
Struktury danych i złożoność obliczeniowa - projekt

SPRAWOZDANIE – Zadanie projektowe nr 1

MARIA KRANZ 263985

Prowadzący: mgr. Inż. Antonii Sterna

# Badanie efektywności operacji dodawania, usuwania oraz wyszukiwania elementów w różnych strukturach danych sprawozdanie

Data oddania projektu: 18.04.2023 r.

# Spis treści

# Wstęp

Zadanie projektowe nr 1 polegało na zbadaniu czasu działania operacji na różnych strukturach danych. Zaimplementowane zostały operacje takie jak **dodawanie**, **usuwanie** czy **wyszukiwanie elementu** na strukturach takich, jak **tablica**, **lista**, **kopiec** i **drzewo RBT**.

Zgodnie z założeniami elementami każdej struktury były 4 bajtowe liczby ze znakiem (integer). A struktury były alokowane dynamicznie i relokowane przy każdym dodaniu/usunięciu elementu. Kopiec został zaimplementowany jako tablica dynamiczna.

## Wstęp teoretyczny

### Tablica dynamiczna

Przy usuwaniu oraz dodawaniu elementu należy przekopiować wszystkie wartości do nowej tablicy.

Przy wyszukiwaniu natomiast, trzeba iterować po kolei tablice, aż do napotkania elementu, bądź jej końca. Stąd też złożoności prezentują się tak:

| Operacja | | Złożoność |
|---|---|---|
| **Dodanie elementu** | **na początek** | O(n) |
| | **na koniec** | O(n) |
| | **w dowolne miejsce** | O(n) |
| **Usunięcie elementu** | **z początku** | O(n) |
| | **z końca** | O(n) |
| | **z dowolnego miejsca** | O(n) |
| **Wyszukanie elementu** | | O(n) |

### Lista dwukierunkowa

Dzięki liście wiązanej, możemy dopisywać kolejne elementy za pomocą wskaźników. Te elementy nie muszą być w jednym miejscu pamięci jak w przypadku tablicy. Nie trzeba zatem przepisywać w nowe miejsce elementów przy zmianie rozmiaru listy.

| Operacja | Złożoność | |
|---|---|---|
| **Dodanie elementu** | **na początek** | O(1) |
| | **na koniec** | O(1) |
| | **w dowolne miejsce** | O(n) |
| **Usunięcie elementu** | **z początku** | O(1) |
| | **z końca** | O(1) |
| | **z dowolnego miejsca** | O(n) |
| **Wyszukanie elementu** | O(n) | |

### Kopiec binarny

W wariancie implementacji kopca jako tablicy dynamicznej złożoność zmiany rozmiaru kopca uwzględnia operacje kopca oraz relokowanie tablicy. Wyszukiwanie zostało zaimplementowane, tak jak wyszukiwanie po tablicy.

| Operacja | Średnia złożoność |
|---|---|
| **Dodanie elementu** | O(nlgn) |
| **Usuniecie elementu** | O(nlgn) |
| **Wyszukanie elementu** | O(n) |

### Drzewo czerwono – czarne

Drzewo, tak jak lista jest strukturą wskaźnikową – nie ma potrzeby relokowania wszystkich elementów przy zmianie rozmiaru struktury.

| Operacja | Złożoność |
|---|---|
| **Dodanie elementu** | O(lgn) |
| **Usuniecie elementu** | O(lgn) |
| **Wyszukanie elementu** | O(lgn) |

## Plan eksperymentu

Wygenerowane zostaną losowe wartości dla rozmiarów

- 10.000,
- 100.000,
- 1.000.000,
- 2.000.000,
- 4.000.000.

Czas zostanie pomierzony dla operacji na 10 różnych populacjach w każdym rozmiarze.

Na bazie wyników z tych populacji wyliczony zostanie średni czas wykonania operacji dla każdego rozmiaru. Wartości te zostaną umieszczone na wykresie.

Wartości będą losowymi elementami należącymi do przedziału <-2147483648, 2147483647>.

```
int FileData::generateRandomValue() {
    random_device rd; // non-deterministic generator
    mt19937 gen( sd: rd()); // random engine seeded with rd()
    uniform_int_distribution<> dist( a: INT32_MIN,  b: INT32_MAX);

    return dist( &: gen);
}
```

*Ilustracja 1. Funkcja generująca losową wartość Integer.*

Czas będzie odmierzany przy użyciu funkcji *QueryPerformanceFrequency*().

```
long long int read_QPC()
{
    LARGE_INTEGER count;
    QueryPerformanceCounter( lpPerformanceCount: &count);
    return((long long int)count.QuadPart);
}

Tests::Tests(FileData* data) {
    this->data = data;
    QueryPerformanceFrequency( lpFrequency: (LARGE_INTEGER *)&frequency);
}
```

*Ilustracja 2. Funkcja odczytu czasu.*

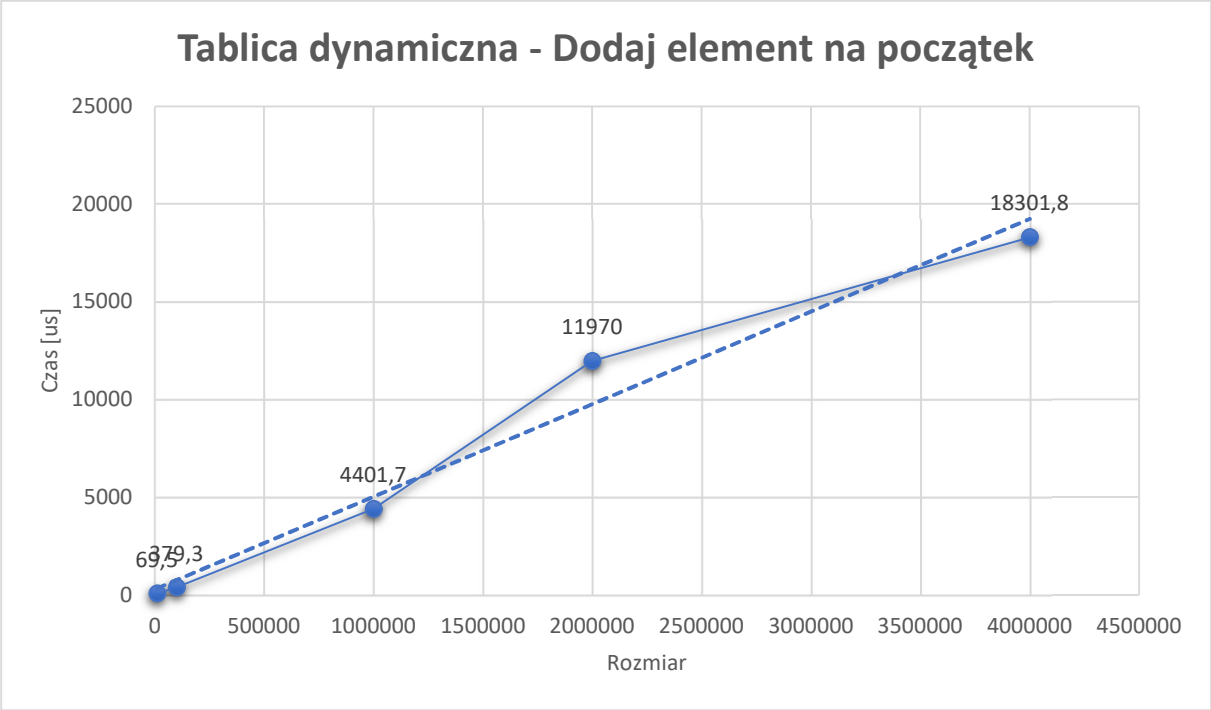
```
start = read_QPC();
tab->insertHead(val);
elapsed = read_QPC() - start;
```

