Struktury danych i złożoność obliczeniowa – Projekt

Sprawozdanie

Uladzimir Kaviaka(257276)

Zadanie projektowe 1	
Badanie efektywności operacji dodawania, usuwania oraz wyszukiwania elementów w różnych strukturach danych	
Prowadzący	Mgr. Inż. Antoni Sterna
Termin zajęć	TN, poniedziałek, 11:15
Termin oddania:	01.04.2021

Spis treści

Wstęp

Projekt polega na napisaniu programu w języku bez maszyny wirtualnej i zmierzenie czasu wykonania dla funkcji dodawania, usuwania i wyszukiwania elementu dla podanych niżej struktur danych:

- Tablica dynamiczna
- Lista dwukierunkowa
- Kopiec binarny typu MAX
- Drzewo BST

Przy realizowaniu projektu uwzględniono następujące założenia:

- Elementem wszystkich struktur jest 4 bajtowa liczba całkowita ze znakiem
- Wszystkie struktury alokowane dynamicznie i zajmują jak najmniej miejsca. Kopiec zaimplementowany w wariancie z tablicą
- Program napisany w języku C++ bez użycia gotowych bibliotek
- Dla tablicy i listy rozpatrzono przypadki dodawania/usuwania elementów z początku/końca/dowolnej pozycji
- Do generowanie liczb użyto maszyny losującej mt19937 (Mersenne twister, Matsumoto i Nishimura 1998)
- Pomiaru czasu dokonano za pomocą funkcji *QueryPerformanceCounter*
- Wyniki w tabelach uśrednione dla 100 pomiarów dla każdej ze struktur

Złożoność obliczeniowa

Definicja

Złożoność obliczeniowa jest definiowana jako określanie ilości zasobów potrzebnych do rozwiązania problemów obliczeniowych. Wyrażana za pomocą funkcji parametru, od której zależy jej wartość.

W zależności od rozważanego zasobu można wyróżnić dwa typy złożoności

1. Złożoność czasowa – wyrażana za pomocą zwykłych jednostek czasu, np. [ms]. Miarą złożoności czasowej jest liczba wykonanych operacji podstawowych (podstawienie, porównanie itd.) w zależności od rozmiaru danych wejściowych.
2. Złożoność pamięciowa – jest miarą ilości wykorzystanej pamięci. Wyraża się za pomocą funkcji użytych zasobów od rozmiaru danych wejściowych. Również możliwe obliczanie rozmiaru potrzebnych zasobów wyrażonych w bitach/bajtach.

Typy złożoności

- Złożoność optymistyczna – określa najkrótszy czas wykonania algorytmu dla najbardziej uporządkowanego zbioru danych wejściowych.
- Złożoność średnia – określa czas wykonania algorytmu dla losowego zbioru danych.
- Złożoność pesymistyczna – określa najdłuższy czas wykonanie algorytmu dla najbardziej nieuporządkowanego zbioru danych wejściowych.

Złożoności badanych struktur danych

Złożoność tablicy dynamicznej

Operacja	Średnia	Pesymistyczna
Dodanie elementu	O(n)	O(n)
Usunięcie elementu	O(n)	O(n)
Wyszukanie elementu	O(n)	O(n)

Złożoność listy dwukierunkowej

Operacja	Średnia	Pesymistyczna
Dodanie elementu	O(1)	O(n)
Usuwanie elementu	O(1)	O(n)
Wyszukanie elementu	O(n)	O(n)

Złożoność kopca binarnego typu MAX

Operacja	Średnia	Pesymistyczna
Dodanie elementu	O(1)	O(log(n))
Usuwanie szczytu	O(log(n))	O(log(n))
Wyszukanie elementu	O(n)	O(n)

Złożoność drzewa BST

Operacja	Średnia	Pesymistyczna
Dodanie elementu	O(log(n))	O(n)
Usuwanie	O(log(n))	O(n)
Wyszukanie elementu	O(log(n))	O(n)

Plan projektu

W projekcie zostały zaimplementowane wymienione wyżej struktury danych. Zgodnie z założeniami w projekcie zostały zaimplementowane funkcje dodania/usuwania z początku/końca/dowolnej pozycji. Oprócz tego wprowadzona została możliwość wczytania danych z pliku.

Dla testowania przyjęte następujące rozmiary struktur: 500, 1000, 2000, 5000, 10000, 15000, 20000, 25000, 30000, 35000, 40000

Struktury generowane następująco:

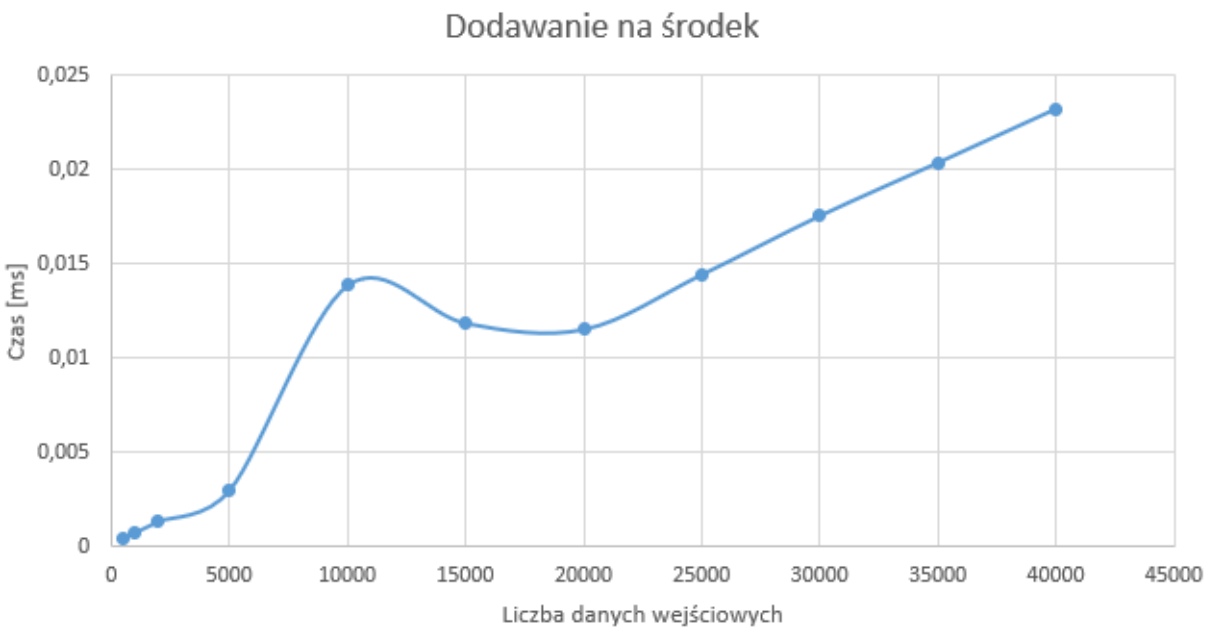
1. Dodanie losowych liczb dla struktury w pętli, warunkiem której były powyższe rozmiary
2. Użycie procedury czas której będzie mierzony
3. Po 100 krotnym wykonaniu funkcji był wyliczony średni czas

