Laboratorio di Programmazione a.a. 2023-24

Progetto "Codifica di Huffman" – Parte I

22 Maggio 2024

1. Codifica di Huffman di testi generati in modo "random"

Definisci un metodo statico in Java che, dato il nome (`String`) del file da creare, vi scrive un testo "random" composto da caratteri i cui codici ASCII sono prodotti in modo casuale, con distribuzione uniforme, nell'intervallo 0–127.

La scelta aleatoria dei caratteri può avvalersi della funzione predefinita della classe Java `Math`

```
public static double random()
```

che ad ogni invocazione restituisce un valore di tipo `double` casuale nell'intervallo [0.0, 1.0[; in particolare, puoi utlizzare l'espressione `(char) (128 * Math.random())` . Per l'accesso e le operazioni sui file di testo fai riferimento alle classi del package `huffman_toolkit` disponibile attraverso le pagine del corso.

Applica il programma di compressione basato sulla tecnica di Huffman al testo "random" creato, e confronta la dimensione del file prodotto con quella del file originale. Cerca quindi di spiegare il risultato ottenuto.

2. Realizzazione di una "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 pertanto prevedere il costruttore e i metodi così specificati:

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

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

Verifica infine la correttezza della tua soluzione dopo aver sostituito tutte le occorrenze di `PriorityQueue<Node>` con `NodeQueue` nel programma `Huffman` (senza modificare altre parti del codice!).[1]

1 A seconda di come alcuni dettagli della realizzazione della coda con priorità vengono gestiti (in particolare nei casi in cui diversi elementi hanno lo stesso peso), i codici di Huffman associati ai caratteri possono risultare diversi, ma ciò non dovrebbe incidere sulla dimensione del file prodotto.