*Ilustracja 3. Przykład implementacji pomiaru czasu w kodzie.*

# Wyniki badań

## Tablica dynamiczna

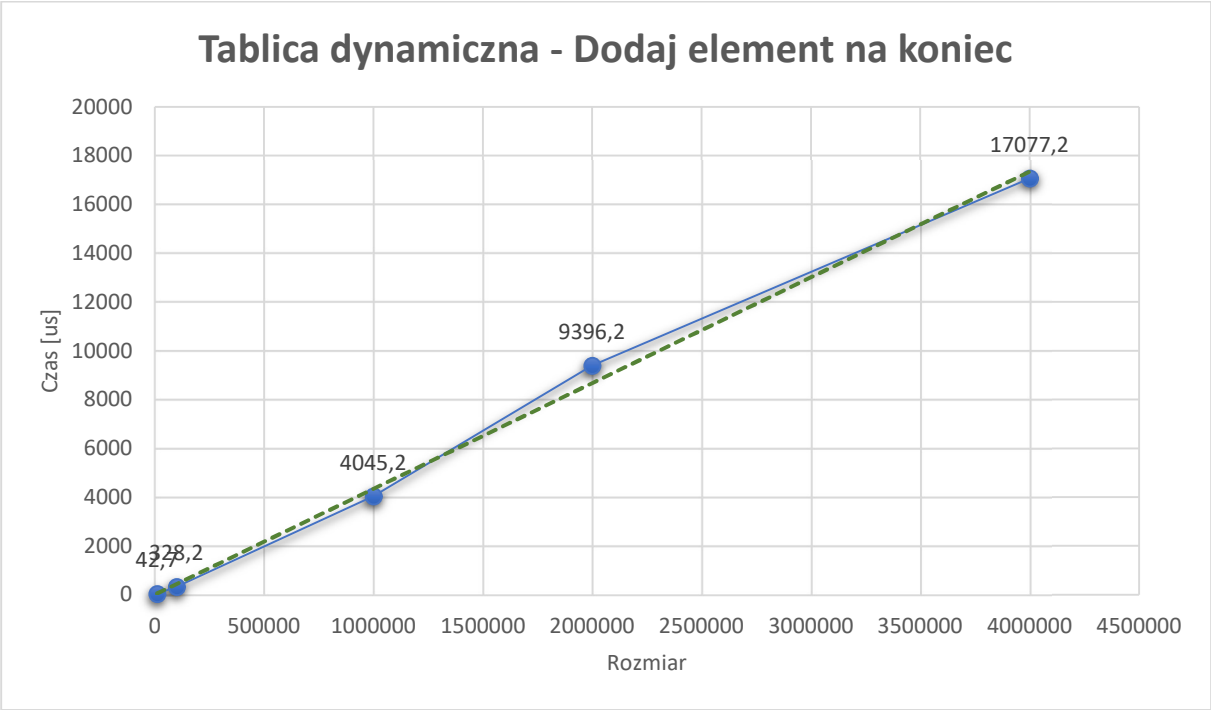
### Dodawanie elementu na pierwszy indeks




*Rysunek 1. Wyniki pomiaru czasu dodawania elementu na początek tablicy dynamicznej.*

Zgodnie z teorią otrzymano wykres linowy złożoności O(n).

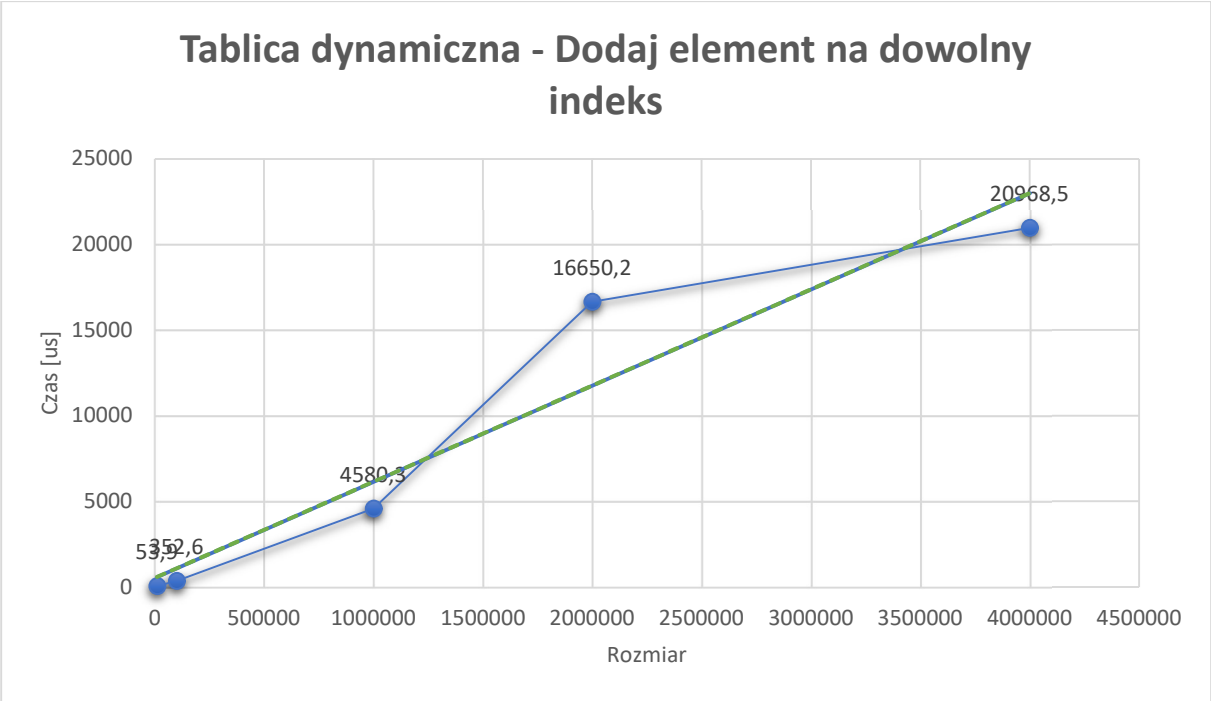
### Dodawanie elementu na ostatni indeks




*Rysunek 2. Wyniki pomiaru czasu dodawania elementu na koniec tablicy dynamicznej.*

Zgodnie z teorią otrzymano wykres liniowy.

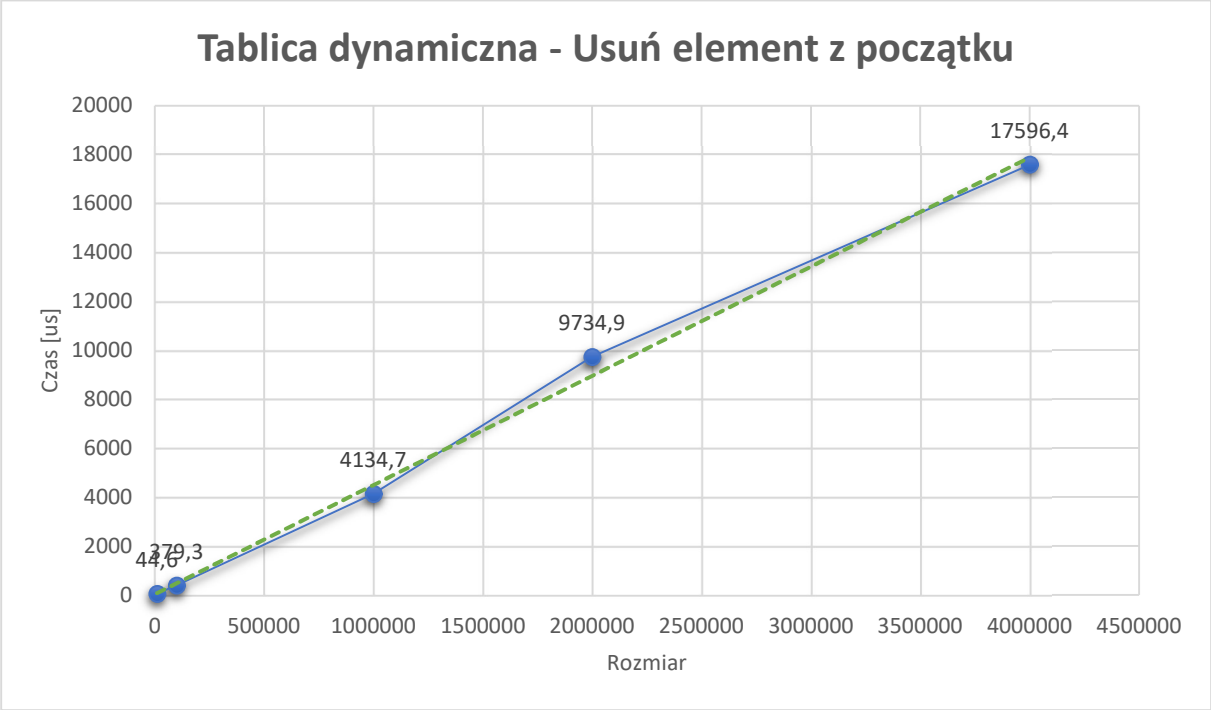
## Dodawanie elementu na dowolny indeks




*Rysunek 3. Wyniki pomiaru czasu dodawania elementu na dowolne miejsce w tablicy dynamicznej.*

Otrzymano w przybliżeniu wykres liniowy.

