

Opis:

Napisz efektywną aplikację działającą na kolejce priorytetowej zawierającej elementy typu Person, w których priorytety się nie powtarzają i zrealizowanej jako drzewo BST.

Przy czym węzły drzewa BST mają postać:

```

class Node {
    public Person info; // element danych (klucz)
    public Node left; // lewy potomek węzła
    public Node right; // prawy lewy potomek węzła
    // konstruktor...
} // koniec klasy Node
  
```

przy czym klasa Person ma postać:

```

class Person {
    public int priority; // priorytet osoby
    public String name;
    public String surname;
    // konstruktor...
}
// koniec klasy Person
  
```

Aplikacja powinna zawierać się trzy moduły: edycji, kolejkowania oraz raportowania i pracować w trybie konsolowym.

Moduły przedstawiają się następująco:

- **Moduł edycji:** powinien zawierać operacje:
 - CREATE order n $x_1 \dots x_n$ – rekurencyjnie tworzy drzewo BST na podstawie listy n elementów x_1 do x_n typu Person, podanych w porządku ich priorytetów, wyznaczonym przez drugi argument (PREORDER albo POSTORDER). Przy czym jeśli drzewo już istnieje, powinno zostać zastąpione nowym.
 - DELETE x – usuwa z drzewa osobę o priorytecie x . W przypadku, gdy w drzewie usuwany węzeł ma dwóch potomków, zamienia go z jego następnikiem.

- **Moduł kolejkowania:** powinien implementować podstawowe operacje na kolejce:
 - ENQUE x – dodaje osobę o priorytecie x do kolejki.
 - DEQUEMAX – zwraca oraz usuwa osobę o najwyższym priorytecie.
 - DEQUEMIN – zwraca oraz usuwa osobę o najniższym priorytecie.
 - NEXT x – zwraca najbliższą osobę o priorytecie większym od x .
 - PREV x – zwraca najbliższą osobę o priorytecie mniejszym od x .
- **Moduł raportowania:** powinien realizować następujące komendy:
 - PREORDER – wypisuje listę osób w porządku preorder.
 - INORDER – wypisuje listę osób w porządku inorder.
 - POSTORDER – wypisuje listę osób w porządku postorder.
 - HEIGHT – rekurencyjnie zwraca wysokość drzewa.

Wejście:

Dane do programu wczytywane są ze standardowego wejścia (klawiatury), zgodnie z poniższą specyfikacją:

Pierwsza linia zawiera liczbę całkowitą n ($1 \leq n \leq 100$) oznaczającą ilość testów.

- Pierwsza linia testu zawiera liczbę całkowitą m ($1 \leq m \leq 100$) oznaczającą ilość komend do wykonania na kolejce.
- W każdej następnej linii znajduje się jedna z wymienionych wyżej operacji i ewentualnie jej argument(y).
- W operacjach CREATE oraz ENQUE jako x przyjmuje się trójkę **<priorytet, imię, nazwisko>**. W operacjach DELETE, PREV oraz NEXT jako x podawany jest tylko priorytet.

Wyjście:

Każdy test powinien zaczynać się od wypisania ciągu „ZESTAW n ” (od 1). Dla każdej operacji w zestawie wypisz w jednej linii jej wynik zgodnie z podanymi przykładami.

Wymagania implementacyjne:

1. Jedynym możliwym importem jest skaner wczytywania z klawiatury.
2. Wszystkie komendy oprócz CREATE oraz HEIGHT muszą być zaimplementowane w wersji iteracyjnej.
3. Można założyć, że żadna operacja nie zostanie wywołana na kolejce przed wykonaniem operacji CREATE.

4. Złożoność wszystkich komend powinna być optymalna. W szczególności, żadna komenda nie powinna mieć złożoności większej niż liniowa.
5. Operacje PREV x oraz NEXT x nie mogą korzystać z listy INORDER.
6. Na końcu kodu przesyłanego submitu proszę dopisać w formie komentarza własne dane wejściowe.
7. Przypominam o komentowaniu aplikacji w formie opisanej w punkcie 3 Regulaminu zaliczania programów na BaCy z roku 2021/2022.

Test jawny:

test.in:	test.out:
1	ZESTAW 1
16	POSTORDER: 12 - Adam Nowak, 33 - Anna Kowalska, 30 - Marek Mickiewicz, 43 - Zofia Krzak, 37 - Ola Nowicka
CREATE POSTORDER 5 12 Adam Nowak 33 Anna Kowalska 30 Marek Mickiewicz 43 Zofia Krzak 37 Ola Nowicka	PREORDER: 37 - Ola Nowicka, 30 - Marek Mickiewicz, 12 - Adam Nowak, 33 - Anna Kowalska, 43 - Zofia Krzak
POSTORDER	INORDER: 12 - Adam Nowak, 30 - Marek Mickiewicz, 33 - Anna Kowalska, 37 - Ola Nowicka, 43 - Zofia Krzak
PREORDER	DEQUEMAX: 43 - Zofia Krzak
INORDER	DEQUEMIN: 12 - Adam Nowak
DEQUEMAX	INORDER: 30 - Marek Mickiewicz, 33 - Anna Kowalska, 37 - Ola Nowicka
DEQUEMIN	NEXT 33: 37 - Ola Nowicka
INORDER	NEXT 93: BRAK
NEXT 33	PREV 50: BRAK
NEXT 93	DELETE 50: BRAK
PREV 50	POSTORDER: 30 - Marek Mickiewicz, 37 - Ola Nowicka
DELETE 50	INORDER: 30 - Marek Mickiewicz, 35 - Andrzej Wolny, 37 - Ola Nowicka
DELETE 33	HEIGHT: 2
POSTORDER	
ENQUE 35 Andrzej Wolny	
INORDER	
HEIGHT	