**HashMap**

Zalety HashMap w Javie:

1. **Szybkie Wyszukiwanie** (O(1)): Dzięki używaniu funkcji haszującej, HashMap oferuje stały czas dostępu (O(1)) do elementów, co oznacza, że czas potrzebny na odnalezienie elementu nie zależy od wielkości mapy.
2. **Elastyczność Rozmiaru**: HashMap automatycznie dostosowuje swój rozmiar w miarę wzrostu danych. Jest to przydatne, gdy nie wiemy dokładnie, ile elementów zostanie przechowywanych.
3. **Prostota Używania**: Interfejs HashMap jest prosty do zrozumienia i używania. Dodawanie, usuwanie i pobieranie elementów jest łatwe i intuicyjne.
4. **Możliwość Przechowywania Kluczy i Wartości o Różnych Typach**: HashMap może przechowywać pary klucz-wartość, gdzie klucze i wartości mogą być różnych typów, co zapewnia elastyczność w przechowywaniu danych.

Wady HashMap w Javie:

1. **Brak Gwarancji Kolejności:** Elementy w HashMap nie są przechowywane w kolejności wstawiania. Nie ma gwarancji, że iteracja po mapie zwróci elementy w określonej kolejności.
2. **Możliwość Kolizji:** Ze względu na zastosowanie funkcji haszującej, istnieje możliwość kolizji, gdy dwa różne klucze mają ten sam wynik funkcji haszującej. To może wpłynąć na wydajność, ponieważ wymaga rozwiązania konfliktów.
3. **Większe Użycie Pamięci**: HashMap może zużywać więcej pamięci niż inne struktury danych, szczególnie gdy jest dużo kolizji. Każda dodatkowa pozycja wewnętrzej tablicy wymaga dodatkowej pamięci.
4. **Niezgodność z Wątkami**: HashMap nie jest w pełni bezpieczna w środowisku wielowątkowym. Jeśli kilka wątków próbuje modyfikować mapę równocześnie, może to prowadzić do nieprzewidywalnych wyników lub błędów.

Mimo tych wad, HashMap jest często używanym narzędziem w Javie do szybkiego i łatwego mapowania kluczy na wartości. Wiele projektów korzysta z niej z powodzeniem do różnych zastosowań.