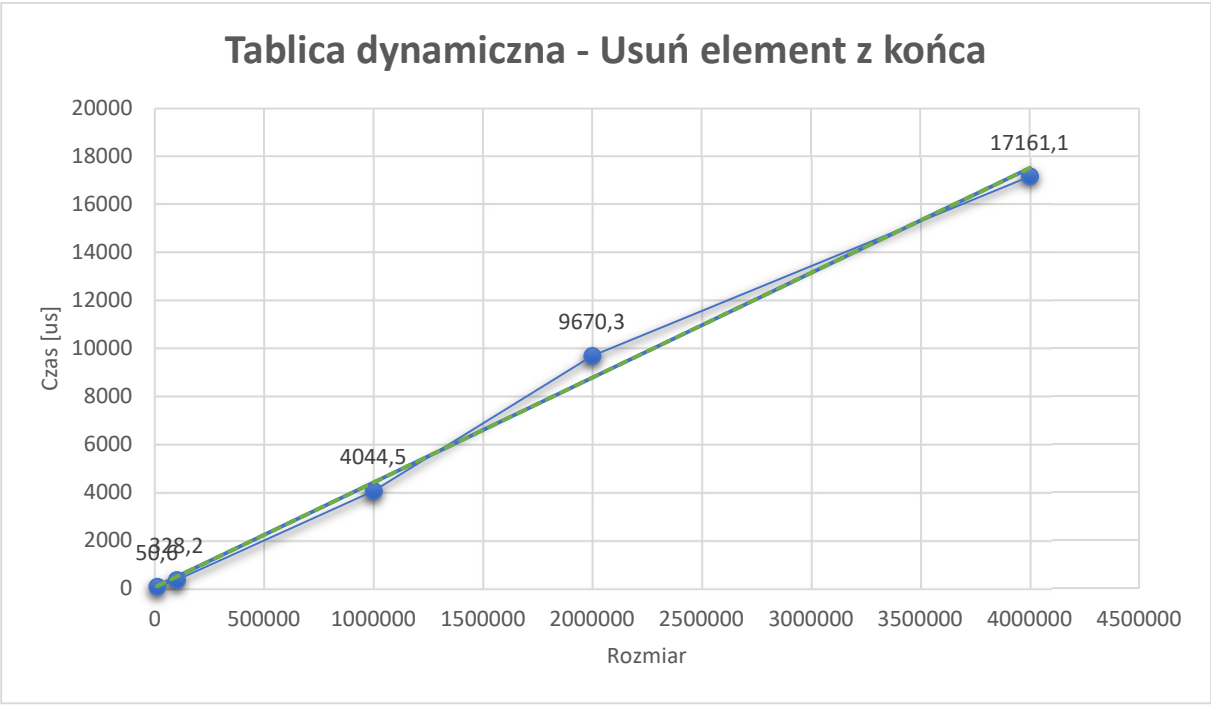
## Usuwanie pierwszego elementu




*Rysunek 4. Wyniki pomiaru czasu usuwania początkowego elementu z tablicy.*

Zgodnie z teorią otrzymano wykres liniowy.

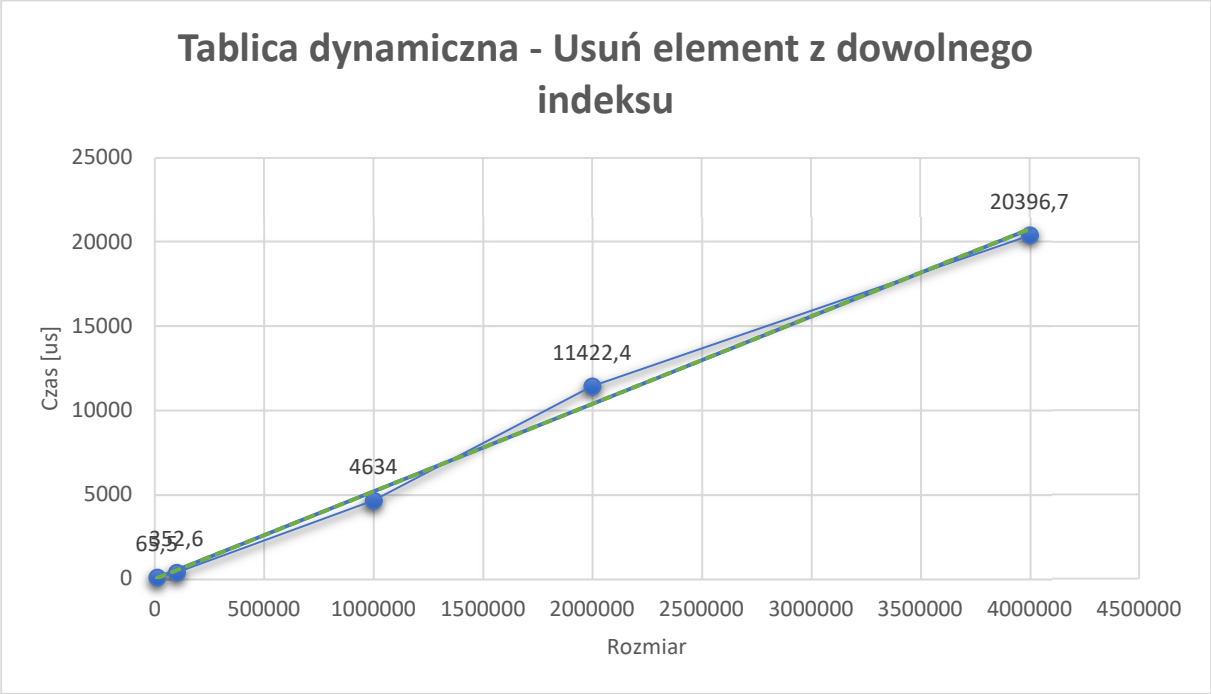
## Usuwanie ostatniego elementu




*Rysunek 5 Wyniki pomiaru czasu usuwania ostatniego elementu z tablicy.*

Zgodnie z teorią otrzymano wykres liniowy.

