Struktury danych	
Kierunek *Informatyczne Systemy Automatyki*	**Termin** *środa TN* $11^{15} - 13^{00}$
Imię, nazwisko, numer albumu *Maksym Kostyshyn 282651, Danil Krassotov 282656*	**Data** *09.04.2025*
Link do projektu `https://github.com/MaksyKost/DataStructures_DynamicArrays_LinkedLists`	

Sprawozdanie – miniprojekt nr1

Wstęp

Celem niniejszego miniptojektu jest implementacja wybranych struktur danych oraz analiza efektywności wykonywania różnych operacji na tych strukturach. W ramach pracy zrealizowano tablicę dynamiczną oraz listy jednokierunkowa i dwukierunkowa. Dla każdej struktury przeprowadzono pomiary czasu wykonania następujących operacji: dodawanie i usuwanie elementu na różnych pozycjach, wyszukiwanie. Teoretyczne złożoności obliczeniowe:

Tablica dynamiczna

Operacja	**Optymistycznie**	**Średnio**	**Pesymistycznie**
Dodanie na dowolnej pozycji	$O(n)$	$O(n)$	$O(n)$
Dodanie na początku	$O(n)$	$O(n)$	$O(n)$
Dodanie na końcu	$O(1)$	$O(1)$	$O(n)$
Usuwanie na początku	$O(n)$	$O(n)$	$O(n)$
Usuwanie na końcu	$O(1)$	$O(1)$	$O(1)$
Usuwanie na dowolnej pozycji	$O(n)$	$O(n)$	$O(n)$
Wyszukiwanie	$O(1)$	$O(n)$	$O(n)$

Tabela 1: Złożoność operacji dla tablicy dynamicznej

Przykłady zastosowań:

- Implementacja stosu, tablic, vector (C++)
- Nadaje się do częstego dostępu do indeksu, wyszukiwania indeksu.

Zalety:

- Szybki dostęp do elementów - indeks
- Szybkie dodawanie na końcu
- Zajmuje mało miejsca, chyba że nastąpi rozszerzanie - droga pamięciowo operacja

Lista jednokierunkowa

Operacja	**Optymistycznie**	**Średnio**	**Pesymistycznie**
Dodanie na dowolnej pozycji	$O(i)$	$O(n)$	$O(n)$
Dodanie na początku	$O(1)$	$O(1)$	$O(1)$
Dodanie na końcu	$O(n)$	$O(n)$	$O(n)$
Usuwanie na początku	$O(1)$	$O(1)$	$O(1)$
Usuwanie na końcu	$O(n)$	$O(n)$	$O(n)$
Usuwanie na dowolnej pozycji	$O(i)$	$O(n)$	$O(n)$
Wyszukiwanie	$O(1)$	$O(n)$	$O(n)$

Tabela 2: Złożoność operacji dla listy jednokierunkowej

Przykłady zastosowań:

- Implementacja kolejki oraz struktur, gdzie ważne jest dodawanie na początku
- Jeśli liczba elementów nie jest znana z góry, pamięć jest przydzielana w razie potrzeby.

Zalety:

- Szybkie dodawanie/usuwanie na początku
- Zajmuje mniej pamięci w porównaniu do listy dwukierunkowej struktur

Lista dwukierunkowa

Operacja	**Optymistycznie**	**Średnio**	**Pesymistycznie**
Dodanie na dowolnej pozycji	$O(min(i, n-1))$	$O(n)$	$O(n)$
Dodanie na początku	$O(1)$	$O(1)$	$O(1)$
Dodanie na końcu	$O(1)$	$O(1)$	$O(1)$
Usuwanie na początku	$O(1)$	$O(1)$	$O(1)$
Usuwanie na końcu	$O(1)$	$O(1)$	$O(1)$
Usuwanie na dowolnej pozycji	$O(min(i, n-1))$	$O(n)$	$O(n)$
Wyszukiwanie	$O(1)$	$O(n)$	$O(n)$

Tabela 3: Złożoność operacji dla lista dwukierunkowej

Przykłady zastosowań:

- Implementacja double-ended queue
- Jeśli liczba elementów nie jest znana z góry, pamięć jest przydzielana w razie potrzeby.

