



Progetto “Codifica di Huffman”

26 Maggio 2021

1. Coda con Priorità

Definisci una classe `NodeQueue` che possa sostituire `PriorityQueue<Node>` nel programma `Huffman` offrendo le stesse funzionalità della classe predefinita 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 <code>n</code> alla coda</i> |

Realizza la rappresentazione interna utilizzando strumenti base di Java, in particolare gli *array*, senza ricorrere all'importazione delle classi rese disponibili dal package di supporto `java.util`.

2. Stack (pila)

Analogamente al punto precedente, definisci una classe `NodeStack` per sostituire `Stack<Node>` nei metodi che rielaborano iterativamente gli schemi ricorsivi utilizzati nei programmi di compressione e decompressione, garantendo le stesse funzionalità della classe predefinita. Il protocollo deve prevedere il costruttore e i metodi così specificati:

| | |
|---|---|
| <code>public NodeStack()</code> | <i>costruttore: creazione di uno stack di nodi vuoto</i> |
| <code>public boolean empty()</code> | <i>verifica se lo stack è vuoto</i> |
| <code>public Node peek()</code> | <i>restituisce l'elemento in cima allo stack (senza rimuoverlo dallo stack)</i> |
| <code>public Node pop()</code> | <i>restituisce l'elemento in cima allo stack e lo rimuove dallo stack</i> |
| <code>public void push(Node n)</code> | <i>aggiunge un nuovo elemento <code>n</code> in cima allo stack</i> |

Anche in questo caso, realizza la rappresentazione interna utilizzando strumenti base di Java, in particolare gli *array*, senza ricorrere all'importazione delle classi rese disponibili dal package di supporto `java.util`.

3. Verifica

Verifica infine la correttezza delle soluzioni utilizzando (senza modificarla!) la versione del programma `Huffman` collegata a questa parte del progetto nella sezione delle pagine del corso dedicata al Laboratorio, versione che fa riferimento alle classi `NodeQueue` e `NodeStack` che avrai realizzato in accordo con i requisiti forniti sopra.