Wyniki pomiarów

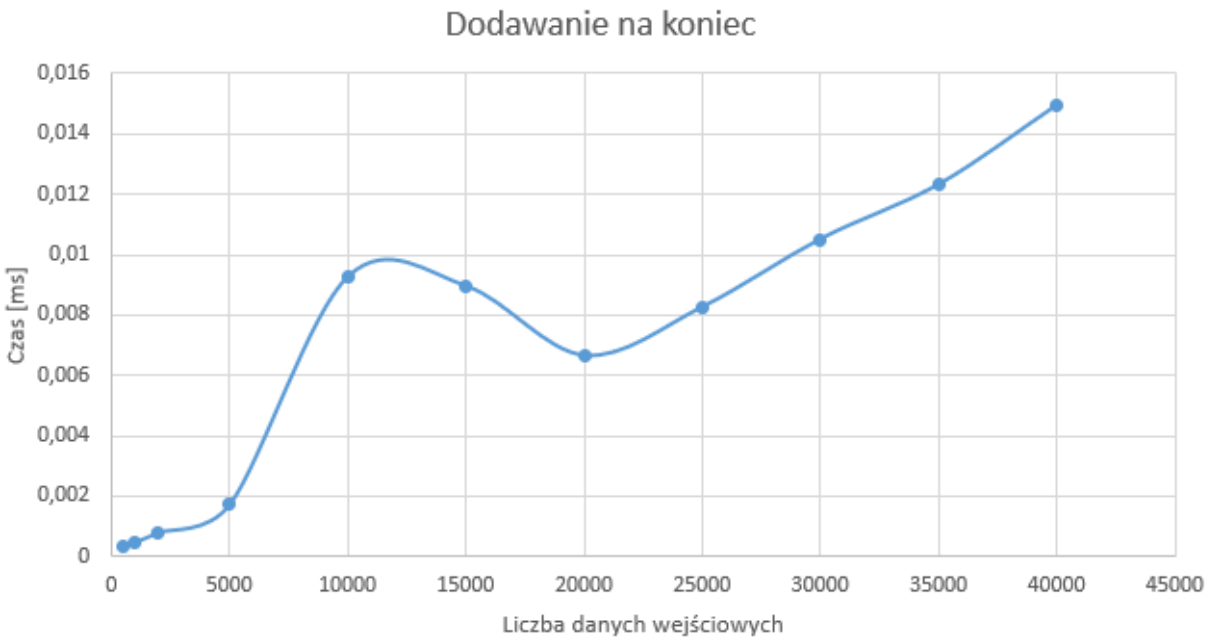
Tablica dynamiczna

	Liczba danych	Dodawanie na początek	Dodawanie w środek	Dodawanie na koniec	Usuwanie początku	Usuwanie środkowego	Usuwanie końca	Szukanie
L.p.	N	Ms	Ms	Ms	Ms	Ms	Ms	Ms
1	500	0,00037	0,00048	0,00034	0,00031	0,00056	0,00033	0,00014
2	1000	0,00053	0,00075	0,00045	0,00051	0,00067	0,00051	0,00024
3	2000	0,00093	0,00134	0,00077	0,00077	0,00115	0,00088	0,00046
4	5000	0,00200	0,00302	0,00172	0,00176	0,00266	0.00152	0,00105
5	10000	0,01562	0,01387	0,00927	0,00347	0,00928	0,00708	0,00216
6	15000	0,00933	0,01183	0,00893	0,00497	0,00956	0,00765	0,00321
7	20000	0,00739	0,01152	0,00664	0,00659	0,01336	0,00723	0,00429
8	25000	0,00938	0,01444	0,00826	0,00888	0,01256	0,00901	0,00535
9	30000	0,01121	0,01756	0,01051	0,00989	0,01509	0,01081	0,00644
10	35000	0,01303	0,02038	0,01232	0,01118	0,01772	0,01275	0,00748
11	40000	0,01495	0,02324	0,01495	0,01271	0,02032	0,01499	0,00858