Zalety:

- Szybkie dodawanie/usuwanie na dowolnej pozycji
- Poruszanie się w obie strony

Zródło: *https://kam.pwr.edu.pl/jaroslaw-rudypwr-edu-pl/files/sd/w3.pdf*

1 Założenia projektowe

Do przeprowadzenia analizy operacji na strukturach, przyjęto następujące rozmiary danych: N={10000, 20000, 30000, 40000, 50000, 60000, 70000, 80000, 90000, 100000}. Elementy zostały wygenerowane za pomocą generatora liczb pseudolosowych *srand()* z zastosowaniem różnych wartości ziarna (seed) wykorzystując funkcje *rand()*, aby uzyskać statystycznie wiarygodne wyniki. Dla każdego rozmiaru struktury wylosowano 10 różnych wartości ziarna, a dla każdej z nich wykonano po 100 niezależnych pomiarów czasu. **W rezultacie, dla każdej metody operacji uzyskano łącznie 1000 pomiarów, z których obliczano wartości średnie.**

Wykorzystany sprzęt:

Testy zostały przeprowadzone na laptopie **Lenovo Legion Slim 5 16AHP9**

- procesor AMD Ryzen 7 8845HS w/ Radeon 780M Graphics 3.80 GHz
- RAM 32,0 GB
- 64-bitowy system operacyjny, procesor oparty na architekturze x64.

2 Wyniki: tabeli i wykresy czasów

2.1 Dane pomiarowe

Tabela 4: Dodanie na początek

Struktura [N]	**10000**	**20000**	**30000**	**50000**	**70000**	**100000**
Tablica dynamiczna [s]	1.05E-05	2.02E-05	3.07E-05	5.06E-05	7.03E-05	9.80E-05
Lista jednokierunkowa [s]	5.85E-08	6.08E-08	7.91E-08	1.03E-07	1.07E-07	1.20E-07
Lista dwukierunkowa [s]	5.17E-08	5.23E-08	6.03E-08	5.28E-08	6.60E-08	6.60E-08

Tabela 5: Dodanie na koniec

Struktura [N]	**10000**	**20000**	**30000**	**50000**	**70000**	**100000**
Tablica dynamiczna [s]	3.29E-08	3.29E-08	3.06E-08	3.26E-08	3.41E-08	3.11E-08
Lista jednokierunkowa [s]	1.34E-05	2.64E-05	4.56E-05	1.02E-04	1.41E-04	2.01E-04
Lista dwukierunkowa [s]	4.56E-08	4.93E-08	4.59E-08	4.90E-08	4.79E-08	5.23E-08

Tabela 6: Dodanie w losowym miejscu

Struktura [N]	**10000**	**20000**	**30000**	**50000**	**70000**	**100000**
Tablica dynamiczna [s]	6.58E-06	1.31E-05	1.89E-05	3.75E-05	5.93E-05	8.99E-05
Lista jednokierunkowa [s]	7.50E-06	7.54E-06	8.44E-06	1.08E-05	1.11E-05	1.17E-05
Lista dwukierunkowa[s]	3.49E-05	6.87E-05	1.10E-04	1.89E-04	2.72E-04	3.98E-04

Tabela 7: Usunięcie z początku

Struktura [N]	**10000**	**20000**	**30000**	**50000**	**70000**	**100000**
Tablica dynamiczna [s]	1.23E-05	2.38E-05	3.62E-05	5.97E-05	8.38E-05	1.16E-04
Lista jednokierunkowa [s]	3.61E-08	3.73E-08	3.69E-08	3.82E-08	3.69E-08	4.19E-08
Lista dwukierunkowa [s]	5.81E-08	1.17E-07	1.36E-07	1.46E-07	1.73E-07	1.89E-07

Tabela 8: Usunięcie z końca

Struktura [N]	**10000**	**20000**	**30000**	**50000**	**70000**	**100000**
Tablica dynamiczna [s]	3.18E-08	2.73E-08	2.63E-08	3.13E-08	3.10E-08	2.88E-08
Lista jednokierunkowa [s]	1.37E-05	2.71E-05	4.53E-05	9.77E-05	1.37E-04	1.96E-04
Lista dwukierunkowa [s]	6.78E-08	1.07E-07	1.67E-07	2.23E-07	2.38E-07	2.27E-07