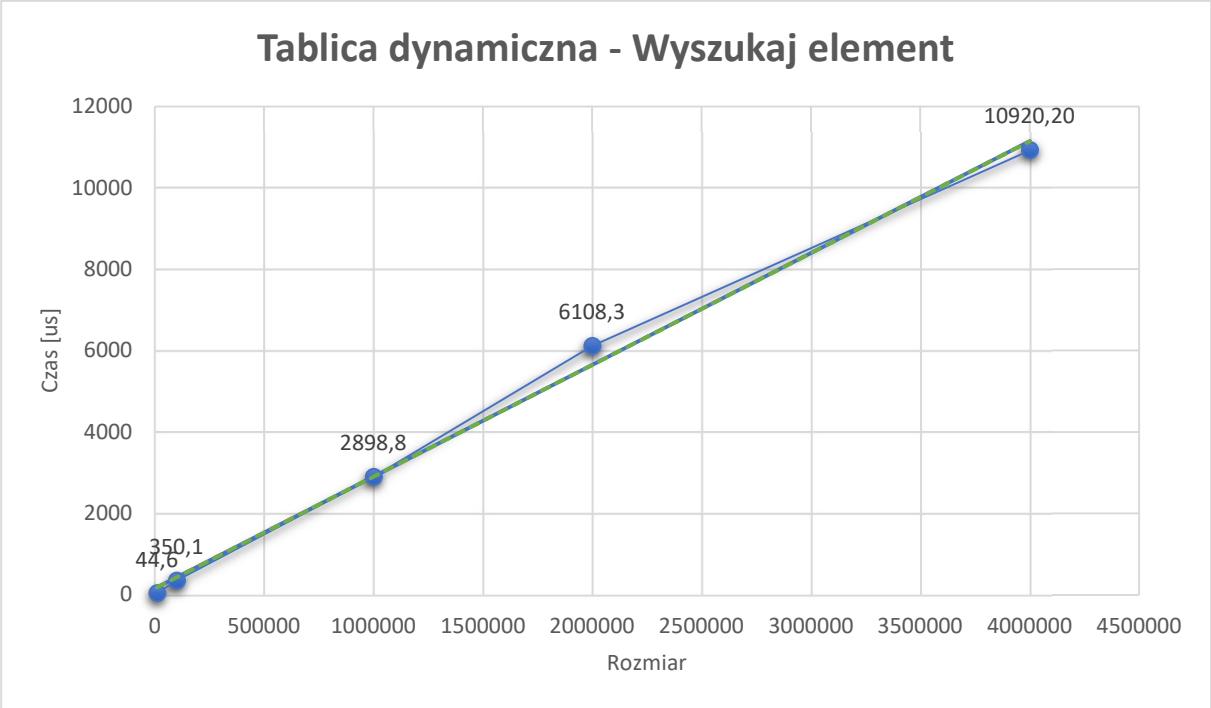
## Usuwanie dowolnego elementu




*Rysunek 6. Wyniki pomiaru czasu usuwania dowolnego elementu z tablicy.*

Zgodnie z teorią otrzymano wykres liniowy.

## Wyszukiwanie elementu

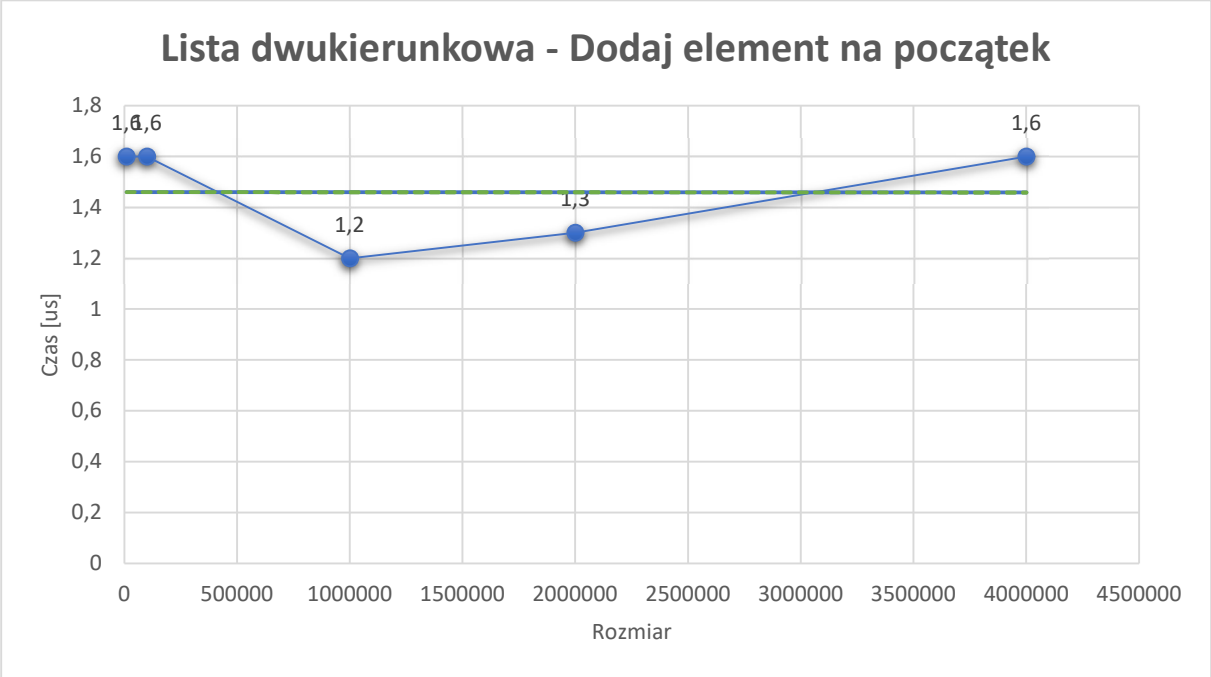



*Rysunek 7. Wyniki pomiaru wyszukiwania elementu w tablicy*

Zgodnie z teorią otrzymano wykres liniowy.

## Lista dwukierunkowa

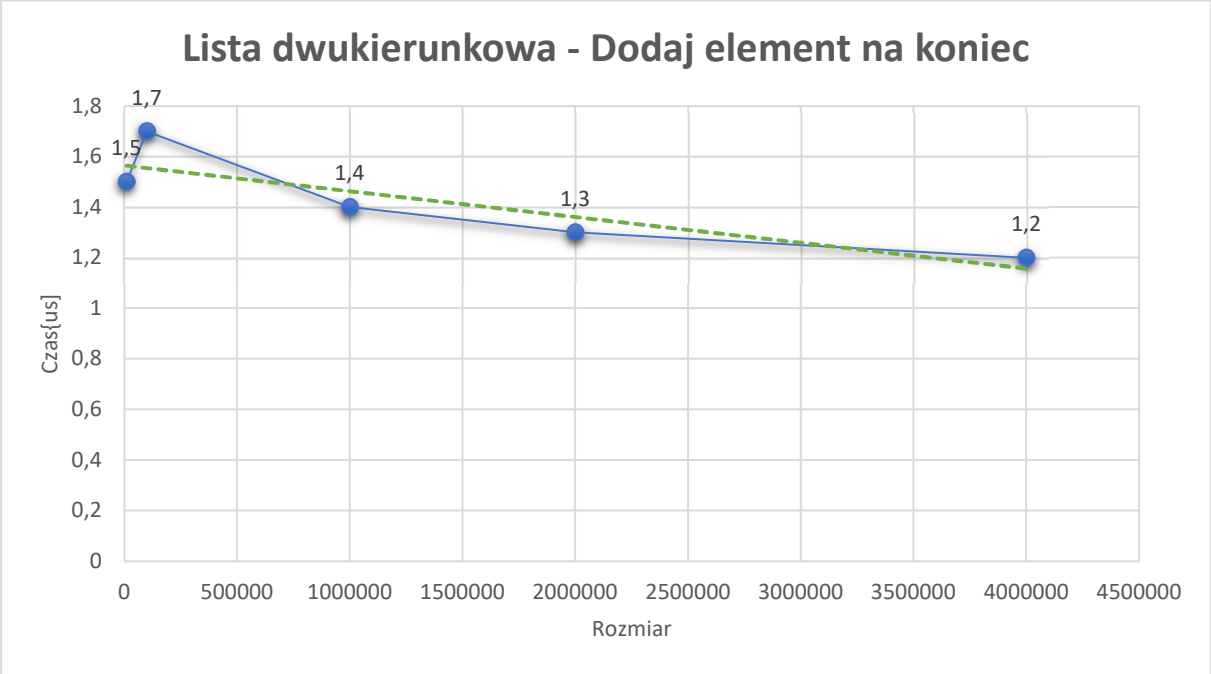
### Dodawanie elementu na pierwszy indeks




*Rysunek 8. Wyniki pomiaru czasu dodawania elementu na początek listy.*

Zgodnie z teorią otrzymano złożoność w przybliżeniu równą O(1), jak i czasy wykonania operacji są bardzo małe.