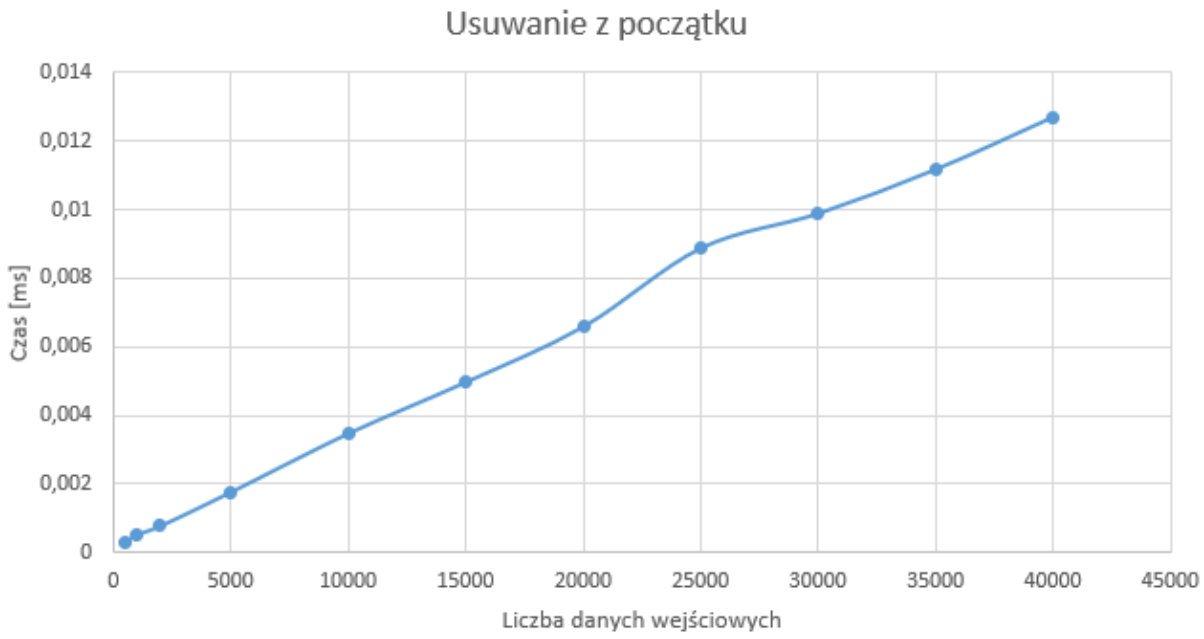
Rysunek 1. Dodawanie elementu na początek tablicy



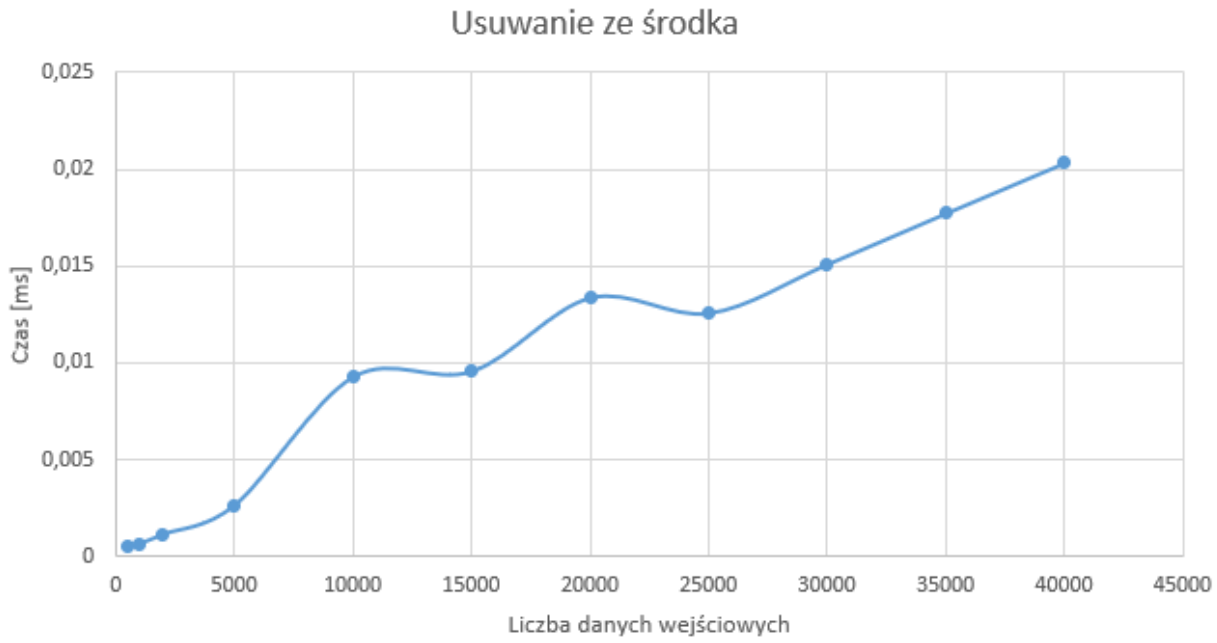
Rysunek 2. Dodawanie elementu w środek tablicy



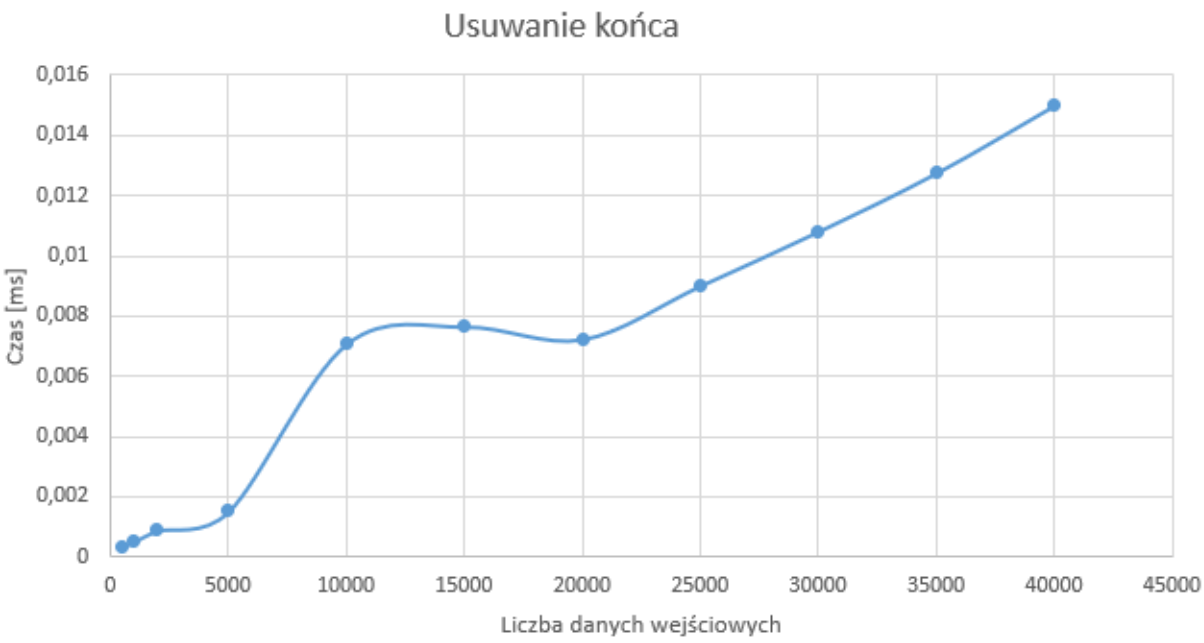
Rysunek 3. Dodawanie elementu na koniec tablicy