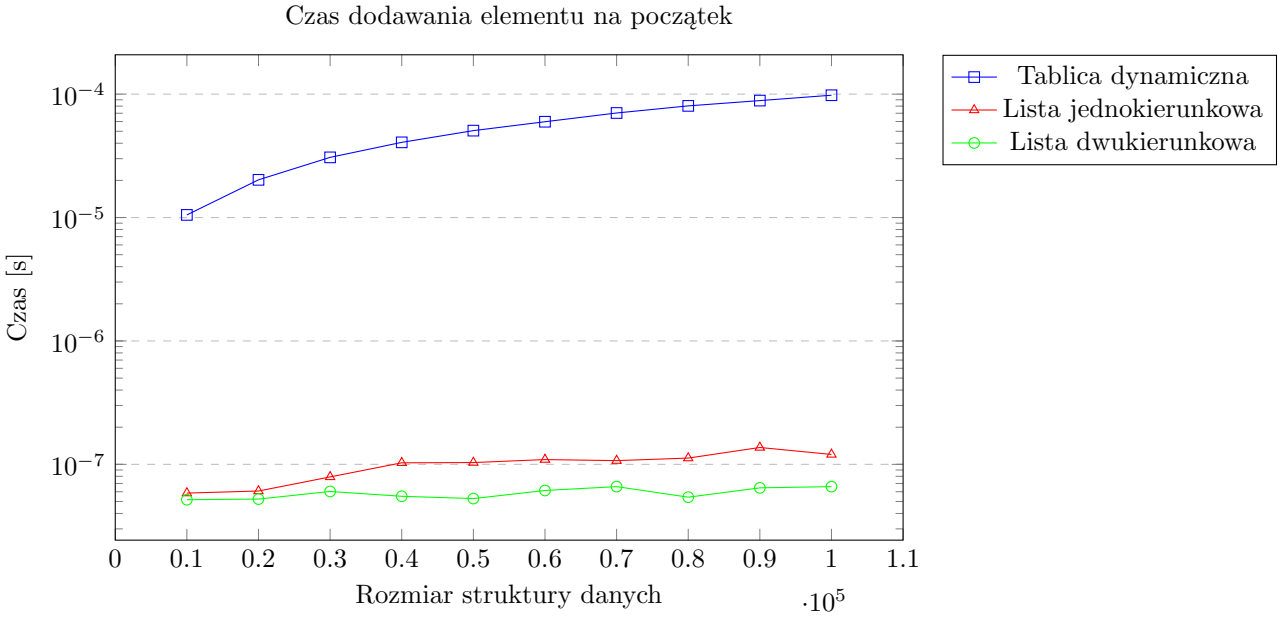
Tabela 9: Usunięcie z losowego miejsca

Struktura [N]	**10000**	**20000**	**30000**	**50000**	**70000**	**100000**
Tablica dynamiczna [s]	1.04E-05	2.19E-05	3.43E-05	5.77E-05	8.23E-05	1.15E-04
Lista jednokierunkowa [s]	2.64E-06	2.74E-06	2.85E-06	3.07E-06	3.17E-06	3.09E-06
Lista dwukierunkowa [s]	1.84E-05	3.62E-05	5.94E-05	1.13E-04	1.57E-04	2.06E-04

