



## Microprogetto “Coda con priorità di nodi”

### 1. Coda con Priorità

Completa la definizione della classe `NodeQueue` (in *code*) per sostituire `PriorityQueue<Node>` nel programma `HuffmanCompression`, offrendo le stesse funzionalità della classe predefinita (`PriorityQueue<Node>`) quando gli oggetti inseriti sono di tipo `Node`. Il protocollo deve quindi prevedere il costruttore e i metodi così specificati:

<code>public NodeQueue()</code>	<i>costruttore: creazione della coda di nodi vuota</i>
<code>public int size()</code>	<i>restituisce il numero di elementi contenuti nella coda</i>
<code>public Node peek()</code>	<i>restituisce l'elemento con “peso minore” (senza rimuoverlo dalla coda)</i>
<code>public Node poll()</code>	<i>restituisce e rimuove dalla coda l'elemento con “peso minore”</i>
<code>public void add( Node n )</code>	<i>aggiunge un nuovo elemento n alla coda</i>

Realizza la rappresentazione interna utilizzando strumenti base di Java, in particolare *array* di nodi (`Node`), pensando a dove si possono registrare i nodi inseriti nella coda e come si può identificare quello di peso minimo. In questo caso non serve definire un metodo `compareTo` perché si sa che per confrontare le priorità si devono confrontare i pesi (numeri di occorrenze) dei nodi.

### 2. Verifica

Verifica il buon funzionamento della soluzione proposta utilizzando il programma contenuta nella cartella *code*; in particolare, la versione di `HuffmanCompression` fa riferimento alla classe `NodeQueue` che avrai realizzato in accordo con i requisiti forniti sopra.