### Dodawanie elementu na ostatni indeks

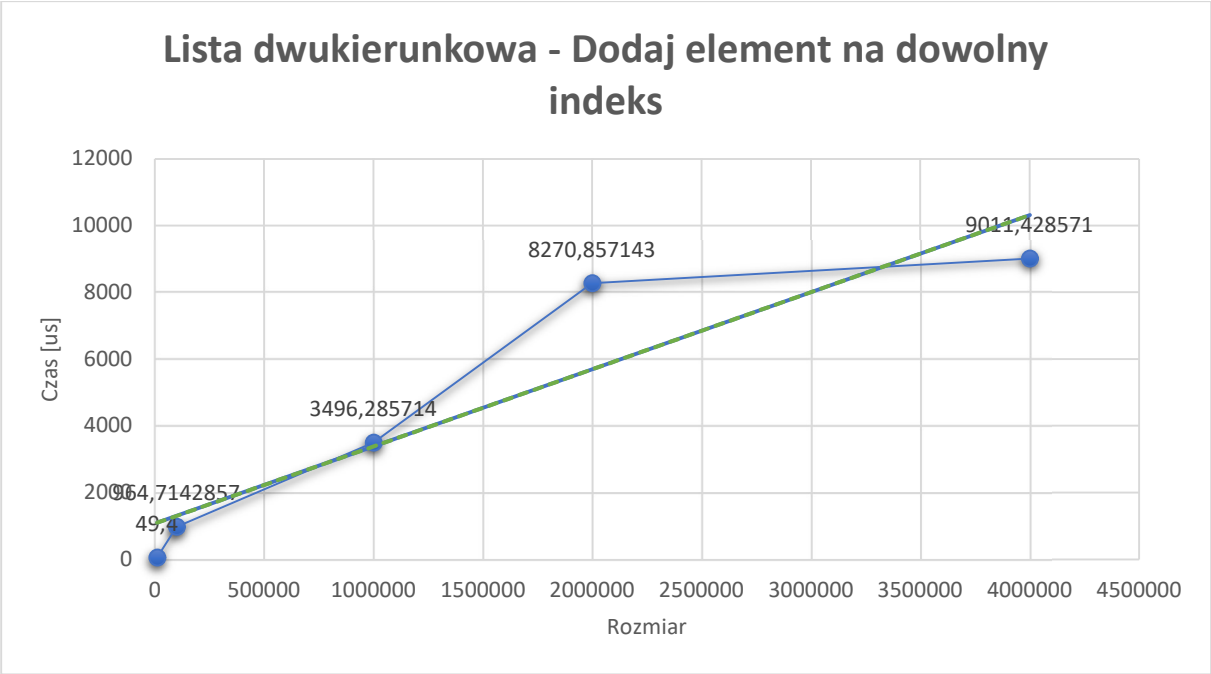



*Rysunek 9. Wyniki pomiaru czasu dodawania elementu na koniec listy.*

Na powyższym wykresie linia trendu sugeruje funkcje funkcję malejącą, co jest fizycznie niemożliwe. Na powyższy wynik najprawdopodobniej wpłynął bardzo mały czas wykonania operacji, przez co mogły powstać względnie duże błędy pomiaru.

Należy uznać, że w przybliżeniu otrzymano złożoność O(1).

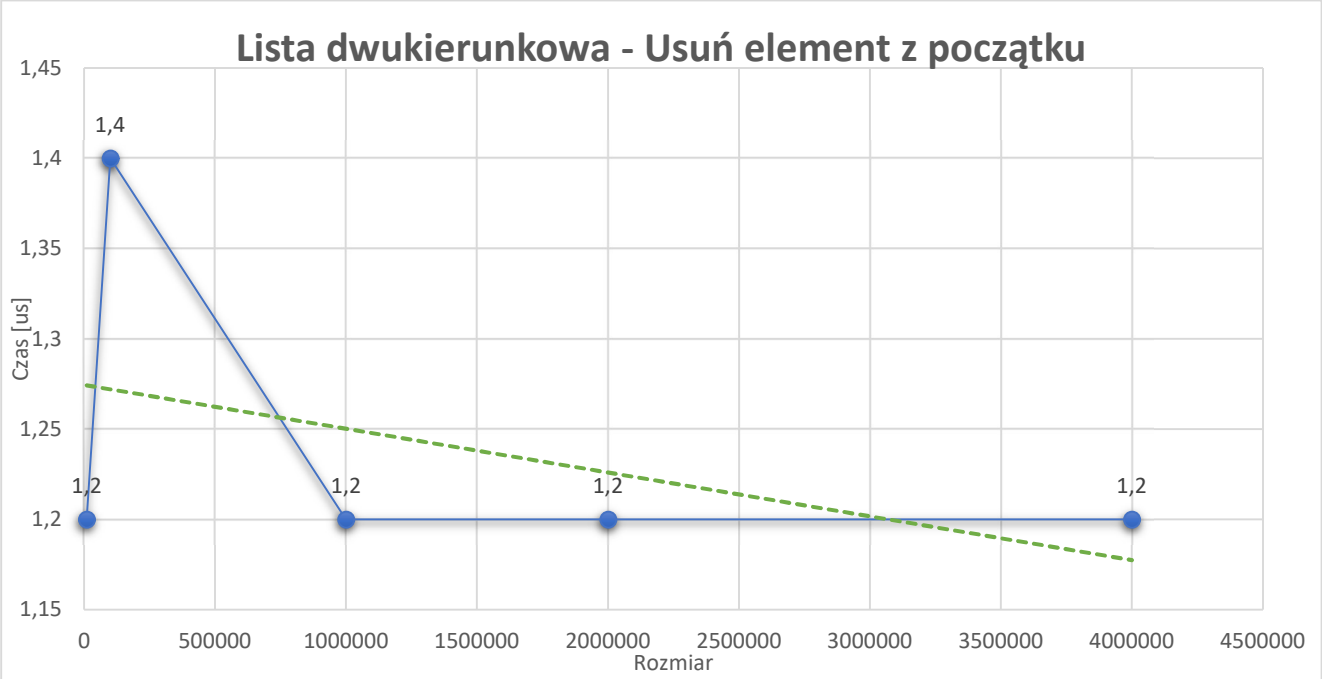
## Dodawanie elementu na dowolny indeks




*Rysunek 10. Wyniki pomiaru czasu dodawania elementu na dowolne miejsce w liście.*

Zgodnie z teorią otrzymano złożoność w przybliżeniu równą O(n). Powstał odchył pomiaru dla wartości 2.000.000 elementów, który mógł być spowodowany innymi procesami działającymi wtedy w komputerze.

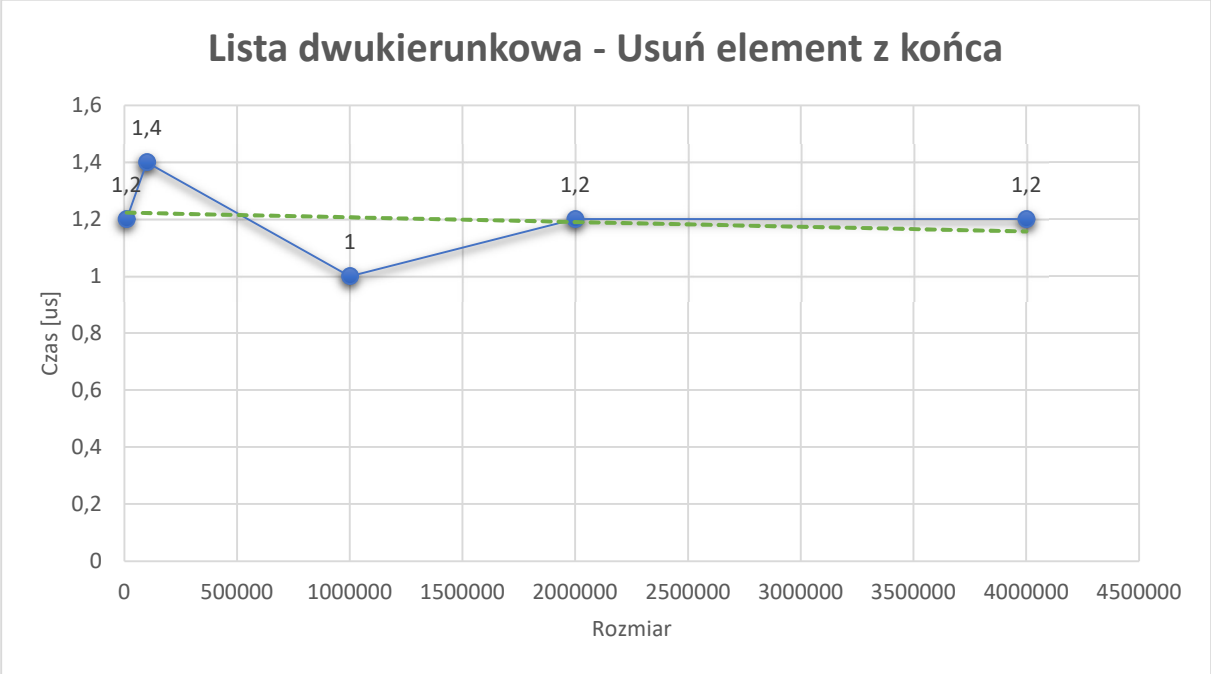
### Usuwanie pierwszego elementu




*Rysunek 11. Wyniki pomiaru czasu usuwania elementu z początku listy.*

Na powyższym wykresie otrzymano nagły skok, który jest względnie duży, jednak trzeba pamiętać, że jest to odchył rzędu setek nanosekund. Na odczyt wpływają więc względnie duże błędy pomiaru. Należy przyjąć, że otrzymana złożoność jest w przybliżeniu równa O(1). Najprawdopodobniej lepsze wyniki dałby pomiar w nanosekundach.