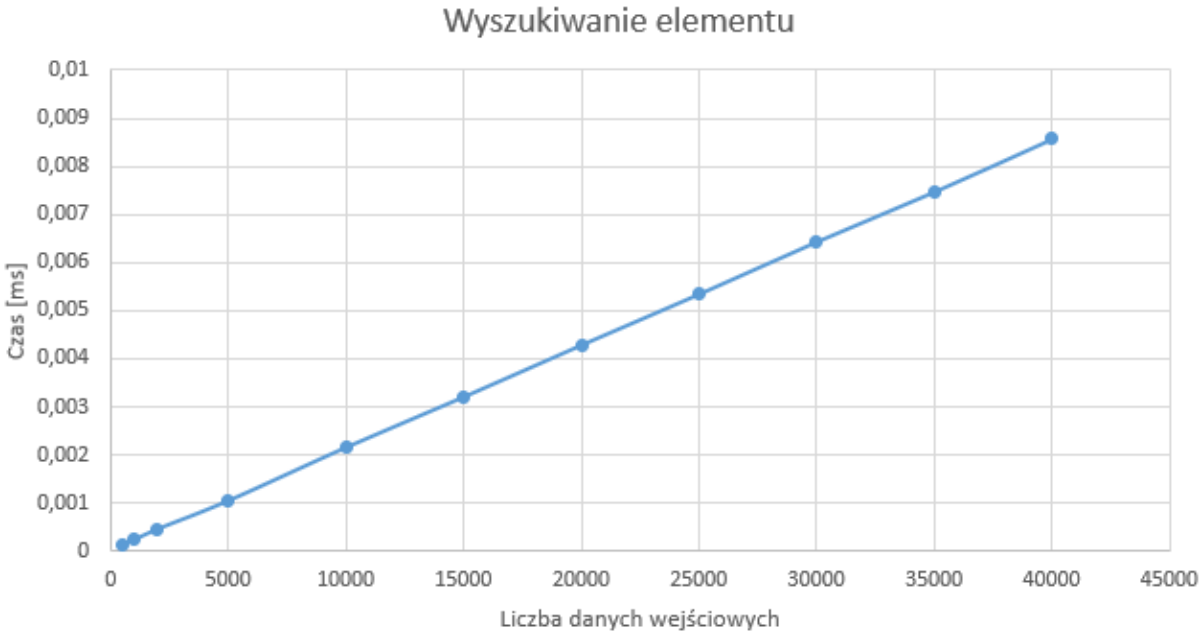
Rysunek 4. Usuwanie elementu z początku tablicy