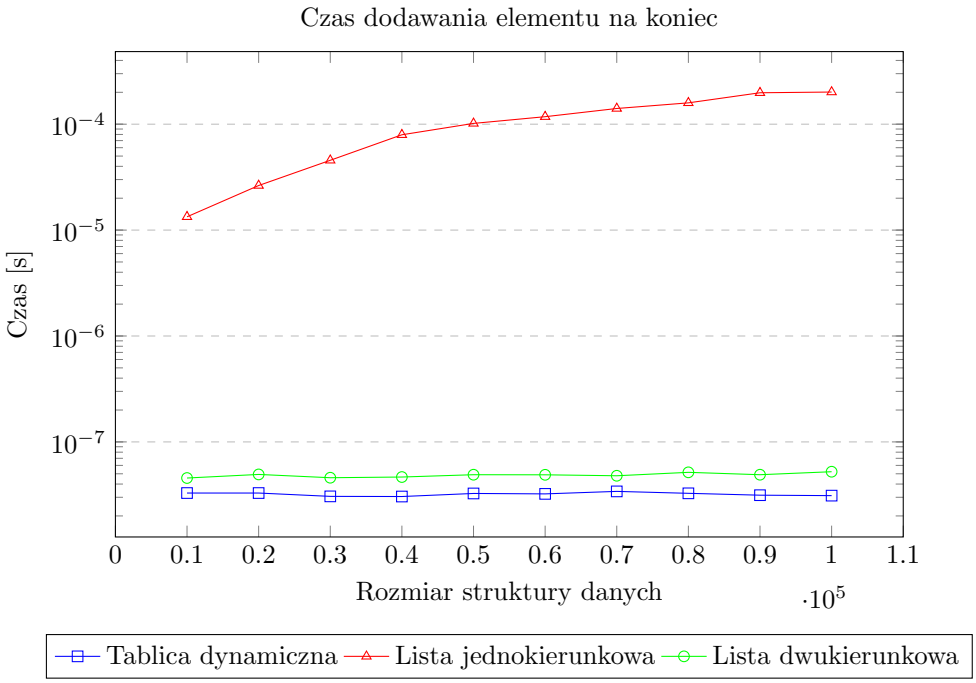
Tabela 10: Wyszukiwanie

Struktura [N]	**10000**	**20000**	**30000**	**50000**	**70000**	**100000**
Tablica dynamiczna [s]	3.03E-08	3.18E-08	2.92E-08	3.10E-08	2.95E-08	2.92E-08
Lista jednokierunkowa [s]	3.63E-08	3.44E-08	3.85E-08	6.56E-08	6.92E-08	6.98E-08
Lista dwukierunkowa [s]	1.03E-05	1.19E-05	1.48E-05	2.16E-05	2.88E-05	2.48E-05

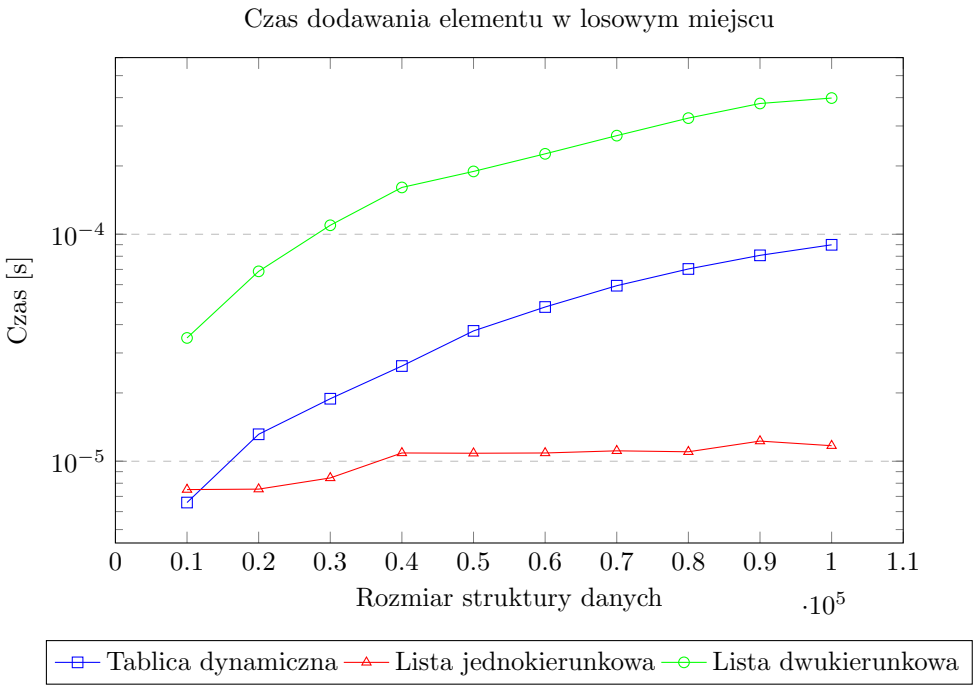
2.2 Wykresy porównawcze



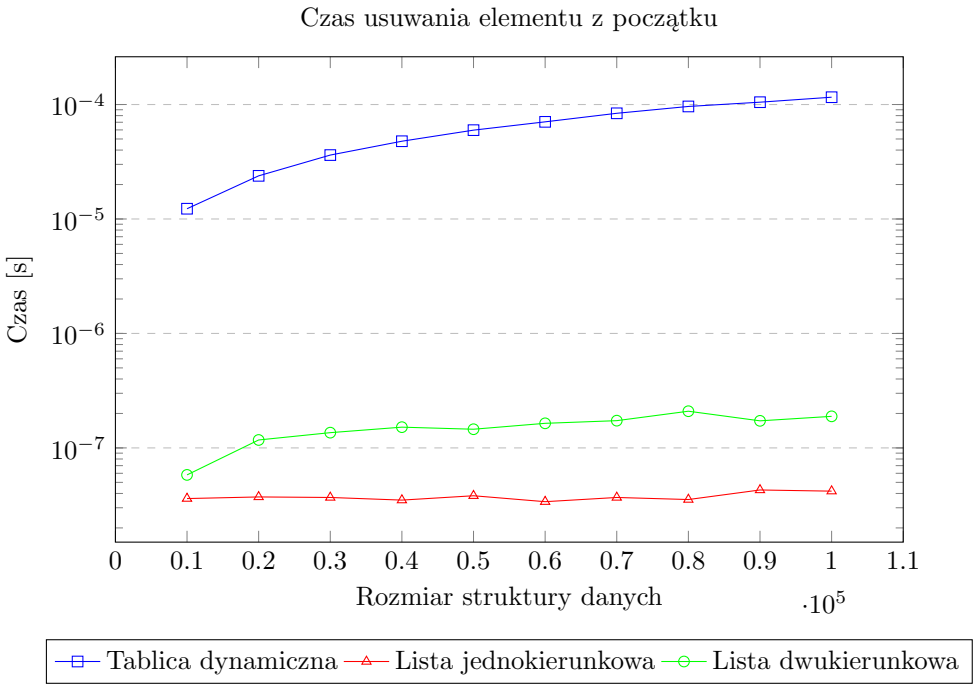
Rysunek 1: Porównanie czasu dodawania elementu na początek dla różnych struktur



