Tabela dla listy rodzaj insertion_time

wartość	czas dla false	czas dla true
1000	0.000	0.000
3000	0.000	0.000
9000	0.000	0.000
15000	0.000	0.000
25000	0.000	0.000
40000	0.000	0.000
50000	0.000	0.000
80000	0.000	0.015
100000	0.000	0.016

Tabela dla listy rodzaj search_time:

wartość	czas dla false	czas dla true
1000	0.000	0.000
3000	0.031	0.016
9000	0.156	0.156
15000	0.453	0.438
25000	1.265	1.219
40000	3.172	3.125
50000	4.973	4.906
80000	12.530	12.512
100000	20.561	19.762

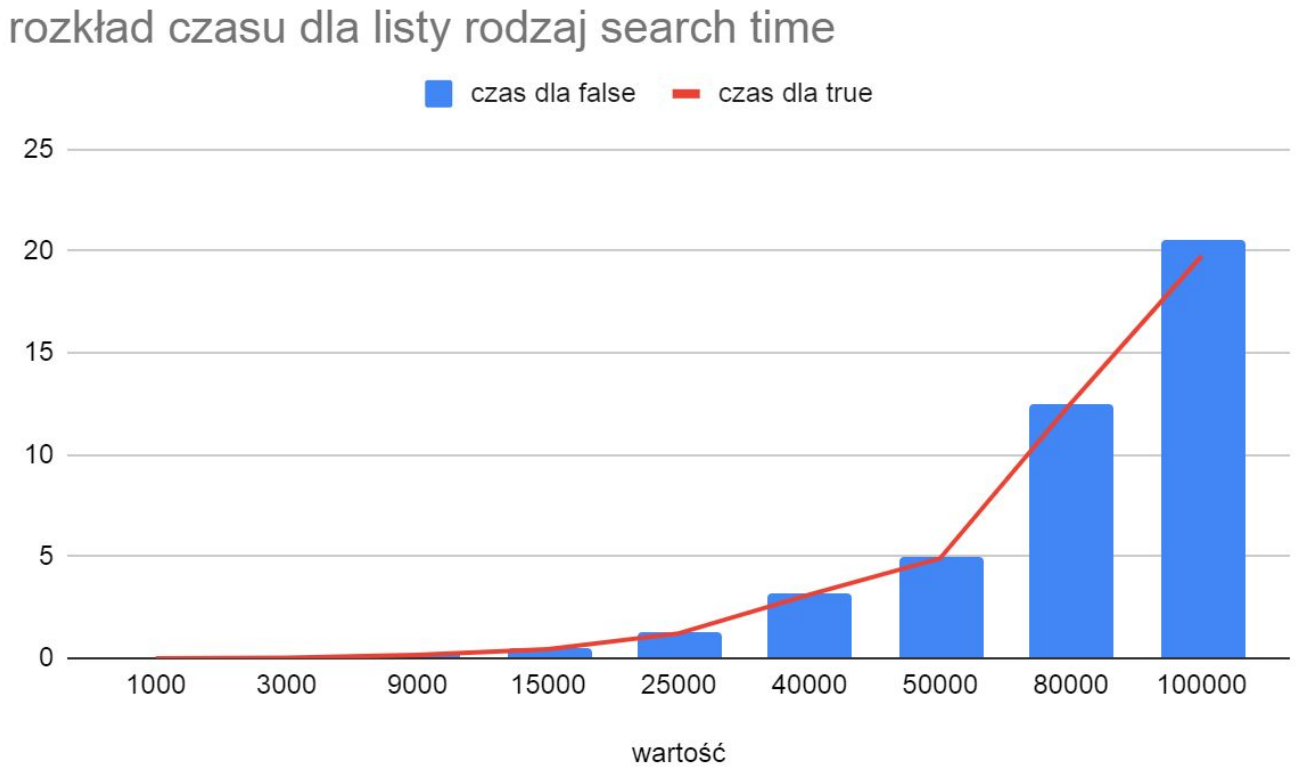

tabela dla drzewa insertion_time:

wartość	increasing	random	binary
1000	0.000	0.000	0.000
3000	0.031	0.000	0.000
5000	0.093	0.000	0.000
7500	0.203	0.000	0.015
10000	0.359	0.000	0.000
20000	1.375	0.000	0.000
30000	3.078	0.016	0.000
40000	5.859	0.015	0.000
50000	8.484	0.031	0.015
60000	17.129	0.031	0.016

