PriorityQueue Class in Java

To process the objects in the queue based on the priority, we tend to use Priority Queue.

Important points about Priority Queue:

- PriorityQueue doesn't allow **null**
- We can't create PriorityQueue of Objects that are non-comparable
- The elements of the priority queue are ordered according to their natural ordering, or by a Comparator provided at queue construction time, depending on which constructor is used.
- The head of this queue is the least element with respect to the specified ordering. If multiple elements are tied for least value, the head is one of those elements — ties are broken arbitrarily.
- The queue retrieval operations poll, remove, peek, and element access the element at the head of the queue.
- It inherits methods from AbstractQueue, AbstractCollection, Collection and Object class.

Constructor: PriorityQueue()

This creates a PriorityQueue with the default initial capacity that orders its elements according to their natural ordering.

Methods:

1. booleanadd(E element): This method inserts the specified element into this priority queue.

2. public remove(): This method removes a single instance of the specified element from this queue, if it is present

3. public poll(): This method retrieves and removes the head of this queue, or returns null if this queue is empty.

4. public peek(): This method retrieves, but does not remove, the head of this queue, or returns null if this queue is empty.

5. iterator(): Returns an iterator over the elements in this queue.

6. booleancontains(Object o): This method returns true if this queue contains the specified element

Sample code Snippet to show usage of Priority Queue Class:

```
// Java progrm to demonstrate working of priority queue in Java
import java.util.*;

class Example
{
    public static void main(String args[])
    {
        // Creating empty priority queue
        PriorityQueue<String> pQueue =
                        new PriorityQueue<String>();

        // Adding items to the pQueue
        pQueue.add("C");
        pQueue.add("C++");
        pQueue.add("Java");
        pQueue.add("Python");

        // Printing the most priority element
        System.out.println("Head value using peek function:"
                                           + pQueue.peek());

        // Printing all elements
        System.out.println("The queue elements:");
        Iterator itr = pQueue.iterator();
        while (itr.hasNext())
            System.out.println(itr.next());

        // Removing the top priority element (or head) and
        // printing the modified pQueue
        pQueue.poll();
        System.out.println("After removing an element" +
                           "with poll function:");
        Iterator<String> itr2 = pQueue.iterator();
        while (itr2.hasNext())
            System.out.println(itr2.next());

        // Removing Java
        pQueue.remove("Java");
        System.out.println("after removing Java with" +
                           " remove function:");
        Iterator<String> itr3 = pQueue.iterator();
        while (itr3.hasNext())
            System.out.println(itr3.next());

        // Check if an element is present
        boolean b = pQueue.contains("C");
        System.out.println ( "Priority queue contains C" +
                             "ot not?: " + b);

        // get objects from the queue in an array and
        // print the array
        Object[] arr = pQueue.toArray();
        System.out.println ( "Value in array: ");
        for (int i = 0; i<arr.length; i++)
          System.out.println ( "Value: " + arr[i].toString()) ;
```

```
    }
}
```

Run on IDE

Output:

```
Head value using peek function:C
The queue elements:
C
C++
Java
Python
After removing an elementwith poll function:
C++
Python
Java
after removing Java with remove function:
C++
Python
Priority queue contains Cot not?: false
Value in array:
Value: C++
Value: Python
```

Applications :

Implementing Dijkstra's and Prim's algorithms.

Maximize array sum after K negations

This article is contributed by **Mehak Kumar.** Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

GATE CS Corner Company Wise Coding Practice

Java

Recommended Posts:

Deque interface in Java with Example
SortedMap Interface in Java with Examples
Queue Interface In Java
Java tricks for competitive programming (for Java 8)
Java.util.HashMap in Java
Java.lang.String.compareTo()
Templates in C++ vs Generics in Java
Converting Text to Speech in Java
Difference between x++ and x=x+1 in Java