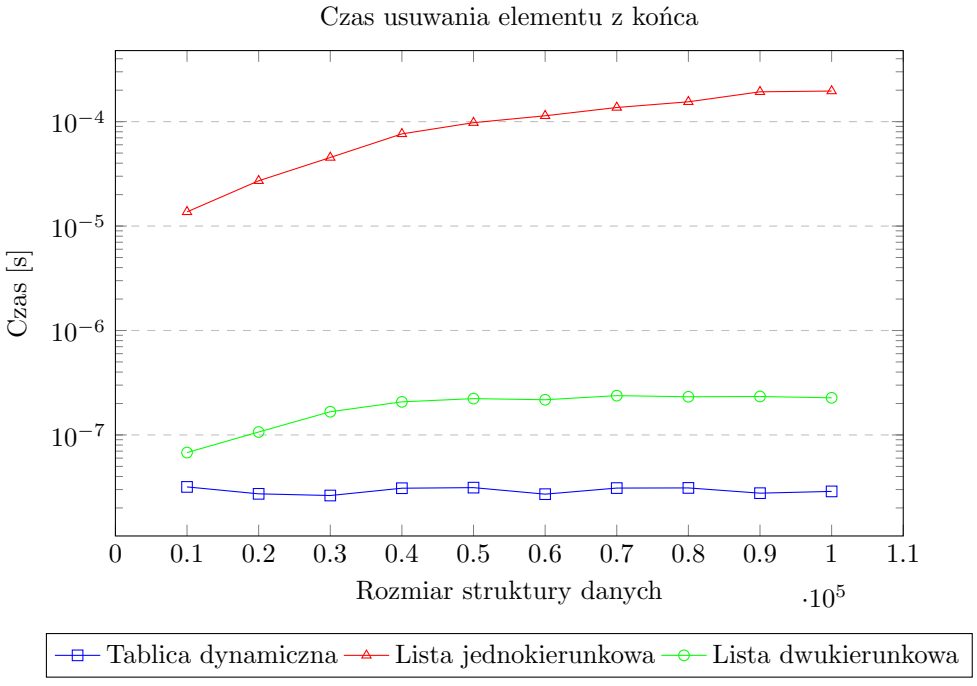
Rysunek 2: Porównanie czasu dodawania elementu na koniec dla różnych struktur



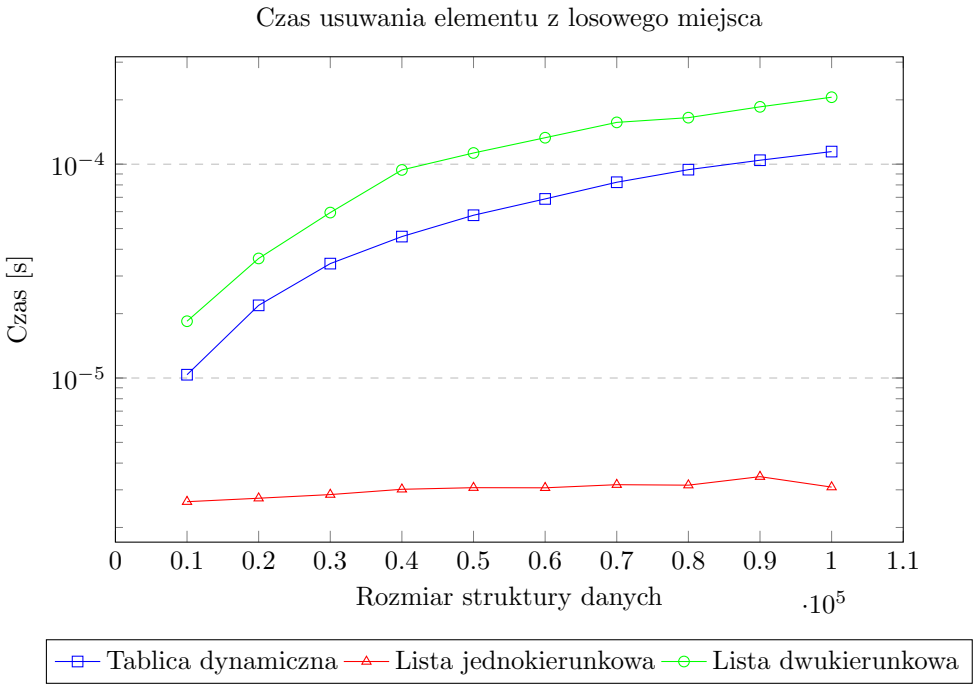
Rysunek 3: Porównanie czasu dodawania elementu w losowym miejscu dla różnych struktur



