 0.014 s | 0.015 s |
|  | **Bandymas, kai n = 30 000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 23.677 s | 23.658 s | 23.680 s | 24.070 s | 23.854 s | 23.788 s |
| **Šelo** | 0.030 s | 0.030 s | 0.030 s | 0.030 s | 0.031 s | 0.030 s |
|  | **Bandymas, kai n = 60 000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 95.758 s | 95.449 s | 95.334 s | 95.440 s | 95.353 s | 95.467 s |
| **Šelo** | 0.064 s | 0.065 s | 0.067 s | 0.067 s | 0.065 s | 0.066 s |
|  | **Bandymas, kai n = 120 000.** | | | | | |
| Nr.1 | Nr.2 | Nr.3 | Nr.4 | Nr.5 | Vidurkis |
| **Įterpimo** | 383.455 s | 379.804 s | 380.305 s | 379.678 | 379.678 | 380.584 |
| **Šelo** | 0.136 | 0.141 | 0.139 | 0.148 | 0.140 | 0.141 |

**Rezultatų analizė:**

Abiejų algoritmų sudėtingumas pagal laiką yra kvadratinis O(n2). Bandymo rezultatai tai patvirtina su įterpimo algoritmu, surūšiavimo laikas didėjo 4 kartus. Tačiau šelo algorimo rezultatai neatitiko teorijos, surūšiavimo laikas didėjo ne 4 o 2 kartus.

**pav. 16** Įterpimo algoritmo skaičiavimo laiko kitimas didinant masyvo narių skaičių

**pav. 17** Šelo algoritmo skaičiavimo laiko kitimas didinant masyvo narių skaičių

# **Išvados:**

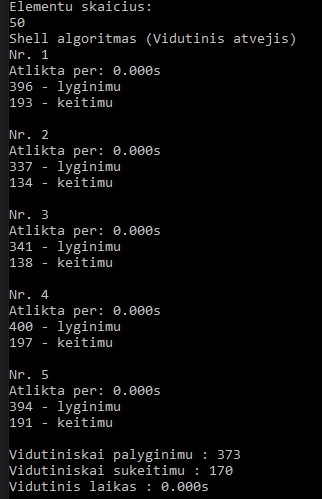
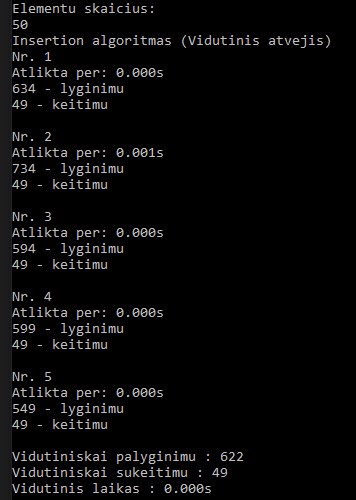
* Šelo algoritas nėra stabilus todėl ne visada teoriniai skaičiavimai atitiko bandymų rezultatus.
* Įterpimo algoritmo metu gauti rezultatai atitiko teorinius iverčius
* Šelo algoritmo rūšiavimas yra daug greitesnis nei įterpimo, išskyrus geriausiu atvėju įterpimo algoritmas rūšiavimą atlieka daug greičiau.
* Įterpimo algoritmas efektyvus tik mažuose intervaluose arba geriausio atvėjo rūšiavimuose

# **Bandymų paveiksliukai:**

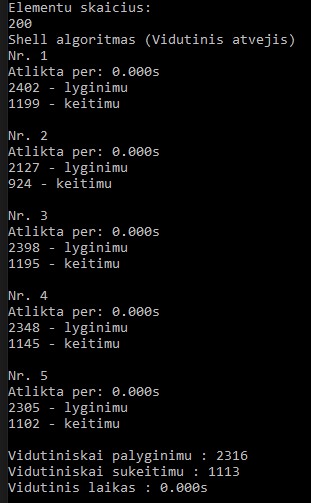
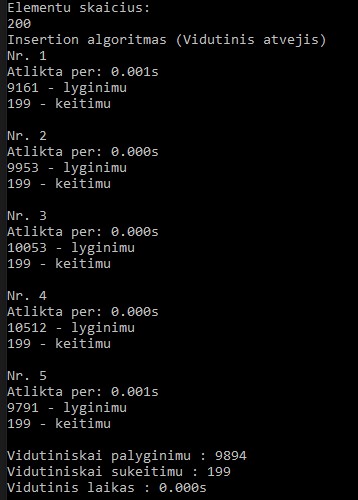
## **Sudėtingumo pagal sukeitimus tyrimas:**

### **Vidutinis atvėjis:**

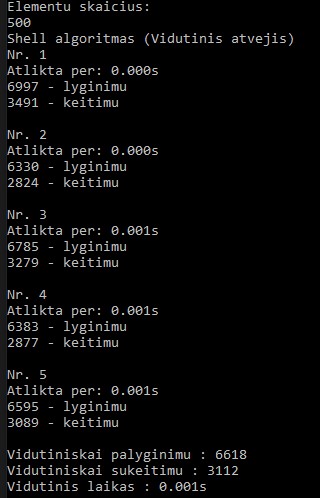
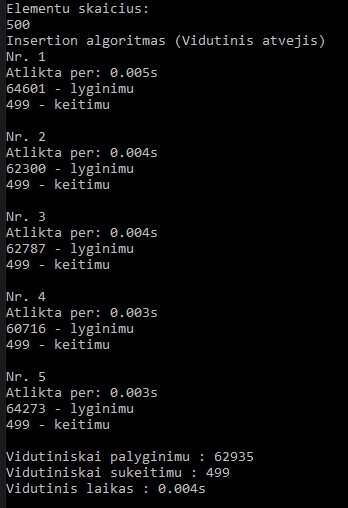
n = 50



