(ADT) Set Damian Kamiński 14.03.2021

W ramach zajęć należało zaimplementować typ abstrakcyjny Set. Wybrałem implementacje z wykorzystaniem wektora.

IsMember

Wykorzystując metodę isMember, przechodzę po wektorze w poszukiwaniu wybranego elementu korzystając z iteratora, od początku zbioru do jego końca. Moja złożoność wyniesie O(N) gdzie N to ilość elementów w zbiorze.

Insert

Średnia złożoność wyniesie O(N) gdzie N to ilość już obecnych elementów.
Na samym początku sprawdzam czy dodawany element nie znajduje się w zbiorze. Jeśli nie to dodaję element na koniec wektora.

Przy dodawaniu elementu do wektora wszystko przebiega sprawnie do momentu gdy ilość elementów przekroczy rozmiar wektora. Wtedy tworzony jest nowy wektor z większym maksymalnym rozmiarem a wszystkie elementy zostają przekopiowane. Średnia złożoność czasowa wynosi O(1) więc dodawania do wektora jest stałe. Wyjątkiem jest sytuacja wspomniana powyżej podczas której dodanie elementu wynosi O(arr.size())

Pop

Przy usuwaniu na samym początku sprawdzam czy podany element znajduje się w zbiorze. Wykorzystując metodę isMember sprawdzam czy element do usunięcia znajduje się w zbiorze, jeśli tak to pobieram iterator do tego elementu i usuwam go ze zbioru za pomocą metody vector::erase(). Złożoność czasowa vector::erase() wynosi O(N) (gdzie N to liczba elementów w wektorze), ponieważ po usunięciu elementu należy przesunąć jeszcze wszystkie dalsze elementy aby wypełnić powstałą dziurę. Złożoność czasowa isMember również wynosi O(N). Łącznie złożoność czasowa wynosi O(N).

Union

Suma zbiorów powstaje przez połączenie dwóch zbiorów, wykluczając duplikaty. Tworzę nowy zbiór do którego będę dodawał elementy z obydwu zbiorów. Najpierw dodaję elementy z 1 zbioru. Korzystam z metody Insert, której złożoność wynosi O(N). Jest ona w pętli, a więc łączna złożoność wynosi O(N * N1), bo dla każdego elementu ze zbioru 1 używam metody Insert. Podobnie ze zbiorem 2, złożoność wynosi O(N * N2). Łączna złożoność wynosi O(N(N1 + N2)).

Intersection

Przecięcie zbiorów, czyli elementy należące jednocześnie do obu zbiorów. W pętli sprawdzam czy elementy zbioru 2 należą do zbioru 1. Jeśli tak to dodaje je do nowego zbioru reprezentującego przecięcie zbiorów. Pętla wykona się N2 razy, w niej sprawdzenie warunku wykona się N1 razy (bo sprawdzam zbiór 1 za pomocą isMember). Łączna złożoność wyniesie O(N2 * N1 * N), gdzie N2, N1, N to liczba elementów w zbiorach.

Difference

Różnica zbioru, czyli elementy należące do zbioru 1 i nie należące do 2. Pętla wykona się N1 razy, warunek w każdym przejściu pętli wykona się N2 razy, a instrukcje w bloku if wykonają się w czasie O(N). Stąd łączna złożoność czasowa wyniesie O(N * N1 * N2).