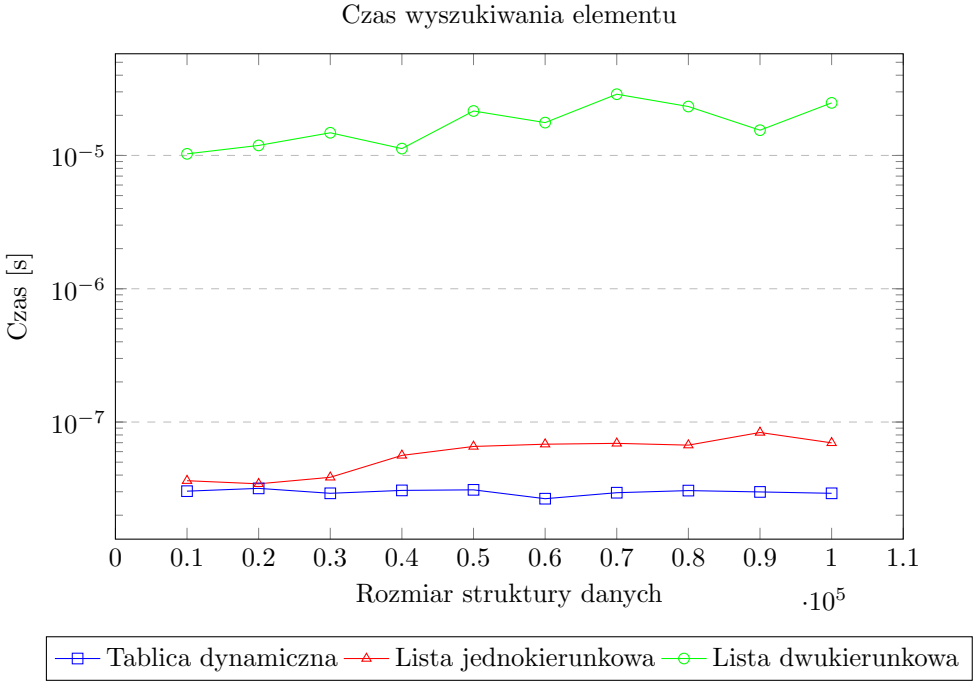
Rysunek 4: Porównanie czasu usuwania elementu z początku dla różnych struktur



Rysunek 5: Porównanie czasu usuwania elementu z końca dla różnych struktur

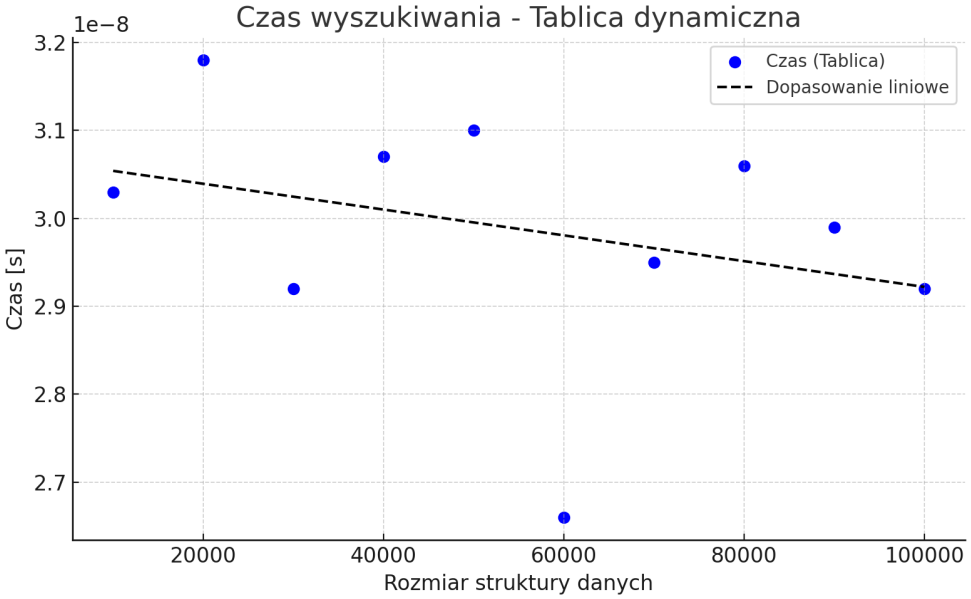


Rysunek 6: Porównanie czasu usuwania elementu z losowego miejsca dla różnych struktur