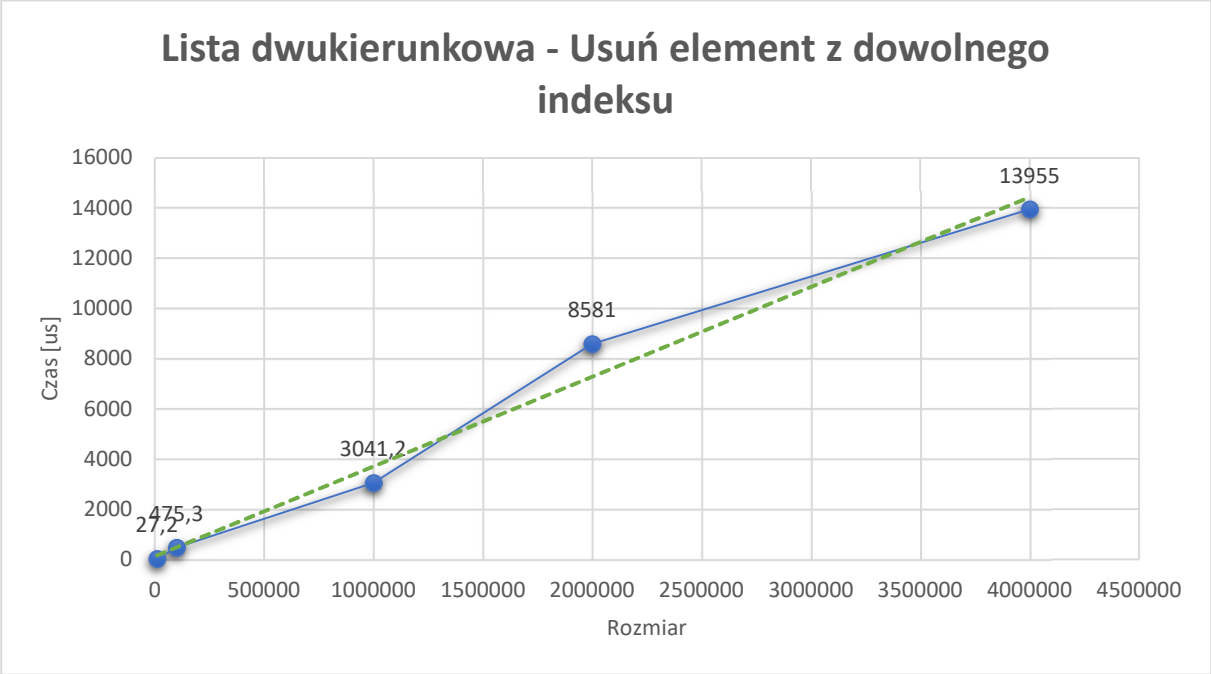
### Usuwanie ostatniego elementu




*Rysunek 12. Wyniki pomiaru czasu usuwania elementu z końca listy.*

Zgodnie z teorią otrzymano złożoność w przybliżeniu równą O(1).

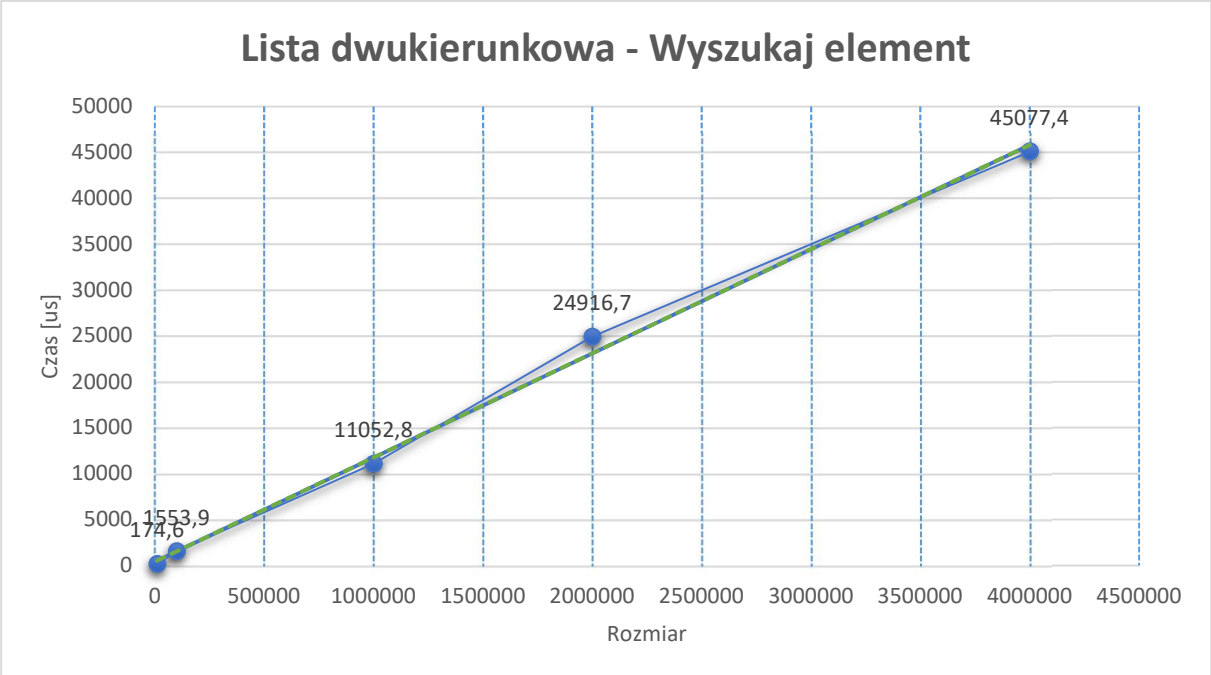
## Usuwanie dowolnego elementu




*Rysunek 13. Wyniki pomiaru czasu usuwania elementu z dowolnego miejsca z listy.*

Zgodnie z teorią otrzymano złożoność w przybliżeniu równą O(n).