Rysunek 5. Usuwanie elementu ze środka tablicy



Rysunek 6. Usuwanie elementu na końcu tablicy

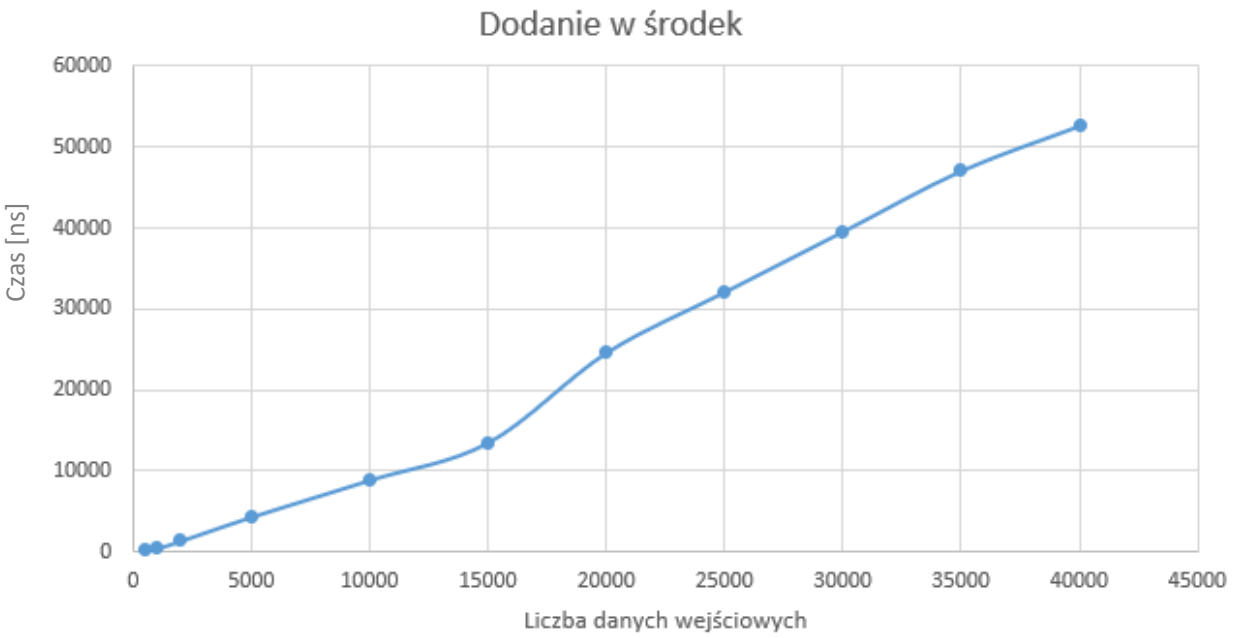


Rysunek 7. Wyszukiwanie losowego elementu w tablicy

Lista dwukierunkowa

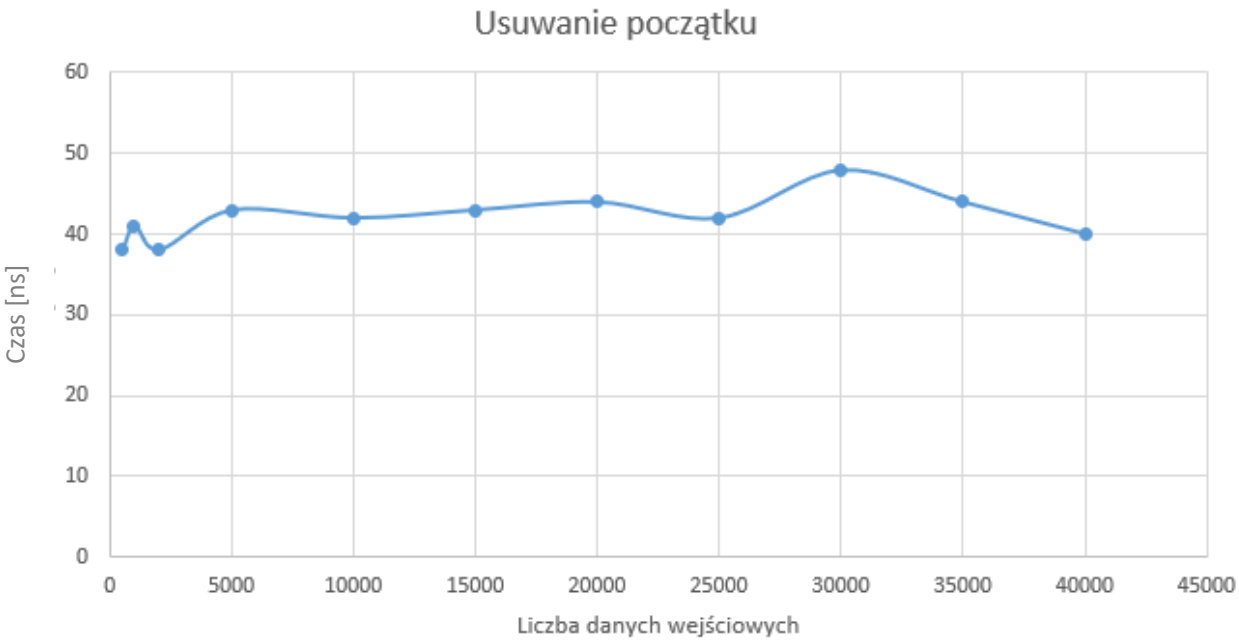
	Liczba danych	Dodawanie na początek	Dodawanie w środek	Dodawanie na koniec	Usuwanie początku	Usuwanie środkowego	Usuwanie końca	Szukanie
L.p.	N	Ns	Ns	Ns	Ns	Ns	Ns	Ns
1	500	50	257	32	38	279	74	474
2	1000	41	466	39	41	494	67	1458
3	2000	47	1412	40	38	1240	65	4406
4	5000	50	4376	34	43	4337	78	10391
5	10000	42	8932	32	42	8700	82	24012
6	15000	46	13572	44	43	17114	88	44138
7	20000	51	24682	39	44	25943	88	54772
8	25000	52	32175	33	42	33640	88	66921
9	30000	56	39639	43	48	40887	105	81863
10	35000	45	47138	43	44	47742	92	95773
11	40000	58	52692	42	40	53905	86	108572

Rysunek 2. Dodanie elementu na początek listy dwukierunkowej



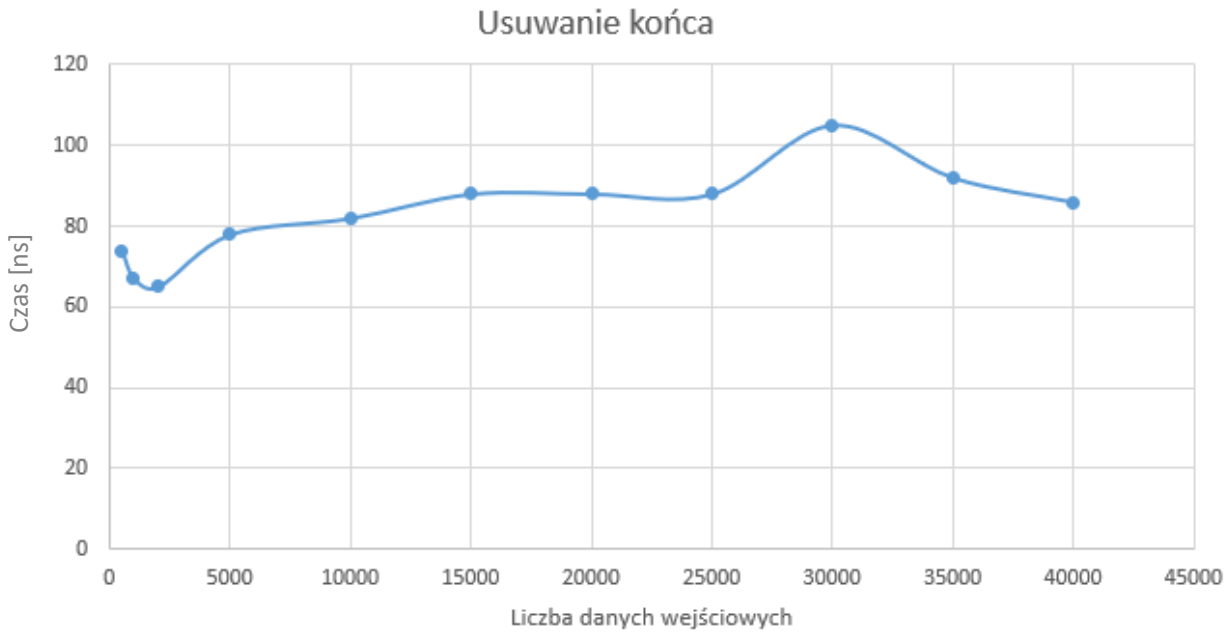
Rysunek 9. Dodanie elementu w środek listy dwukierunkowej