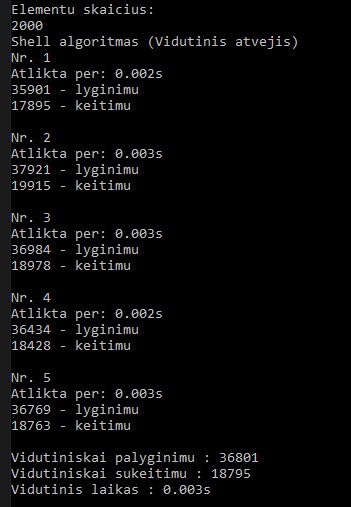
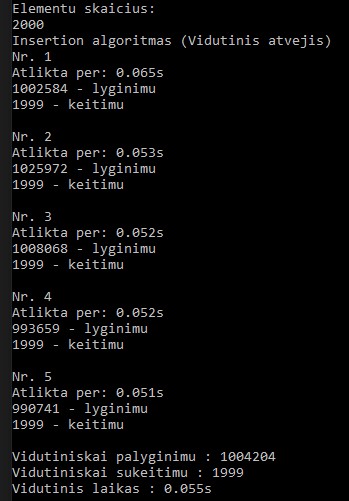
n = 200



n = 500

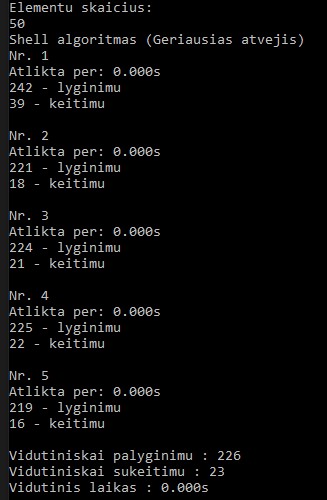
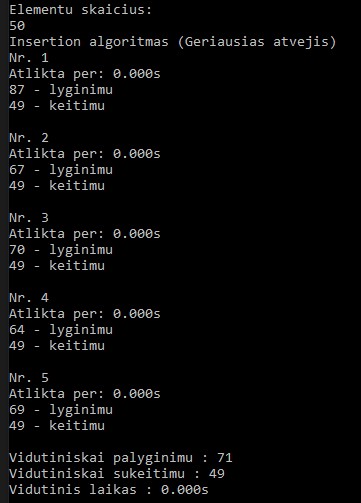


n = 2000

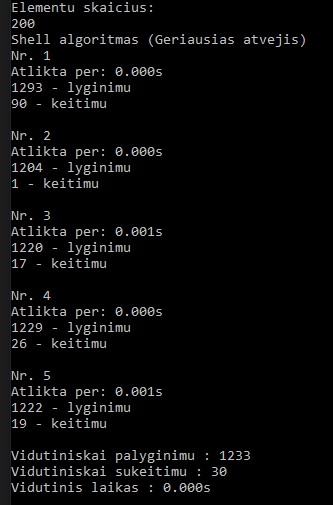
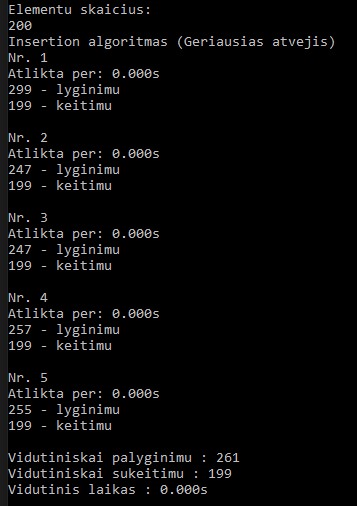


### **Geriausias atvėjis:**

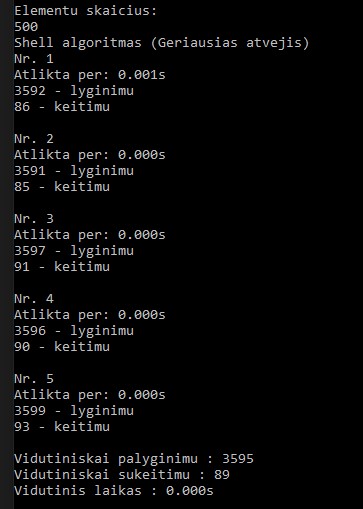
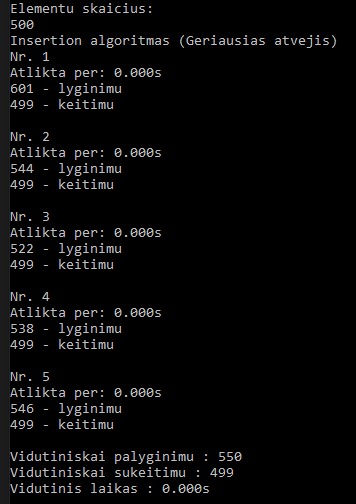
n = 50



