Zbiory i iteratory

8 listopada 2018

1 Wypisywanka

Szablon pliku początkowego znajduje się w pliku zbiory_start.cpp.

We wszystkich poniższych zadaniach na wejściu dostajemy liczbę n oraz ciąg n liczb postaci.
$n \quad a_1 \quad a_2 \quad \ldots \quad a_n$
Przykładowy ciąg z pliku test.in

21 1 2 3 3 5 6 7 8 9 10 11 10 9 8 7 6 5 3 3 2 1

Od tego momentu uznajemy, że jeśli w zbiorze istnieją dwa takie same elementy to wykonujemy na nich operację tylko raz. Rozpatrujemy zbiór różnych elementów.
Na początku zobacz przykładowe użycie iteratorów. Nastepnie przejdź do rozwiązywania zadań.

1. Korzystając z funkcji insert. wczytaj liczby do zbioru.
2. Wypisz liczby w porządku rosnącym przy użyciu pętli for-range. Nie używaj funkcji begin() oraz end().
3. Korzystając z funkcji begin() i end().
   (a) Wypisz najmniejszą liczbę.
   (b) Wypisz 4 najmniejsze liczby. Używając pętli..
   (c) Wypisz największą liczbę. Przydatny może się okazać operator- -.
   (d) Wypisz liczby w porządku rosnącym korzystając z iteratorów.
4. Wypisz liczby korzystając z odwrotnych iteratorów oraz funkcji rbegin() oraz rend(), erase().
   (a) Wypisz 4 największe liczby.
   (b) Wypisz czwartą największą liczbę. Przydatna może się okazać funkcja advance() z $< algorithm >$
   (c) Usuń 4 największe liczby.

5. Wypisz rozmiar zbioru. size()
6. Sprawdź przynależność do zbioru liczb 3 oraz 8.
   (a) Funkcją count()
   (b) Funkcją find(). Zwróć uwagę na to co zwraca funkcja find(), gdy element nie należy do zbioru.
7. *Przetestuj działanie funkcji lower_bound() oraz upper_bound() dla liczb 3 i 4.
8. **Przejrzyj dokumentację, a następnie zmodyfikuj operator porównywania elementów w zbiorze, tak aby elementy większe były na początku.

2 Złożoność

Przeczytaj paragrafy o złożoności ('complexity') oraz ważności iteratorów ('iterator validity') wszystkich wymienionych powyżej funkcji.

3 Sortowanie

1. Stwórz prostą strukturę uczeń (string imie, string nazwisko).
2. Posortuj uczniow przy pomocy $< set >$ pisząc odpowiednią funkcję porównującą.
3. Posortuj uczniów przy pomocy vectora i funkcji sort.

Przy sortowaniu najistotniejsza jest kolejność nazwisk. Jeśli nazwiska są równe to interesuje nas kolejność imion.
Poniżej podane są przykładowy nieposortowany i posortowany ciąg uczniów.
(A, Nowak) (C, Kowalski) (B, Kowalski)
→ (B, Kowalski) (C, Kowalski) (A, Nowak)