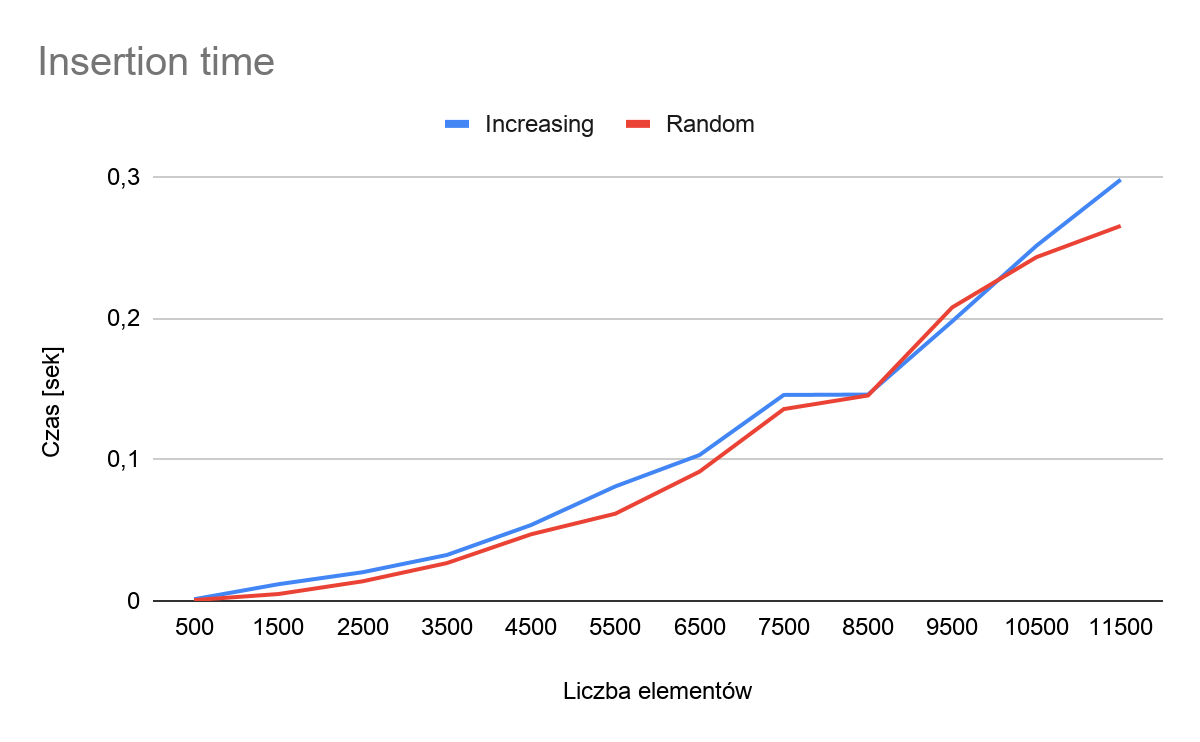
Katarzyna Zawrotniak

Kamil Albrycht

**Sprawozdanie**

**Lista jednokierunkowa**  
  
**Insertion time**

|  |  |  |
| --- | --- | --- |
| **Liczba elementów** | **Increasing** | **Random** |
| **500** | 0,00122 | 0,000633 |
| **1500** | 0,011833 | 0,0049 |
| **2500** | 0,020315 | 0,013888 |
| **3500** | 0,032497 | 0,026771 |
| **4500** | 0,05368 | 0,047136 |
| **5500** | 0,08099 | 0,061687 |
| **6500** | 0,103186 | 0,091454 |
| **7500** | 0,145655 | 0,135652 |
| **8500** | 0,145913 | 0,145268 |
| **9500** | 0,197737 | 0,207575 |
| **10500** | 0,251156 | 0,243078 |
| **11500** | 0,297873 | 0,26517 |