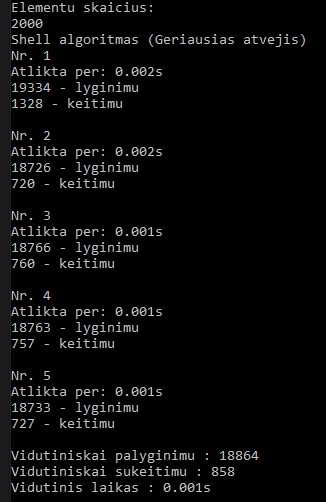
n = 200



n = 500

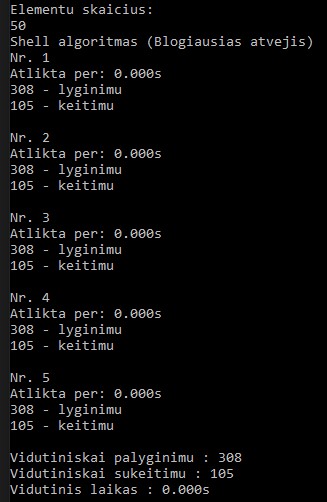
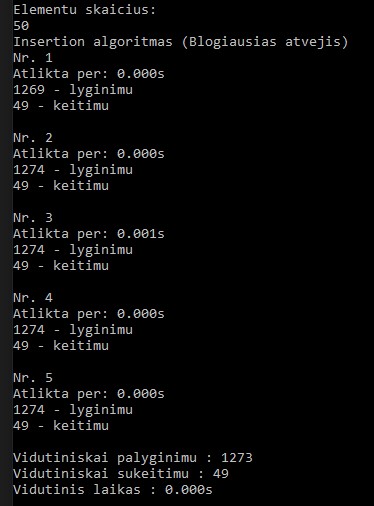


n = 2000

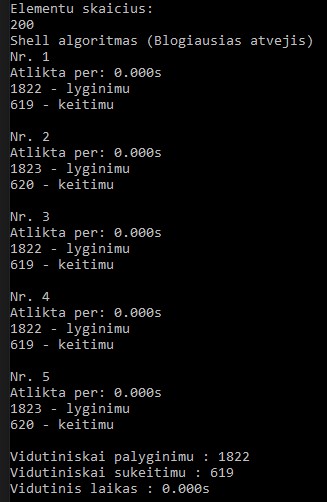
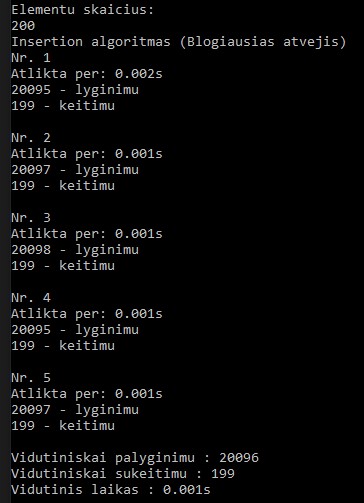


### **Blogiausias atvejis:**

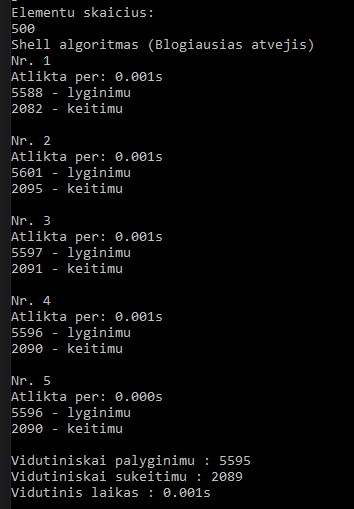
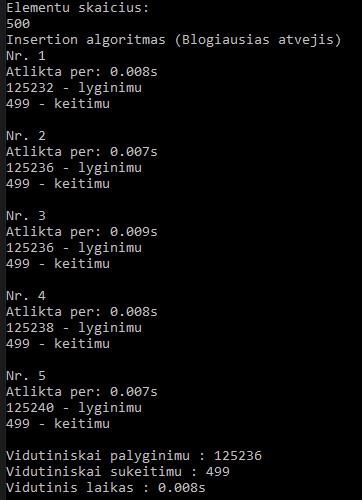
n = 50