Rysunek 10. Dodanie elementu na koniec listy dwukierunkowej



Rysunek 3. Usuwanie elementu początkowego listy dwukierunkowej

Rysunek 4. Usuwanie elementu ze środka listy dwukierunkowej

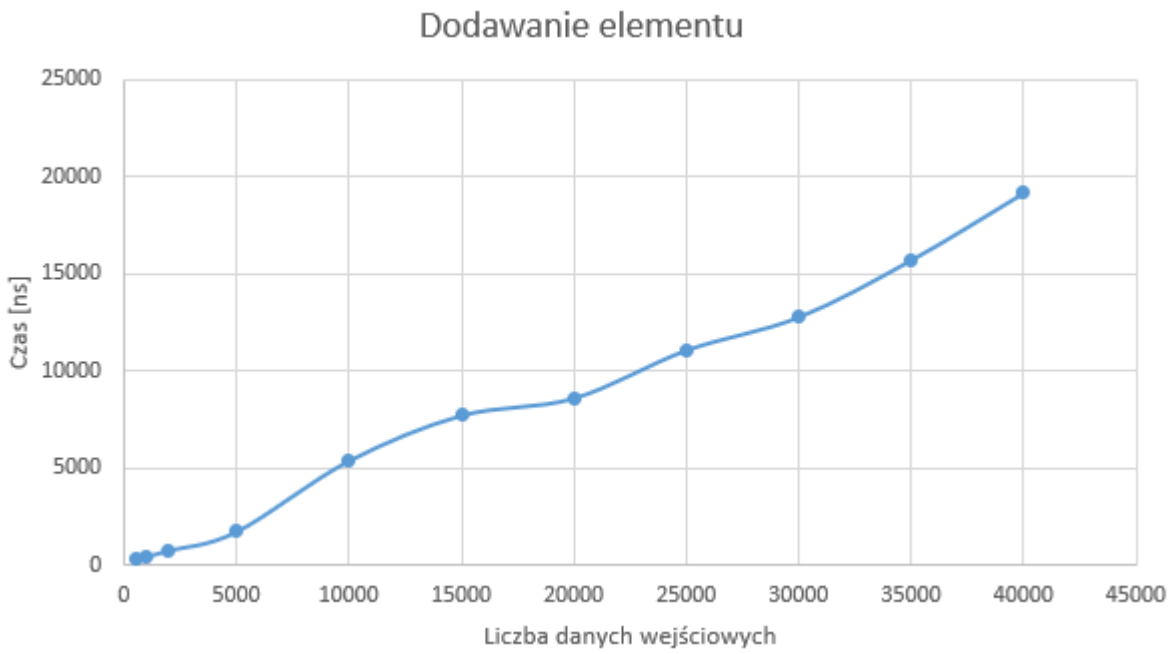


Rysunek 13. Usuwanie elementu z końca listy dwukierunkowej

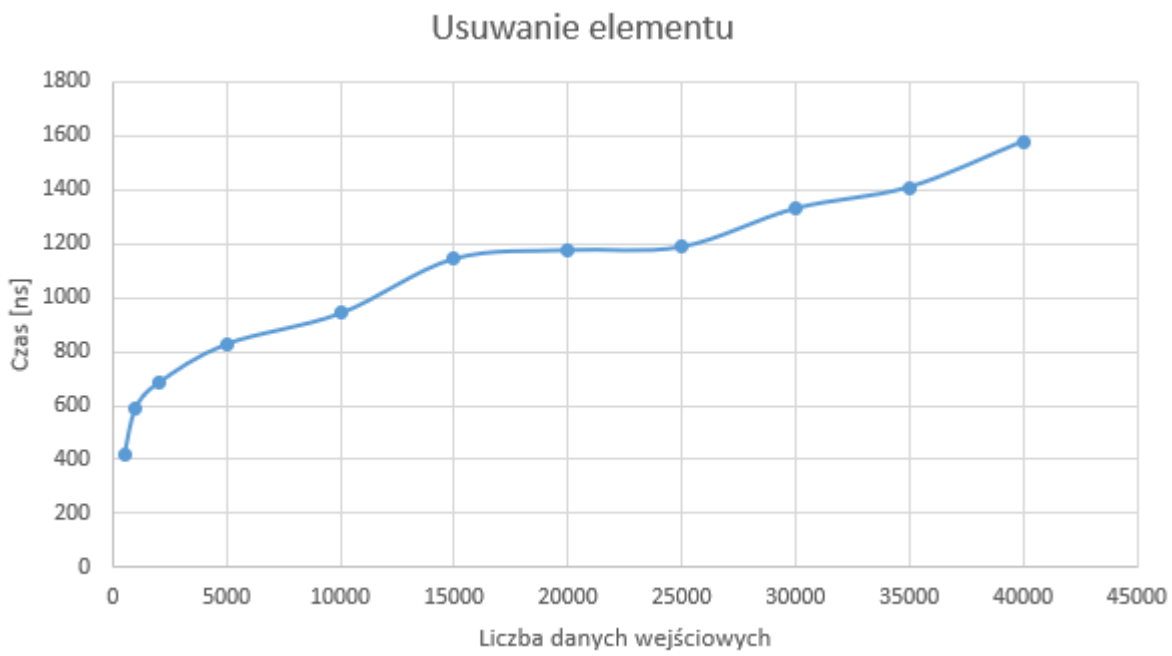
Rysunek 14. Wyszukiwanie losowego elementu w liście dwukierunkowej

