Porównanie szybkości wyszukiwania elementów w haszu z uwzględnieniem rozmiaru hasza.

Potęgi liczby 2 jako rozmiar hasza

Lp.	Rozmiar hasza	Średni czas wyszukania 100 000 elementów [ms]
1	4 096	38.13
2	8 192	16.17
3	16 384	9.16
4	32 768	5.87
5	65 536	4.03
6	131 072	3.07
7	262 144	2.46

Liczby pierwsze jako rozmiar hasza

Lp.	Rozmiar hasza	Średni czas wyszukania 100 000 elementów [ms]
1	4 093	31.45
2	10 009	13.78
3	25 037	6.85
4	50 951	4.62
5	95 987	3.66
6	157 243	3.00
7	262 147	2.62

Wnioski:

1. Największy wpływ na czas wyszukiwania w haszu ma jego rozmiar. Jest to zrozumiałem, ponieważ żeby znaleźć wartość w krótszym haszu, należy przejść przez więcej elementów listy liniowej dla danego hashCode'u.

2. Najwyraźniejsze przeskoki widać do rozmiaru 65 536 dla potęg dwójki, przy tym rozmiarze współczynnik wypełnienia wynosi 1.53. Dalsze podajanie rozmiaru hasza nie skutkuje znaczącymi zyskami na wydajności. Podobnie sytuacja wygląda dla wartości 50 951 pośród liczb pierwszych, tutaj współczynnik wypełnienia wynosi 1,96.

3. Zastosowanie liczb pierwszych nie miało dużego wpływu na pomiar. Pomimo różnicy blisko 7 ms dla pierwszego pomiaru na korzyść liczb pierwszych, dla siódmego pomiaru widzimy różnicę na korzyść potęg dwójki, która zwarzywszy na bardzo niewielka różnicę rozmiarów, jest zauważalna. Z tego uważam, że różnice są spowodowane, jedynie niedokładnością pomiaru, a nie przewagą liczb pierwszych.

4. Pozostałe różnice między wartościami pomiarów, wynikają z tego, że dla liczb pierwszych na konkretnych pomiarach konsekwentnie był wybierany większy rozmiar hasza. Potwierdzać może to fakt, że po przekroczeniu wartości opisanych w punkcie 2. różnice stają się pomijalne i wynikają głownie z niedokładności pomiaru.

5. Uważam, ze najodpowiedniejszym rozmiarem hasza, z powyżej testowanych, dla tablicy 100 000 słów, jest rozmiar 65 536. Jest niewiele wolniejszy od większych haszy, jedocześnie zajmując od nich zdecydowanie mniej miejsca.