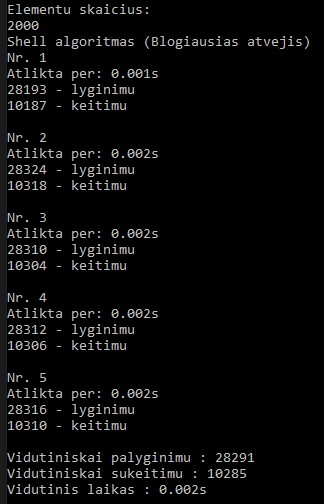
n = 200



n = 500



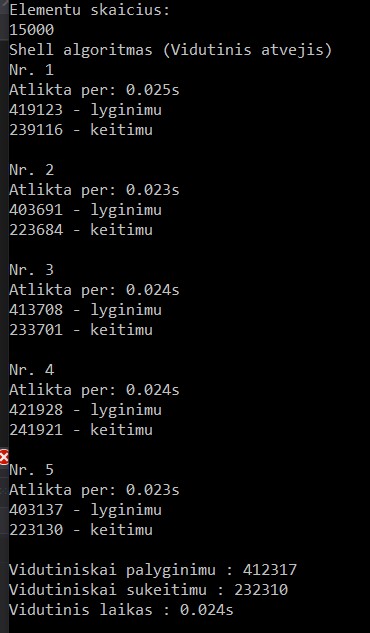
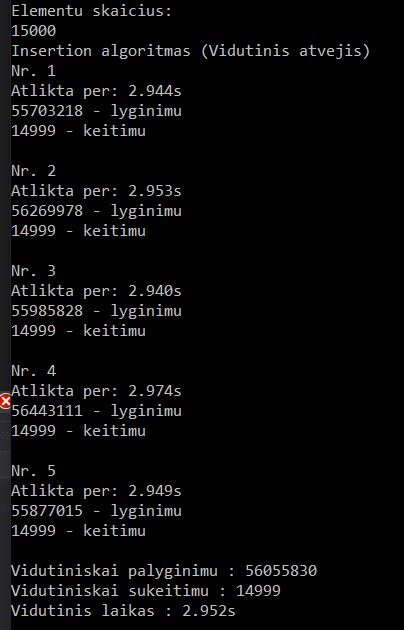
n = 2000



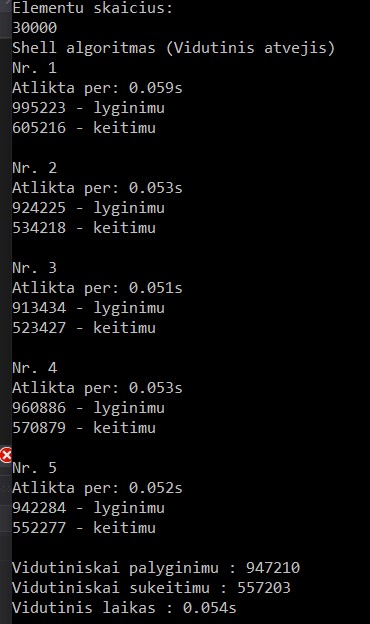
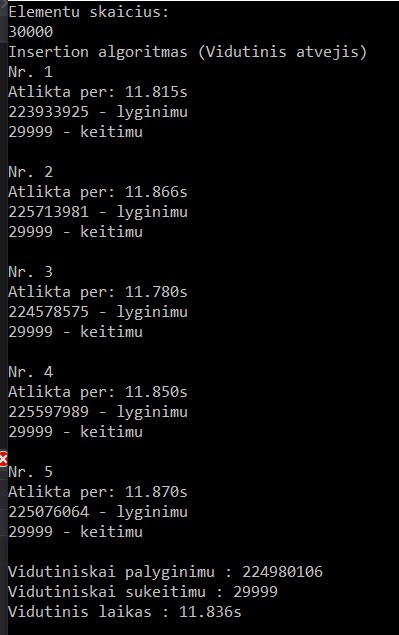
## **Sudėtingumo pagal laiką tyrimas:**

### **Vidutinis atvejis:**

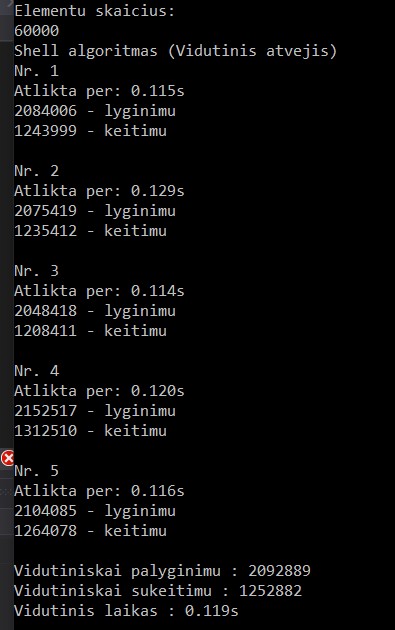
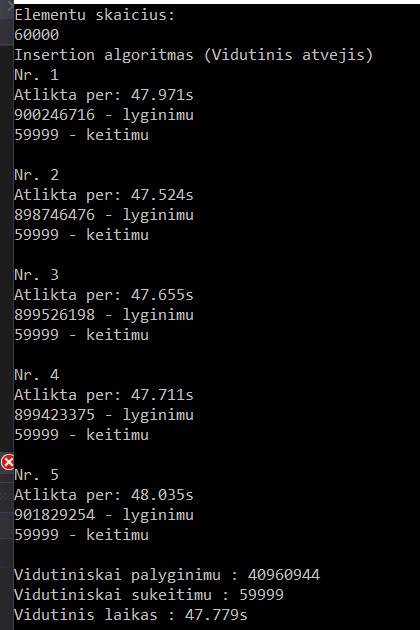
n = 15 000