## Wyszukiwanie elementu

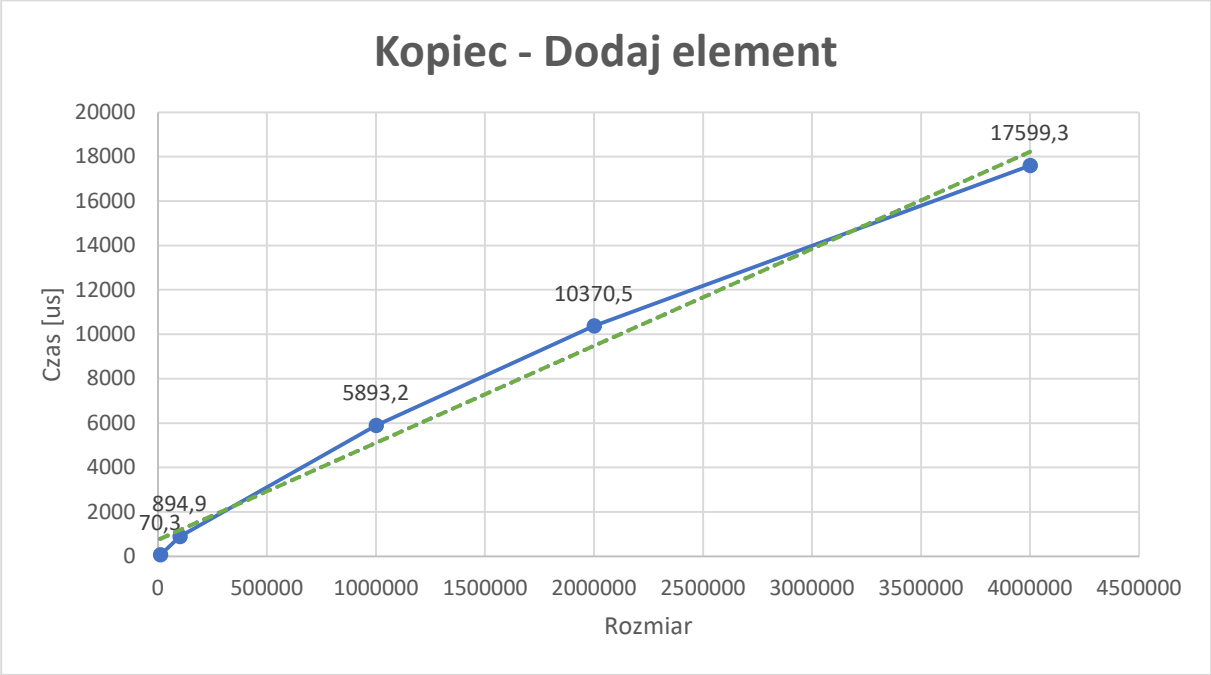



*Rysunek 14. Wyniki pomiaru szukania elementu w liście.*

Zgodnie z teorią otrzymano złożoność w przybliżeniu równą O(n).

## Kopiec binarny

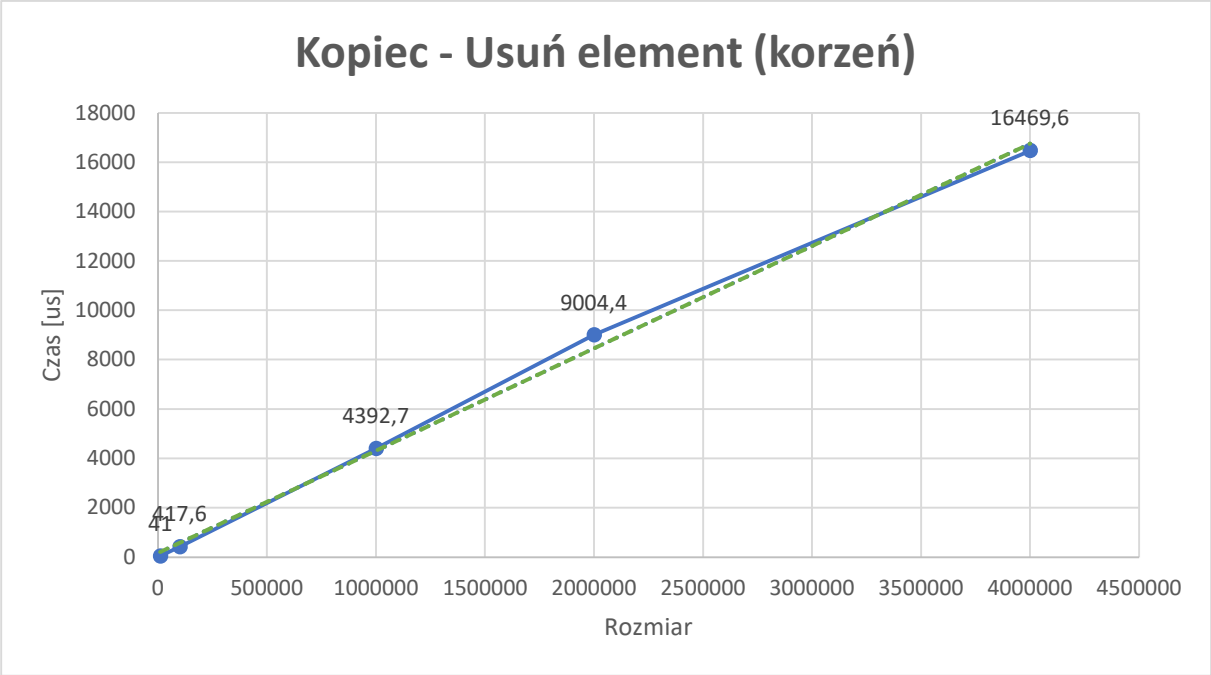
### Dodawanie elementu




*Rysunek 15. Wyniki pomiaru dodawania elementu do kopca.*

Zgodnie z teorią otrzymano złożoność w przybliżeniu równą O(nlgn).

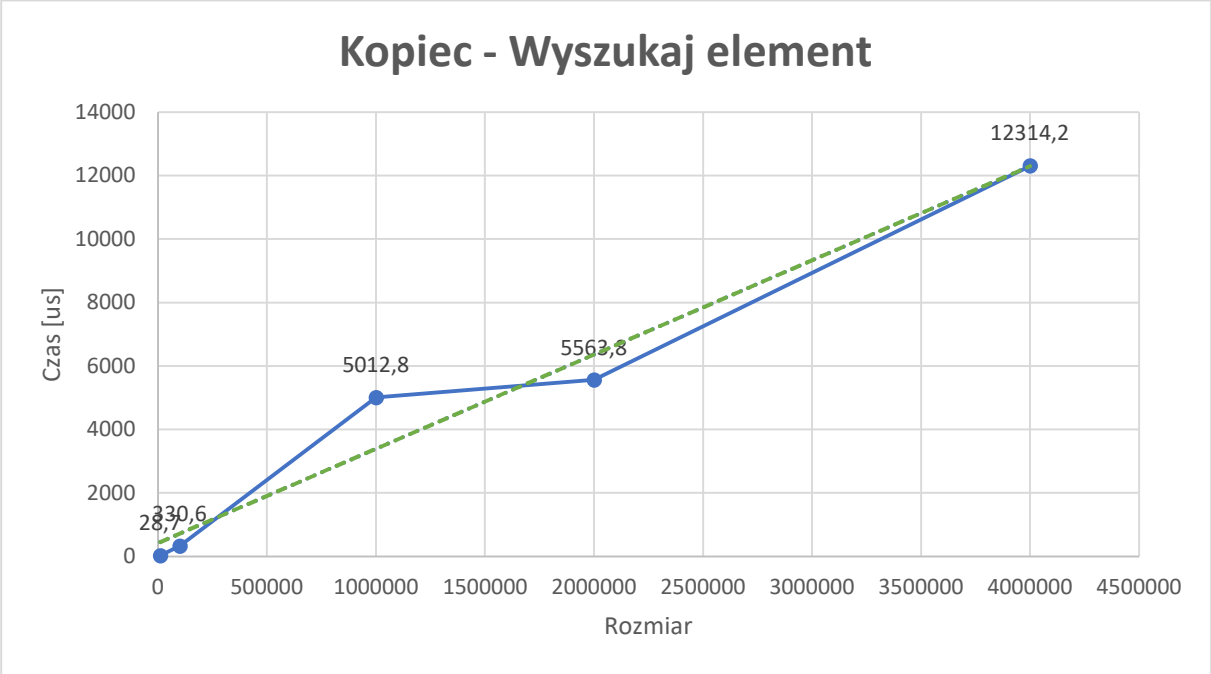
### Usuwanie elementu




*Rysunek 16. Wyniki pomiaru usuwania korzenia z kopca.*

Zgodnie z teorią otrzymano złożoność w przybliżeniu równą O(nlgn).