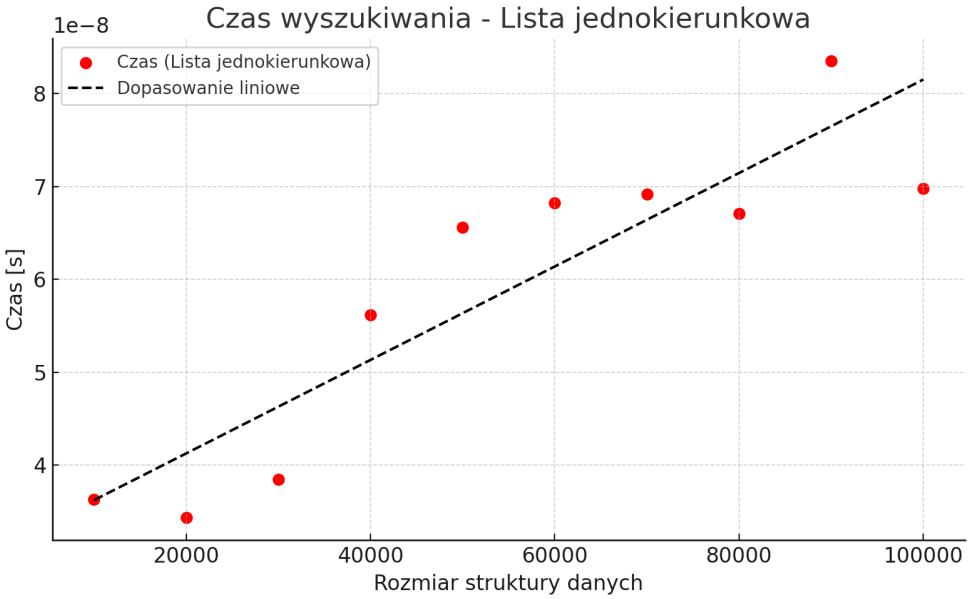


Rysunek 7: Porównanie czasu wyszukiwania elementu dla różnych struktur

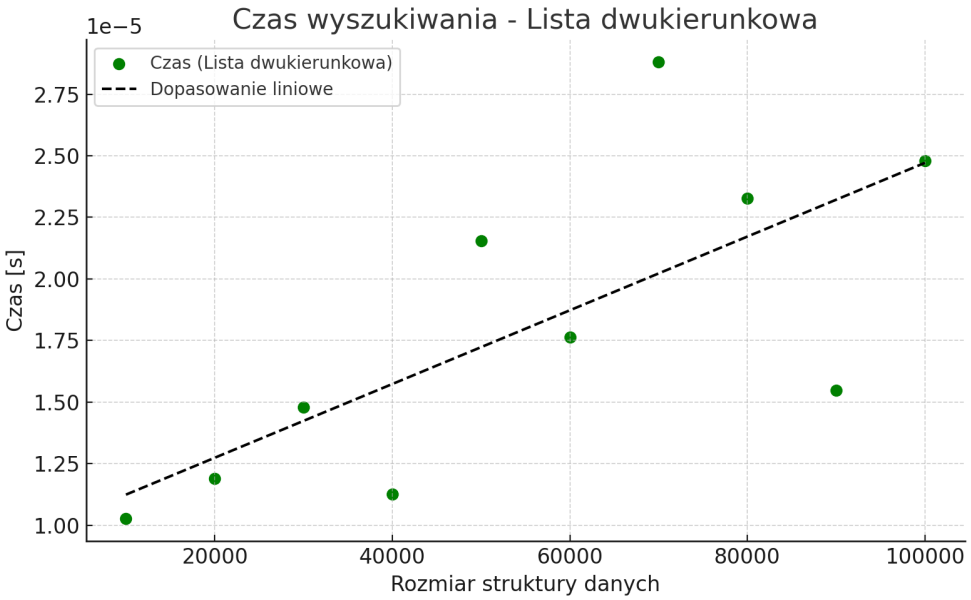
2.3 Dopasowanie liniowe



Rysunek 8: Dopasowanie liniowe do **tablicy dynamicznej**.