Kopiec binarny typu MAX

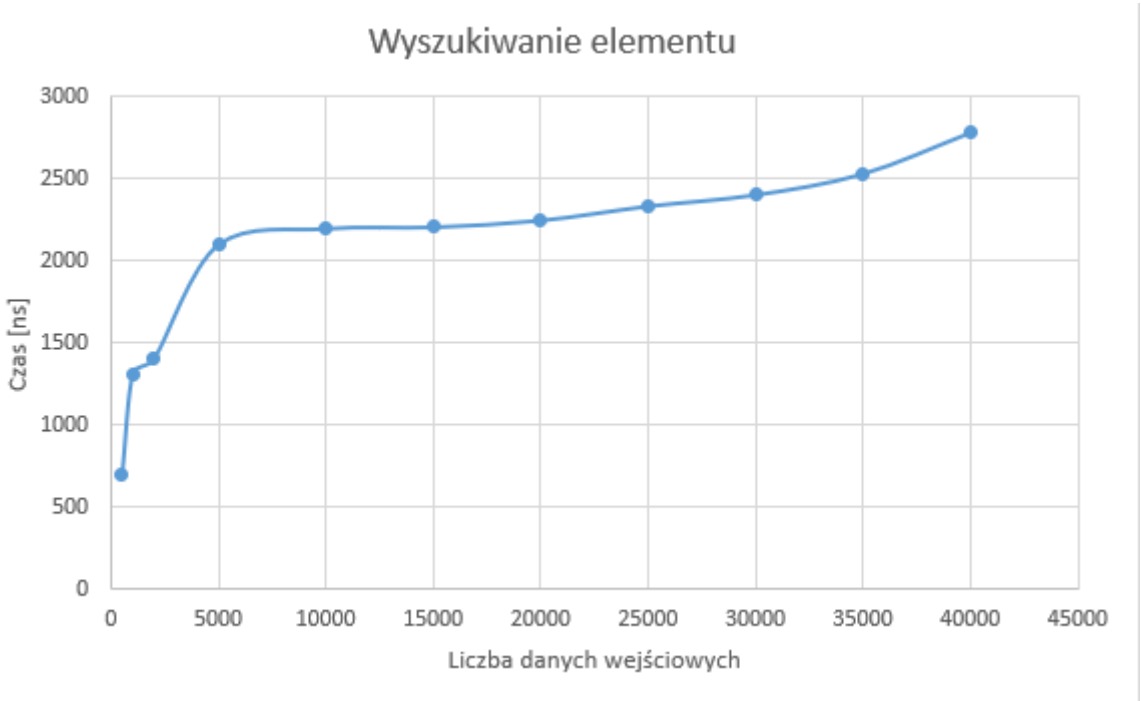
	Liczba danych	Dodawanie	Usuwanie	Szukanie
L.p.	N	Ns	Ns	Ns
1	500	339	417	701
2	1000	478	591	1312
3	2000	803	685	1407
4	5000	1758	829	2101
5	10000	5402	942	2198
6	15000	7737	1144	2207
7	20000	8625	1176	2248
8	25000	11090	1188	2335
9	30000	12792	1330	2404
10	35000	15721	1409	2532
11	40000	19172	1578	2783



Rysunek 15. Dodawanie elementu do kopca



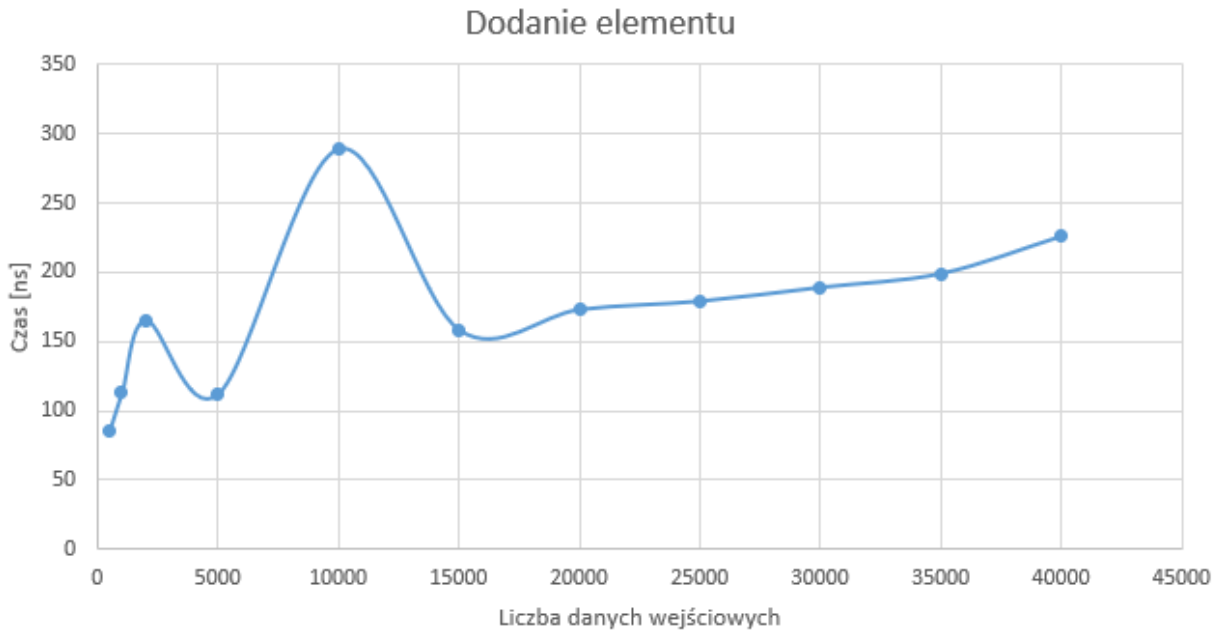
Rysunek 16. Usuwanie szczytu w kopcu



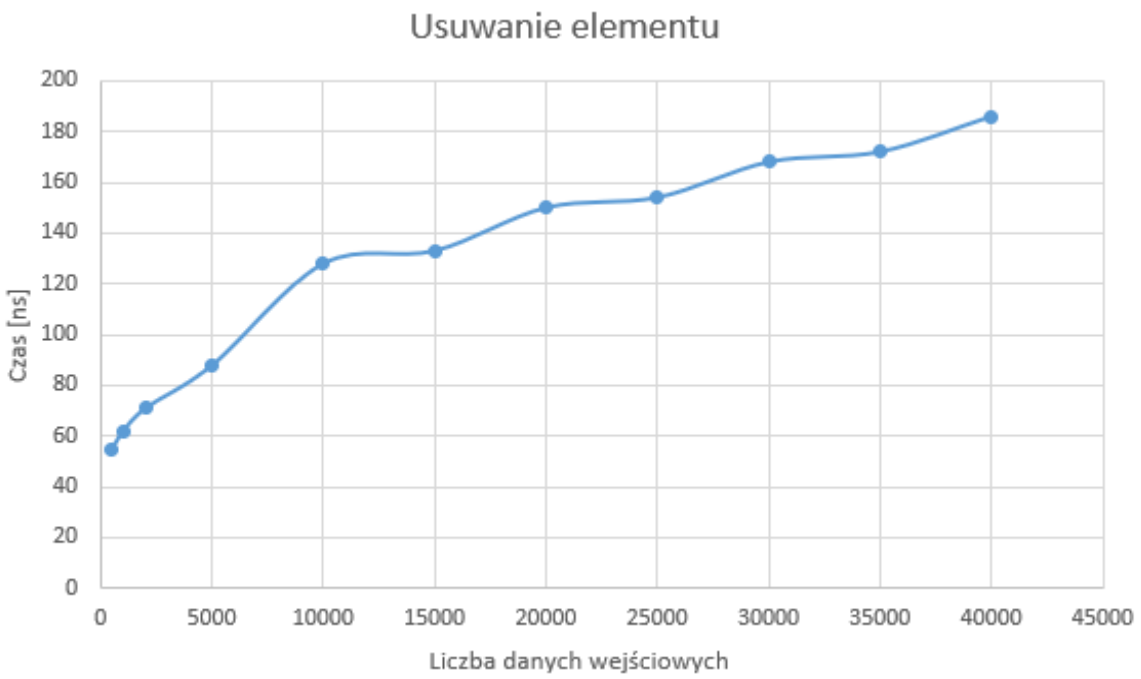
Rysunek 17. Wyszukiwanie losowego elementu w kopcu