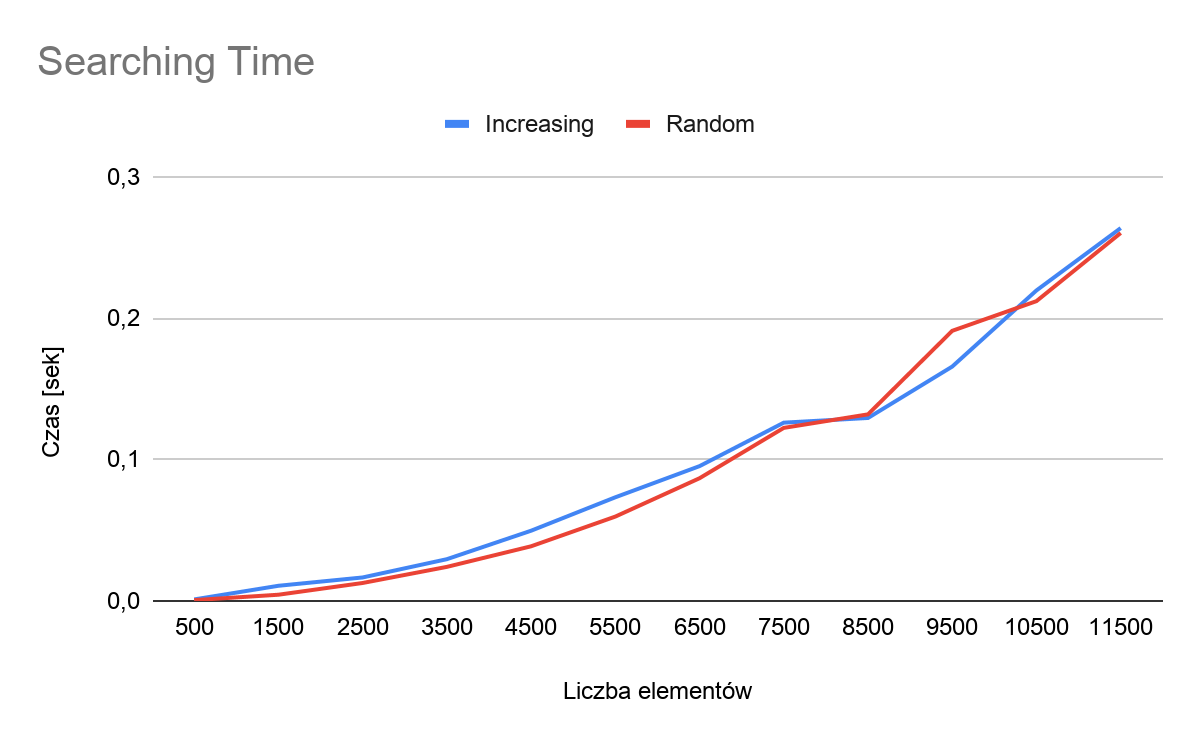


Wnioski:

W przedziale pomiędzy 500 a 4500 elementów dodawanie zachodzi szybciej w przypadku liczb losowych jednak nie jest to bardzo duża różnica. Zwiększa się ona po przekroczeniu 4500 aż do 6500 elementów, można wtedy zaobserwować skrócenie czasu dodawania elementów losowych. W przypadku 7500-8500 elementów czas operacji jest praktycznie taki sam w przypadku liczb rosnących, natomiast w przypadku liczb losowych nieznacznie zwalnia. Następnie możemy zaobserwować dość gwałtowny wzrost czasu, a przy liczbie 9500 elementów krótszy jest czas liczb rosnących w porównaniu do losowych. Przy 11500 elementach ponownie liczby losowe dodawane są w krótszym czasie.

**Searching time**

|  |  |  |
| --- | --- | --- |
| **ilość elementów** | **Increasing** | **Random** |
| **500** | 0,001085 | 0,000594 |
| **1500** | 0,010730 | 0,004409 |
| **2500** | 0,016616 | 0,012706 |
| **3500** | 0,029531 | 0,024096 |
| **4500** | 0,049658 | 0,038652 |
| **5500** | 0,073295 | 0,059649 |
| **6500** | 0,095328 | 0,086775 |
| **7500** | 0,126061 | 0,122365 |
| **8500** | 0,129374 | 0,131929 |
| **9500** | 0,165720 | 0,191001 |
| **10500** | 0,219577 | 0,211976 |
| **11500** | 0,263623 | 0,260163 |