### Wyszukiwanie elementu

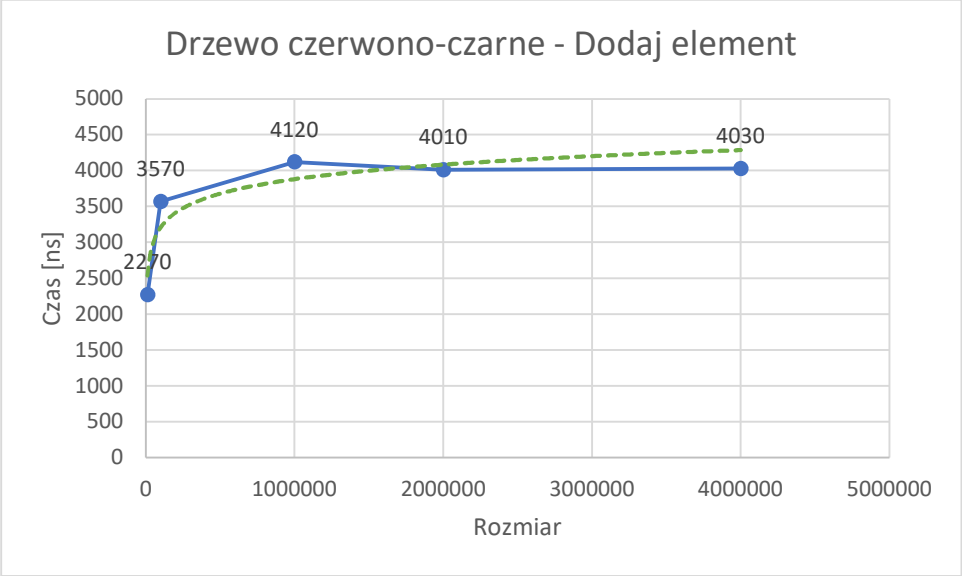



*Rysunek 17. Wyniki pomiaru wyszukiwania elementu w kopcu.*

Zgodnie z teorią otrzymano złożoność w przybliżeniu równą O(n).

## Drzewo czerwono – czarne

### Dodawanie elementu

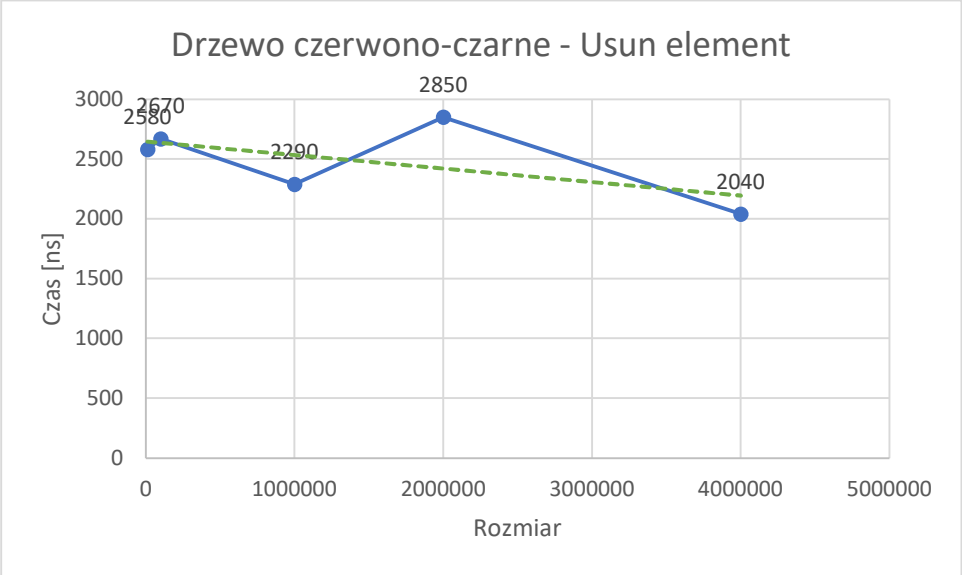



*Rysunek 18 Wyniki pomiaru dodawania elementu do drzewa RB.*

Zgodnie z teorią wykres przypomina złożoność O(lgn).

Czas (mierzony w nanosekundach) potrzebny do przeprowadzenia powyższej operacji był najmniejszy wśród wszystkich struktur.

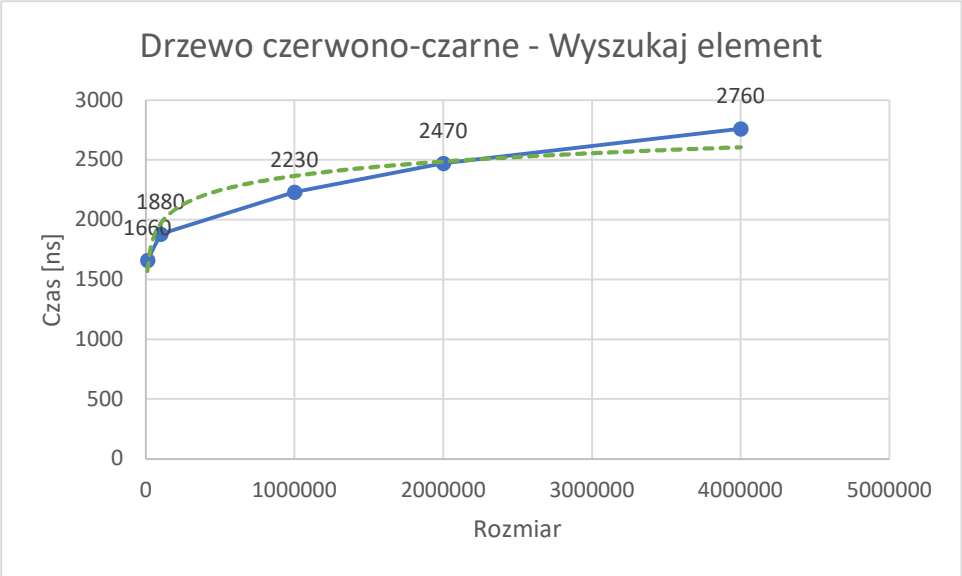
### Usuwanie elementu




*Rysunek 19. Wyniki pomiaru usuwania elementu z drzewa RB.*

Otrzymano wykres zupełnie niepodobny do wykresu oczekiwanego złożoności O(lgn). Na zakłócenie pomiaru mogły wpłynąć inne procesy działające wtedy w komputerze. Pomiary należałoby wykonać raz jeszcze.

### Wyszukiwanie elementu




*Rysunek 20. Wyniki pomiaru wyszukiwania elementu w drzewie RB.*

Zgodnie z teorią wykres przypomina złożoność O(lgn).

Czas (mierzony w nanosekundach) potrzebny do przeprowadzenia powyższej operacji był najmniejszy wśród wszystkich struktur.

# Wnioski

Wyniki powyższych badań były zgodne z założeniami teoretycznymi. Można stwierdzić, iż:

- Lista jest lepsza od tablicy w operacji dodawania oraz usuwania elementów. Tworzenie nowego elementu oraz ustawianie odpowiednio wskaźników wymaga mniej czasu oraz zasobów pamięci aniżeli alokowanie i przepisywanie nowej tablicy w inne miejsce pamięci.
- Tablica jest lepsza od listy w operacji wyszukiwania elementu. Indeksowanie po kolejnych elementach tablicy zajmuje znacznie mniej czasu niż „skakanie” po pamięci zgodnie z informacją zawartą we wskaźnikach elementów listy.
- Kopiec zaimplementowany jako tablica dynamiczna jest najmniej optymalną strukturą. Czasy wykonania poszczególnych operacji obejmują czas alokacji/relokacji tablicy dynamicznej oraz wykonania operacji charakterystycznych dla kopca.
- Drzewo RB jest najoptymalniejszą strukturą do przeprowadzenia operacji wstawiania, wyszukiwania elementu. Usuwania teoretycznie również, natomiast powyższy eksperyment tego nie dowodzi.

# Literatura


[1] Cormen Thomas H., Leiserson Charles E., Rivest Ronald L., Wprowadzenie do algorytmów, Wydanie czwarte, Wydawnictwo Naukowe PWN, Warszawa 2020

[2] Drozdek A., C++ Algorytmy i struktury danych, Wydanie drugie, Helion, Gliwice 2004

[3] Banachowski L., Rytter W., Marian K., Algorytmy i struktury danych, Wydawnictwo Naukowe PWN, Warszawa 2020

[4] Wałaszek J., Strona interentowa I Lo w Tarnowie: https://eduinf.waw.pl/inf/alg/001_search/index.php