Drzewo BST

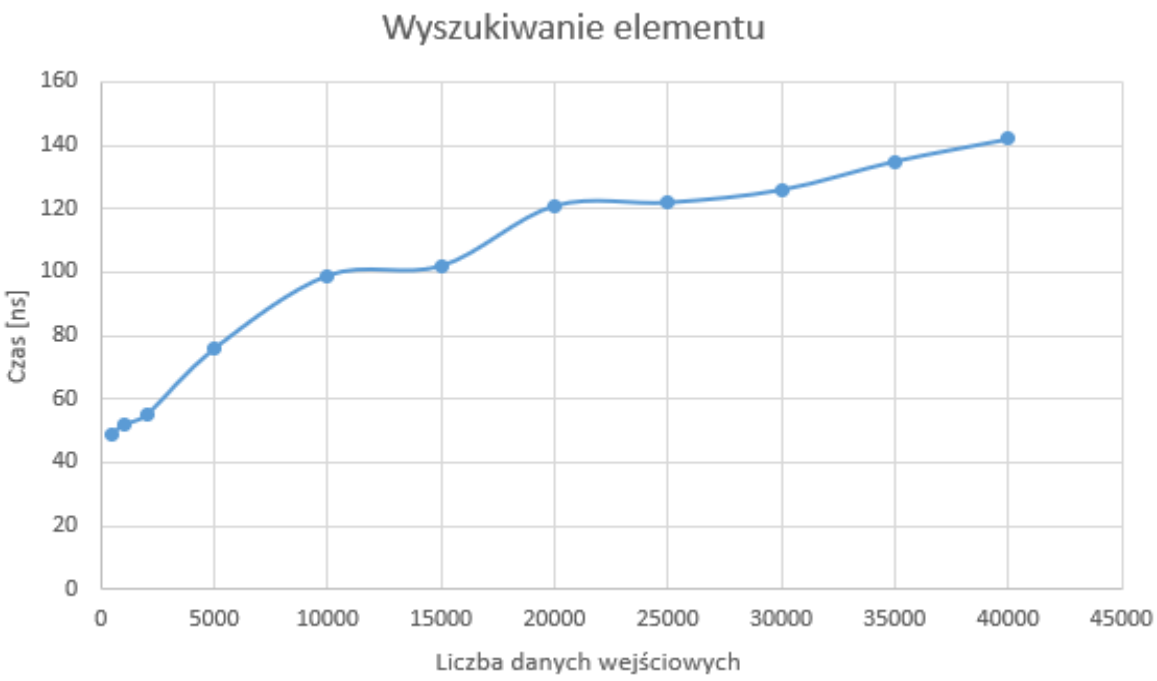
	Liczba danych	Dodawanie	Usuwanie	Szukanie
L.p.	N	Ns	Ns	Ns
1	500	86	55	49
2	1000	113	62	52
3	2000	165	71	55
4	5000	112	88	76
5	10000	289	128	99
6	15000	158	133	102
7	20000	173	150	121
8	25000	179	154	122
9	30000	189	168	126
10	35000	199	172	135
11	40000	226	186	142



Rysunek 6. Dodanie elementu do drzewa BST



Rysunek 8. Usuwanie elementu losowego w drzewie BST



Rysunek 7. Szukanie losowego elementu w drzewie BST

Wnioski

Eksperymenty na tablicy dynamicznej, rozmiar której zależy od ilości elementów, pokazują następne wyniki, że czas potrzebny do usunięcia, dodania oraz wyszukiwania elementu zależy od wielkości tej struktury. Wykresy pokazują prawie prostą linię zgodnie ze złożonością teoretyczną, jednak dla rozmiaru 10000 wartość czasu jest zbyt duża. Lista dwukierunkowa pokazuje wyniki zgodne z założeniami teoretycznymi, wyszukiwanie elementu oraz dodanie/usuwanie do środka ma złożoność zależnej od rozmiaru struktury. Dodawanie na początek/koniec zarówno jak i usuwanie elementu początkowego/końcowego na wykresie prawie stałe co jest zrozumiałe, bo mając wskaźniki na początek i koniec te operacje nie muszą przechodzić przez zawartość listy. Dla kopca binarnego czas dodawania/usuwania elementu zależy również od rozmiaru danej struktury co nie pasuje do teoretycznych złożoności możliwą przyczyną jest sposób implementowania kopca za pomocą tablicy. Eksperymenty na drzewie BST pokazują, że usunięcie elementu przypomina zależność logarytmiczną, czas dodanie elementów różni się dla liczby danych 10000 występuje zaburzenie, ale nie biorąc pod uwagę zaburzenie zależność podobna do logarytmicznej. Szukanie elementu zależy od rozmiaru struktury i również przypomina zależność logarytmiczną. Wyniki eksperymentalne dla drzewa w całości pasują do teoretycznych złożoności, osiągnięto to za pomocą operacji równoważenia drzewa.

Materiały

[1] https://www.bigocheatsheet.com/

[2] https://en.wikipedia.org/wiki/Binary_heap

[3] https://en.wikipedia.org/wiki/Binary_search_tree

[4] https://en.wikipedia.org/wiki/Doubly_linked_list

[5] https://pl.wikipedia.org/wiki/Algorytm_DSW