****

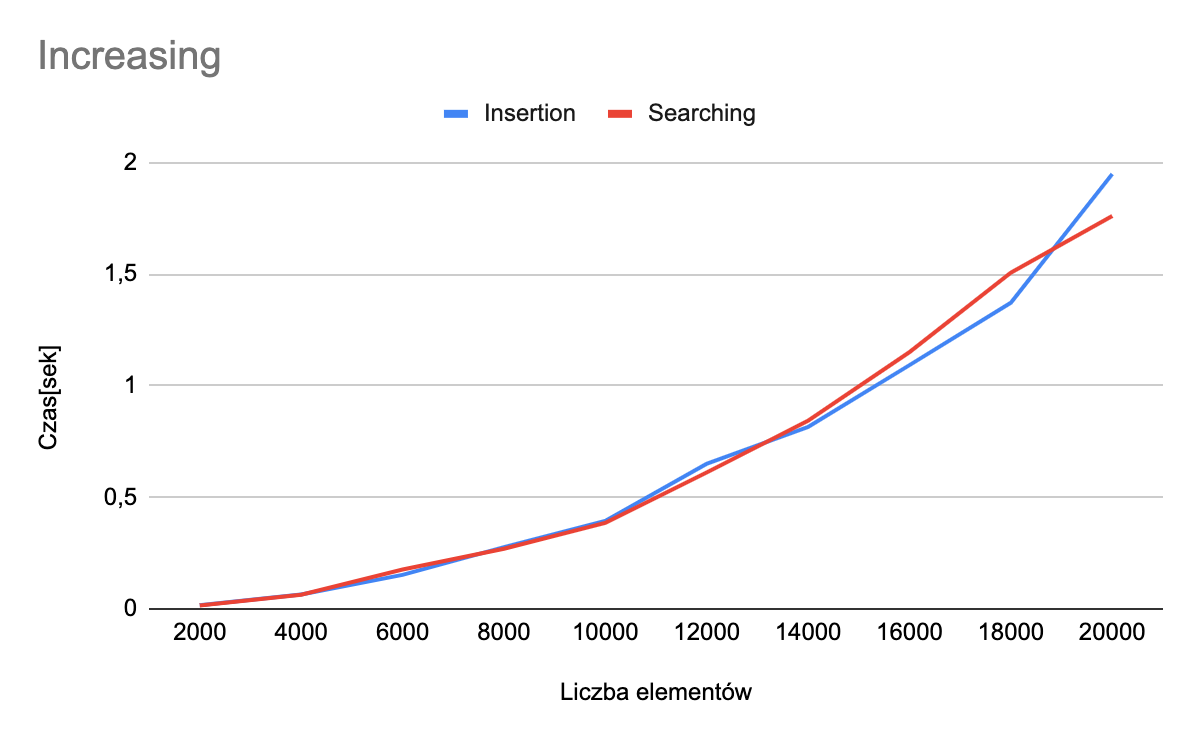
Wnioski:

W przedziale pomiędzy 500 a 3500 elementów szukanie zachodzi szybciej w przypadku liczb losowych jednak nie jest to bardzo duża różnica. Zwiększa się ona po przekroczeniu 3500 aż do 7500 elementów, można wtedy zaobserwować skrócenie czasu szukania elementów losowych. W przypadku 7500-8500 elementów czas operacji jest praktycznie taki sam dla obu rodzaju liczb. Następnie możemy zaobserwować dość gwałtowny wzrost czasu, a jednocześnie krótszy jest czas liczb rosnących w porównaniu do losowych. Pomiędzy 9500 a 10500 elementów ponownie liczby losowe są wyszukiwane szybciej i tak pozostaje cały czas włącznie z 11500 elementów, lecz różnica czasu pomiędzy nimi jest nieznaczna.

**Drzewo**

**Rosnące**

|  |  |  |
| --- | --- | --- |
| Liczba elementów | Insertion | Searching |
| 2000 | 0,016156 | 0,015417 |
| 4000 | 0,064366 | 0,063496 |
| 6000 | 0,152281 | 0,176394 |
| 8000 | 0,276715 | 0,26952 |
| 10000 | 0,393866 | 0,385743 |
| 12000 | 0,650621 | 0,61097 |
| 14000 | 0,814873 | 0,84318 |
| 16000 | 1,092342 | 1,150892 |
| 18000 | 1,372415 | 1,506288 |
| 20000 | 1,948533 | 1,760739 |