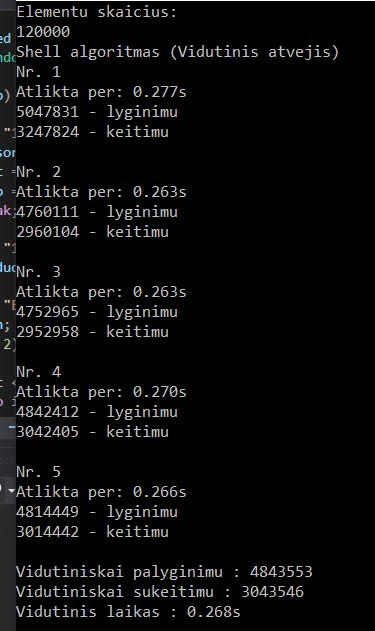
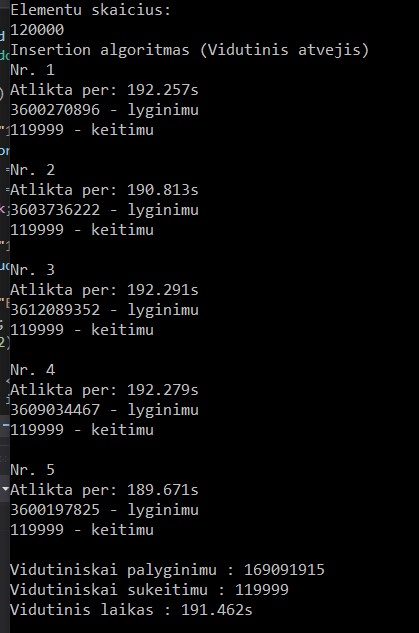
n = 30 000



n = 60 000

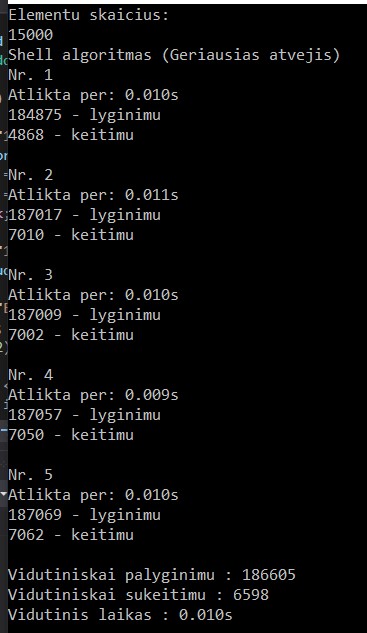
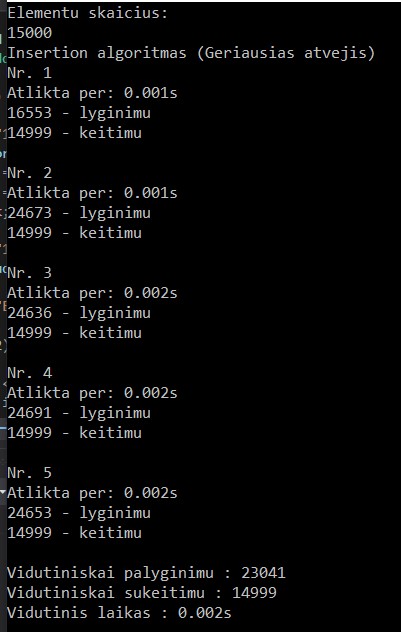


n = 120 000

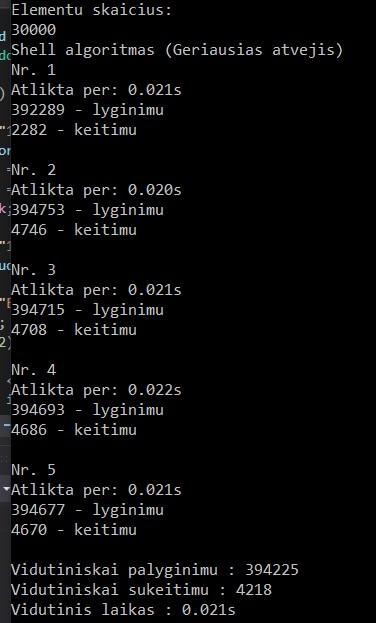
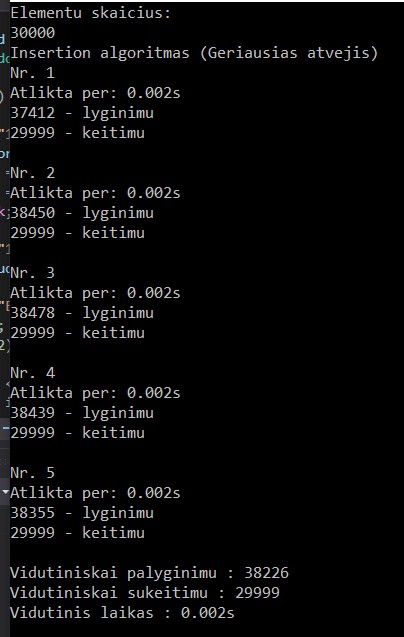