Rysunek 9: Dopasowanie liniowe do **listy jednokierunkowej**.



Rysunek 10: Dopasowanie liniowe do **listy dwukierunkowej**.

Regresje (przybliżone współczynniki)

Po przeliczeniu, uzyskamy:

1. **Tablica dynamiczna**:
   $y_{array} = -1.66 \cdot 10^{-13}x + 3.08 \cdot 10^{-8}$
2. **Lista jednokierunkowa**:
   $y_{lista1} = 3.79 \cdot 10^{-13}x + 2.89 \cdot 10^{-8}$
3. **Lista dwukierunkowa**:
   $y_{lista2} = 1.47 \cdot 10^{-10}x + 2.8 \cdot 10^{-6}$

Interpretacja:

- **Tablica dynamiczna**: czas wyszukiwania praktycznie nie zależy od rozmiaru (płaska linia), zgodnie z teoretycznym $O(1)$.
- **Lista jednokierunkowa**: minimalny wzrost z rozmiarem – zgodnie z $O(n)$, ale dla dużych danych trend jest liniowy.
- **Lista dwukierunkowa**: wyraźna zależność liniowa – też zgodna z $O(n)$, ale z większym współczynnikiem.

3 Wnioski

Na podstawie przeprowadzonych badań oraz analizy dopasowania liniowego można wyciągnąć następujące wnioski:

- **Dodawanie na początek:** Listy (zarówno jednokierunkowa, jak i dwukierunkowa) są znacznie szybsze od tablicy dynamicznej, co jest zgodne z teorią – w liście ta operacja ma złożoność $O(1)$, natomiast w tablicy dynamicznej wymaga przesunięcia elementów ($O(n)$).
- **Dodawanie na koniec:** Tablica dynamiczna oraz lista dwukierunkowa oferują podobną wydajność, co również pokrywa się z teorią ($O(1)$ w średnim przypadku). Lista jednokierunkowa jest wolniejsza z powodu konieczności iteracji przez całą listę, mimo posiadania wskaźnika `tail`.

- **Dodawanie w losowym miejscu:** Wyniki pokazują lepszą wydajność listy jednokierunkowej niż dwukierunkowej, co może wynikać z konkretnej implementacji lub kosztu aktualizacji dwóch wskaźników (prev i next) w liście dwukierunkowej.
- **Usuwanie z początku:** Zarówno lista jednokierunkowa, jak i dwukierunkowa wypadają znacznie lepiej niż tablica dynamiczna, co jest zgodne z teorią – usunięcie z początku listy ma złożoność $O(1)$, podczas gdy w tablicy wymaga przesunięcia elementów ($O(n)$).
- **Usuwanie z końca:** Tablica dynamiczna jest najszybsza, co wynika z możliwości bezpośredniego dostępu do ostatniego elementu ($O(1)$). Lista dwukierunkowa jest umiarkowanie szybka, a lista jednokierunkowa najwolniejsza, ze względu na konieczność przejścia całej listy do przedostatniego elementu.
- **Usuwanie z losowego miejsca:** Lista jednokierunkowa wypada najlepiej, co może być efektem konkretnej implementacji. Teoretycznie, wszystkie struktury mają tutaj złożoność $O(n)$.
- **Wyszukiwanie:** Na podstawie wykresów i dopasowania liniowego widać, że:
  - Dla **tablicy dynamicznej** czas wyszukiwania praktycznie nie zależy od rozmiaru struktury – zgodnie z teoretyczną złożonością $O(1)$. Dopasowanie liniowe to niemal pozioma linia.
  - Dla **listy jednokierunkowej** oraz **dwukierunkowej** widoczny jest wzrost czasu liniowo względem rozmiaru, co potwierdza złożoność $O(n)$.
  - Lista dwukierunkowa ma największy współczynnik kierunkowy w dopasowaniu liniowym, co wskazuje na największy koszt wyszukiwania spośród badanych struktur.

Porównując struktury między sobą:

- **Tablica dynamiczna:** Najlepsza w operacjach na końcu oraz wyszukiwaniu, najsłabsza przy operacjach na początku.
- **Lista jednokierunkowa:** Bardzo dobra w operacjach na początku i przy losowym dostępie, słabsza na końcu.
- **Lista dwukierunkowa:** Stabilna w operacjach na końcu i początku, ale najwolniejsza w wyszukiwaniu i losowym dostępie.

Wyniki eksperymentalne w większości potwierdzają teoretyczne złożoności czasowe poszczególnych operacji. Drobne odchylenia mogą wynikać z implementacji, działania kompilatora lub wpływu środowiska uruchomieniowego (np. cache procesora).