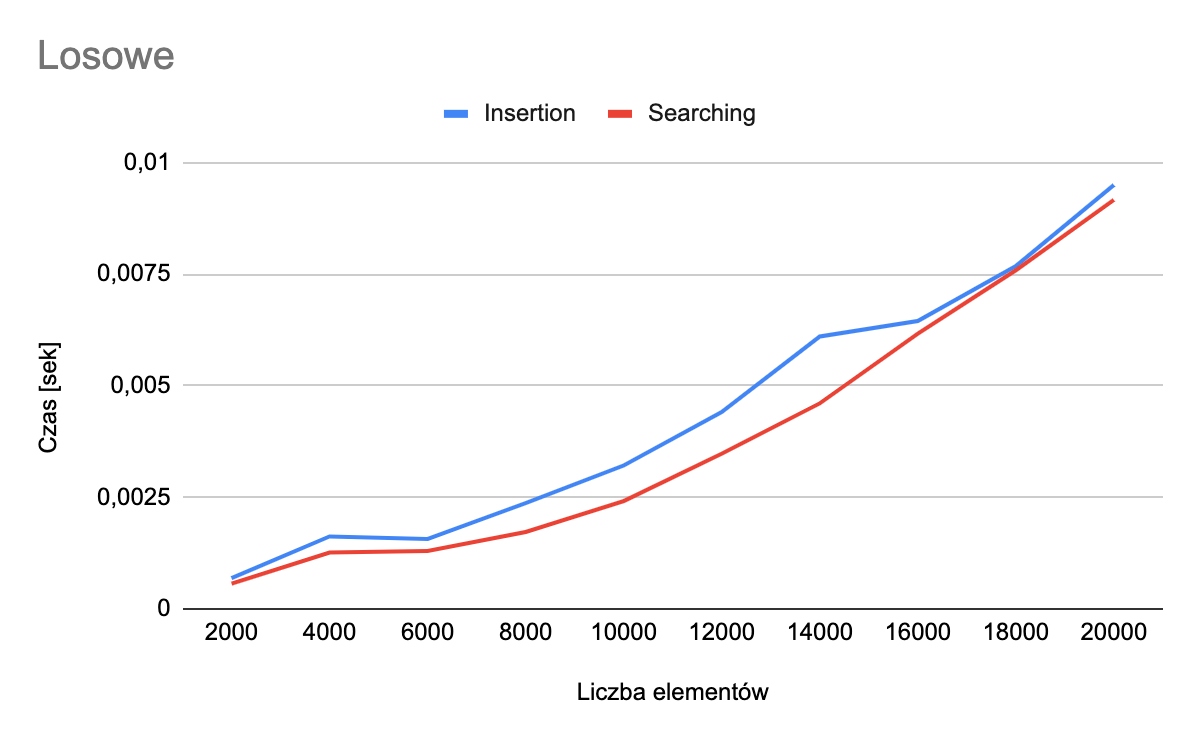
****

Wnioski:

W przypadku danych w porządku rosnącym zarówno dodawanie jak i szukanie przebiega w podobnym czasie. Największą rozbieżność możemy zaobserwować w przedziale od 16000 do nieco ponad 18000 liczb. Szybciej wtedy następuje dodawanie liczb niż ich odszukiwanie. Nieco przed 20000 liczb i wyżej liczby te są w krótszym czasie wyszukiwane niż dodawane.

**Losowe**

|  |  |  |
| --- | --- | --- |
| Liczba elementów | Insertion | Searching |
| 2000 | 0,000689 | 0,000568 |
| 4000 | 0,001623 | 0,001263 |
| 6000 | 0,001567 | 0,001296 |
| 8000 | 0,002372 | 0,001724 |
| 10000 | 0,003212 | 0,002416 |
| 12000 | 0,00441 | 0,003477 |
| 14000 | 0,006105 | 0,004604 |
| 16000 | 0,006449 | 0,006166 |
| 18000 | 0,007679 | 0,007587 |
| 20000 | 0,009498 | 0,009162 |