### **Geriausias atvejis:**

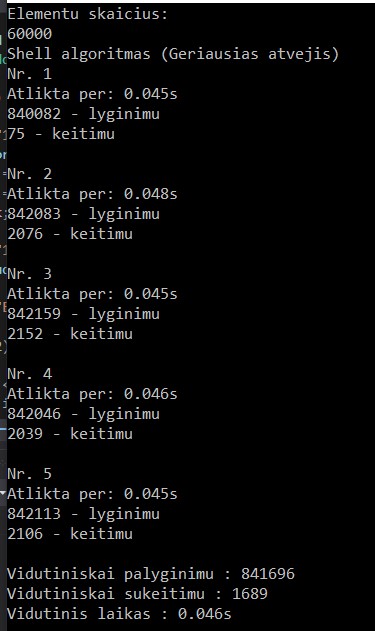
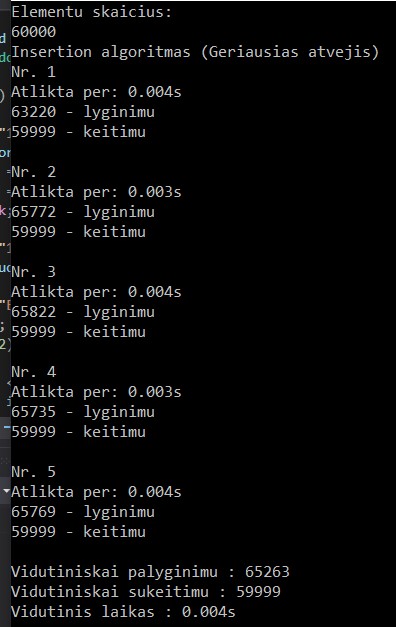
n = 15 000



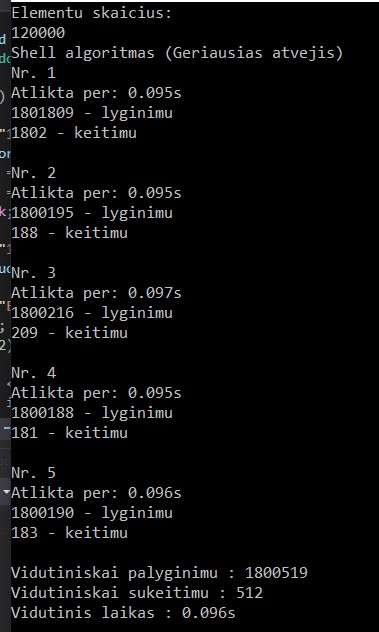
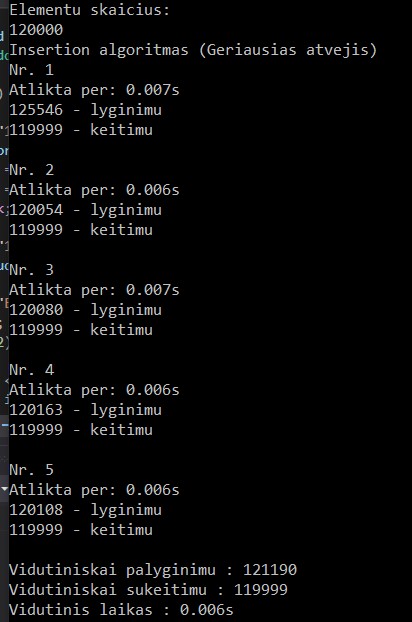
n = 30 000



n = 60 000

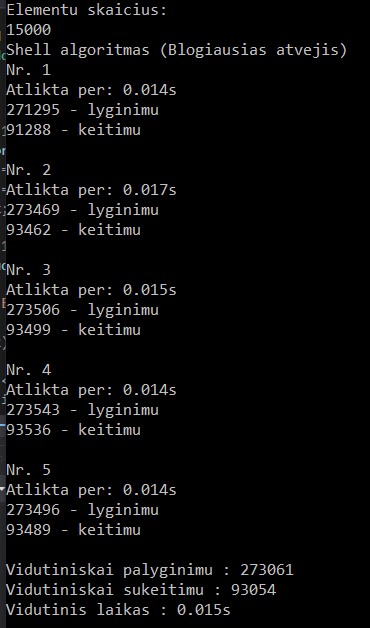
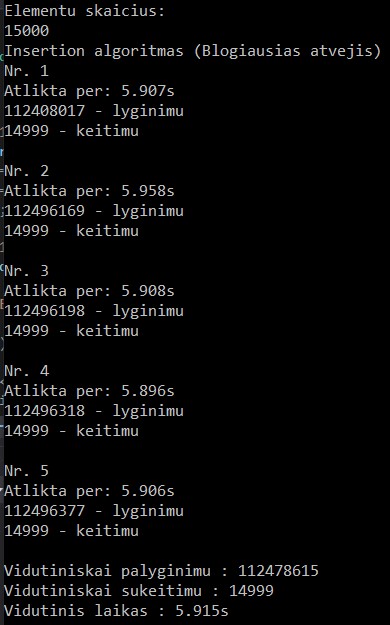