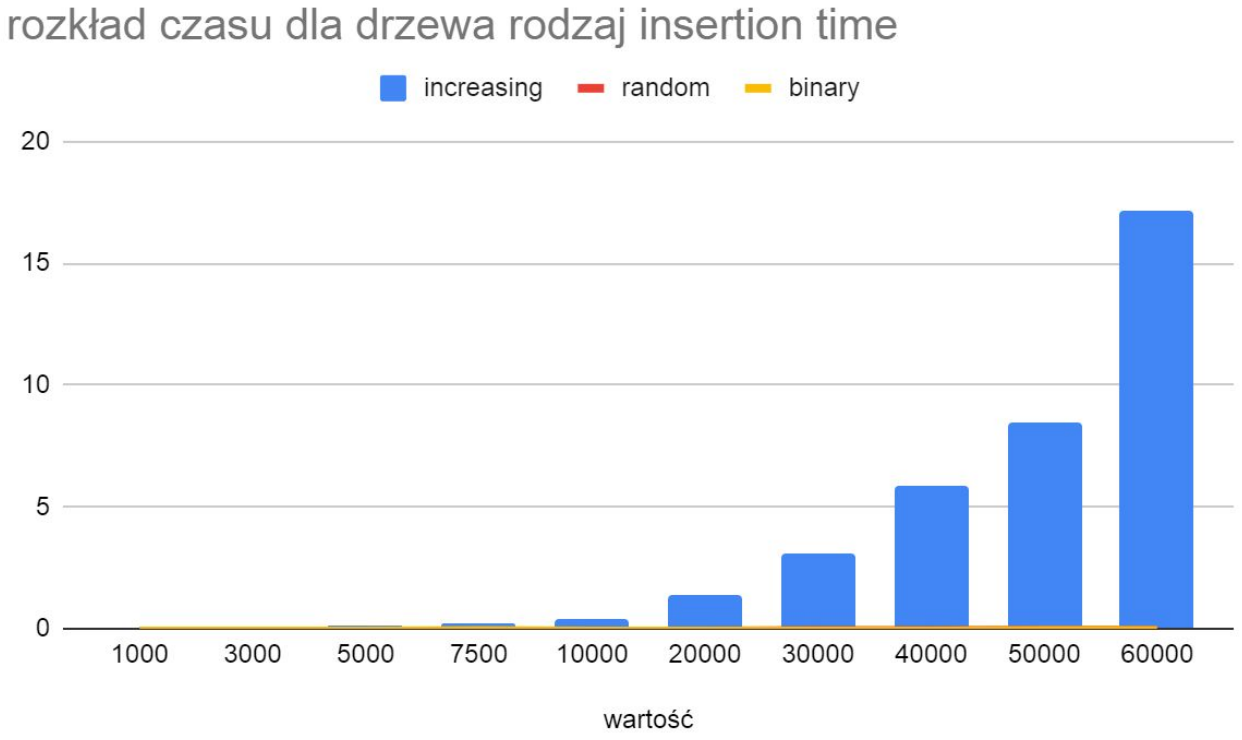

Tabela dla drzewa wartość serach_time:

wartość	increasing	random	binary
1000	0.000	0.000	0.000
3000	0.032	0.000	0.000
5000	0.079	0.000	0.000
7500	0.187	0.016	0.000
10000	0.328	0.016	0.000
20000	1.380	0.016	0.0016
30000	3.047	0.015	0.000
40000	5.359	0.016	0.015
50000	8.421	0.015	0.016
60000	13.840	0.031	0.015

Wnioski:

Jak widać z zamieszczonych danych czasy poszczególnych funkcji w drzewie i liście różnią się. Czas dodawania jest najkrótszy dla listy chociaż, drzewo binarne w tej kategorii osiąga zbliżony czas. Najgorszy czas dodawania uzyskamy przy użyciu drzewka increasing. Jeśli chodzi natomiast o czas szukania to sytuacja wygląda odwrotnie, drzewo wygrywa z lista, jednak najszybsza opcja nadal pozostaje rozwiazanie binarne dlatego stwierdzamy iż jest ono najlepsze spośród wszystkich zaproponowanych zarówno jeśli chodzi o czas dodawania i czas szukania.

Jan Kociński
Michał Szczepanowicz