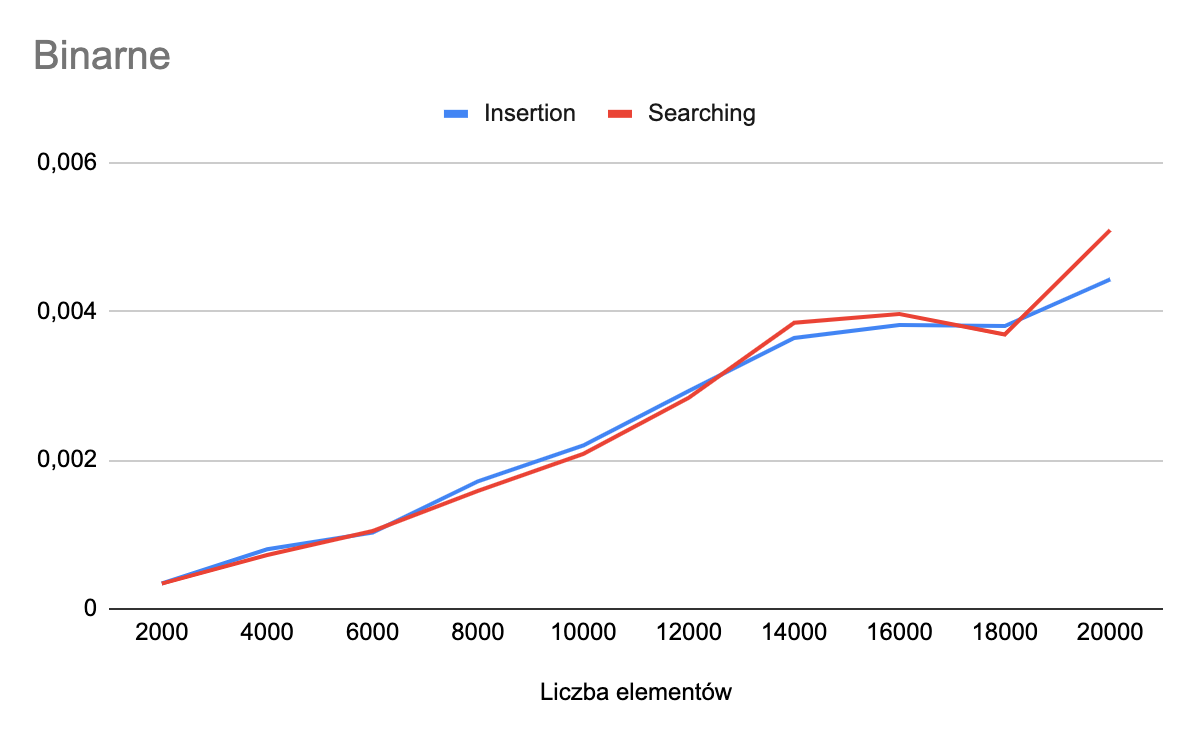
****

Wnioski:

W przypadku danych losowych znacznie szybciej przebiega operacja wyszukiwania liczb niż ich dodawania. Szczególną rozbieżność można zaobserwować w przedziale od 6000 do 16000 liczb. Następnie różnica ta delikatne się zmniejsza lecz dalej liczby czas wyszukiwania jest krótszy niż czas dodawania.

**Binarne**

|  |  |  |
| --- | --- | --- |
| Liczba elementów | Insertion | Searching |
| 2000 | 0,000343 | 0,000339 |
| 4000 | 0,000802 | 0,000724 |
| 6000 | 0,001025 | 0,001047 |
| 8000 | 0,001715 | 0,001587 |
| 10000 | 0,002197 | 0,002085 |
| 12000 | 0,002928 | 0,002838 |
| 14000 | 0,003643 | 0,003846 |
| 16000 | 0,003816 | 0,003963 |
| 18000 | 0,003802 | 0,003689 |
| 20000 | 0,00443 | 0,005091 |

****

Wnioski:

W przypadku drzewa binarnego w przedziale od 2000 do 14000 dodawanie i wyszukiwanie przebiega w bardzo podobnym czasie. Nieznacznie szybciej liczby są wyszukiwane, lecz jest to prawie niezauważalna różnica. Po przekroczeniu 16000 liczb następuje skrócenie czasu w przypadku wyszukiwania, natomiast w przypadku dodawania czas ten utrzymuje się na bardzo podobnym poziomie jak w przypadku 14000 liczb. Po przekroczeniu 18000 liczb następuje dość gwałtowne spowolnienie czasu zarówno dodawania jak i wyszukiwania.