n = 120 000

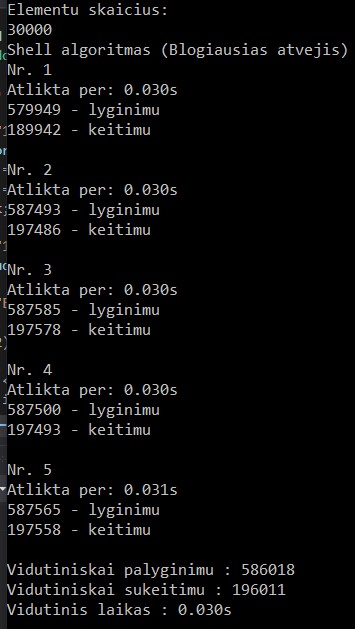
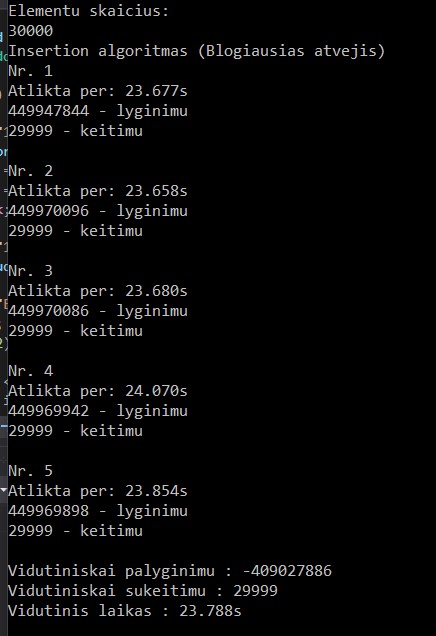


### **Blogiausias atvejis:**

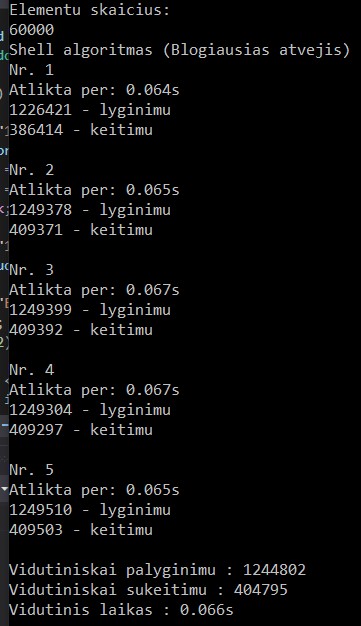
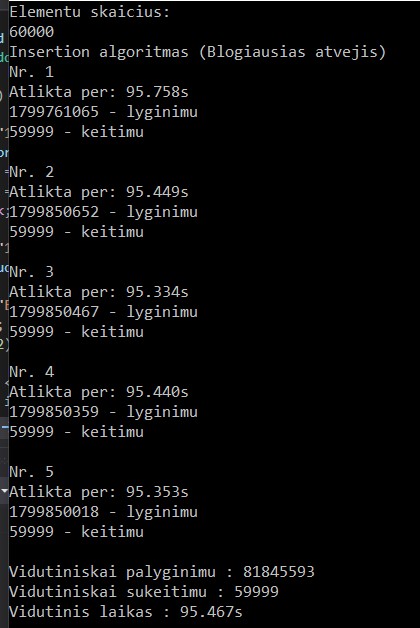
n = 15 000



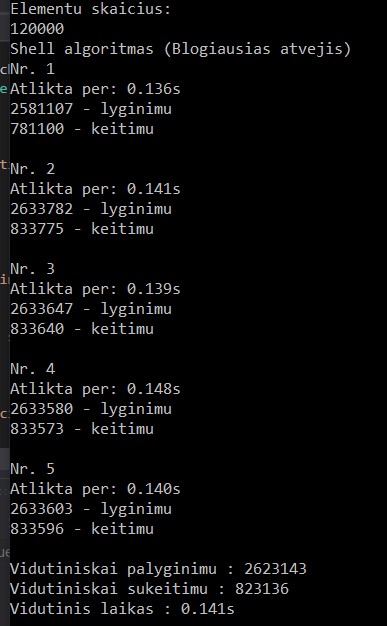
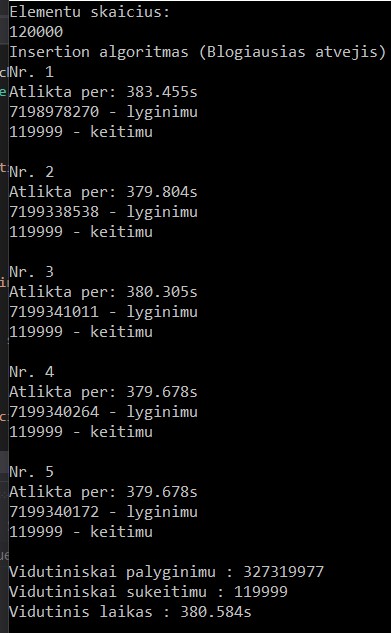
n = 30 000



n = 60 000



n = 120 000



# **Programos kodas:**

#include <iostream>

#include <ctime>

#include <time.h>

#include <chrono>

#include <random>

#include <algorithm>

#include <vector>

#include <numeric>

using namespace std;

void printArray(vector<int> array, int size) {

for (int i = 0; i < size; i++) {

cout << array[i] << " ";

}

cout << endl;

}

void insertionSort(vector<int> &array, int size, unsigned long long int &palyg, int &keit) {

for (int i = 1; i < size; i++) {

int key = array[i];

int j = i - 1;

while (j >= 0 && key < array[j] ) {

palyg++;

array[j + 1] = array[j];

--j;

}

array[j + 1] = key;

keit++;

palyg++;

}

}

void shellSort(vector<int> &array, int n, unsigned long long int &palyg, int &keit) {

for (int interval = n / 2; interval > 0; interval /= 2) {

for (int i = interval; i < n; i++) {

int temp = array[i];

int j;

palyg++;

for (j = i; j >= interval && array[j - interval] > temp; j -= interval) {

array[j] = array[j - interval];

keit++;

palyg++;

}

array[j] = temp;

}

}

}

int main() {

int sort, n;

int duom;

int loop = 1;

bool stop = false;

double t;

vector<int> data2;

vector<unsigned long long int> VidP, VidK;

vector<double> VidT;

unsigned long long int palyg = 0;

int keit = 0;

unsigned int seed = std::chrono::steady\_clock::now().time\_since\_epoch().count();

std::default\_random\_engine engine(seed);

while (!stop)

{

cout << "1- Insertion Sort, 2- Shell Sort, 3- Stop" << endl;

cin >> sort;

if (sort == 3) {

stop = true;

break;

}

cout << "1- Vidutinis atvejis, 2- Geriausias atvejis, 3- Blogiausias atvejis\n";

cin >> duom;

ived :

cout << "Elementu skaicius: " << endl;

cin >> n;

if (n < 2)

{

cout << "Skaicius per mazas!\n";

goto ived;

}

srand(time(NULL));

int r = rand() % n-1;

srand(time(NULL));

int numb = rand() % 9999;

ciklas:

switch (duom)

{

case 1: // vidutinis

for (int i = 0; i < n; i++)

{

std::uniform\_int\_distribution<int> distr(1, 9999);

int a = distr(engine);

data2.push\_back(a);

}

break;

case 2: // geriausias

for (int i = 0; i < n; i++)

{

std::uniform\_int\_distribution<int> distr(1, 9999);

int a = distr(engine);

data2.push\_back(a);

}

std::sort(data2.begin(), data2.end());

data2[r] = numb;

break;

case 3: // blogiausias

for (int i = 0; i < n; i++)

{

std::uniform\_int\_distribution<int> distr(1, 9999);

int a = distr(engine);

data2.push\_back(a);

}

std::sort(data2.begin(), data2.end(), std::greater<int>());

break;

}

if (sort == 1) // Insertion sort

{

if (loop == 1) {

cout << "Insertion algoritmas ";

if (duom == 1)

cout << "(Vidutinis atvejis)\n";

else if (duom == 2)

cout << "(Geriausias atvejis)\n";

else if (duom == 3)

cout << "(Blogiausias atvejis)\n";

}

while (loop < 6)

{

clock\_t tStart = clock();

insertionSort(data2, n, palyg, keit);

t = (double)(clock() - tStart) / CLOCKS\_PER\_SEC;

cout << "Nr. " << loop << endl;

printf("Atlikta per: %.3fs\n", t);

cout << palyg << " - lyginimu\n";

cout << keit << " - keitimu\n";

VidP.push\_back(palyg);

VidK.push\_back(keit);

VidT.push\_back(t);

palyg = 0;

keit = 0;

data2.clear();

cout << endl;

loop++;

goto ciklas;

}

cout << "Vidutiniskai palyginimu : " << accumulate(VidP.begin(), VidP.end(), 0) / (loop - 1)<< endl;

cout << "Vidutiniskai sukeitimu : " << accumulate(VidK.begin(), VidK.end(), 0) / (loop - 1) << endl;

printf("Vidutinis laikas : %.3fs\n" , accumulate(VidT.begin(), VidT.end(), 0.000) / (loop - 1));

cout << endl;

loop = 1;

VidP.clear();

VidK.clear();

VidT.clear();

}

else if (sort == 2) // Shell sort

{

if (loop == 1) {

cout << "Shell algoritmas ";

if (duom == 1)

cout << "(Vidutinis atvejis)\n";

else if (duom == 2)

cout << "(Geriausias atvejis)\n";

else if (duom == 3)

cout << "(Blogiausias atvejis)\n";

}

while (loop < 6)

{

clock\_t tStart = clock();

shellSort(data2, n, palyg, keit);

t = (double)(clock() - tStart) / CLOCKS\_PER\_SEC;

cout << "Nr. " << loop << endl;

printf("Atlikta per: %.3fs\n", t);

cout << palyg << " - lyginimu\n";

cout << keit << " - keitimu\n";

VidP.push\_back(palyg);

VidK.push\_back(keit);

VidT.push\_back(t);

palyg = 0;

keit = 0;

data2.clear();

cout << endl;

loop++;

goto ciklas;

}

cout << "Vidutiniskai palyginimu : " << accumulate(VidP.begin(), VidP.end(), 0) / (loop - 1) << endl;

cout << "Vidutiniskai sukeitimu : " << accumulate(VidK.begin(), VidK.end(), 0) / (loop - 1) << endl;

printf("Vidutinis laikas : %.3fs\n", accumulate(VidT.begin(), VidT.end(), 0.000) / (loop - 1));

cout << endl;

loop = 1;

VidP.clear();

VidK.clear();

VidT.clear();

}

}

}