Esame 20250730

Esercizio lode

(1) **Esercizio Lode**

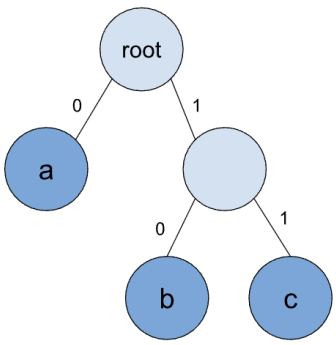
ESSAY | marked out of 1 | penalty 0 | File picker

In informatica, l'algoritmo di Huffman é una tecnica di compressione dati largamente utilizzata per ridurre la quantitá di spazio necessaria a memorizzare o trasmettere informazioni. La sua idea fondamentale é quella di rappresentare i simboli (come lettere, numeri o qualsiasi unitá di informazione) attraverso codici binari di lunghezza variabile, in modo tale che i simboli che compaiono piú frequentemente vengano codificati con sequenze di bit piú corte, mentre quelli meno frequenti con sequenze piú lunghe.

La codifica di Huffman si basa sulla costruzione di un albero binario speciale, detto albero di Huffman. In questo albero, ogni foglia corrisponde a un simbolo del messaggio e porta con sè la frequenza di quel simbolo (ossia quante volte esso appare nel testo o nei dati da codificare). L'albero viene costruito seguendo una procedura iterativa: partendo da una foresta di nodi singoli (ognuno rappresentante un simbolo con la sua frequenza), si uniscono due nodi con le frequenze piú basse creando un nuovo nodo interno, la cui frequenza é la somma di quelle due, e si ripete questo processo fino ad avere un solo albero.

Il percorso dalla radice di questo albero fino a ciascuna foglia definisce il codice binario del simbolo associato. Per convenzione, scendere lungo un ramo sinistro corrisponde all'aggiunta di un bit 0 al codice, mentre scendere lungo un ramo destro corrisponde all'aggiunta di un bit 1. Di conseguenza, i simboli piú frequenti, essendo piú vicini alla radice, avranno codici piú corti, mentre quelli meno frequenti, piú lontani dalla radice, avranno codici piú lunghi. Questa struttura assicura che nessun codice sia prefisso di un altro, cioé che la codifica sia univocamente decodificabile senza ambiguitá e ció consente di ricostruire esattamente i dati originali a partire dalla versione compressa. L'algoritmo di Huffman é quindi estremamente efficiente e trova applicazioni in molti ambiti.

Per esempio, dato l'albero di Huffman:



I codici associati saranno:

a $\rightarrow$ "0"
b $\rightarrow$ "10"
c $\rightarrow$ "11"

Completare il programma `lode.cpp` inserendo la dichiarazione e la definizione della funzione ricorsiva:

```
void CodificaParola(NodoHuffman* radice, const char* parola,
cha* codiceRisultato);
```

dove:

- `NodoHuffman` é una struttura dati giá definita che rappresenta un nodo dell albero di Huffman, con i campi: char carattere; (il simbolo, oppure un carattere speciale per i nodi interni) NodoHuffman sinistro; NodoHuffman destro;
- `radice` é il puntatore alla radice dell'albero di Huffman.
- `parola` é la stringa di caratteri da codificare, terminata da `\0`.
- `codiceRisultato` é un array di `char`, non inizializzato, di dimensione massima 100 caratteri (incluso terminatore), in cui salvare la parola codificata in forma binaria (sequenza di 0 e 1), terminata da `\0`.

La funzione deve:

- Per ogni carattere della parola, cercare il codice binario corrispondente nell'albero di Huffman.
- Concatenare i codici binari trovati in codiceRisultato, assicurandosi che non si eccede la dimensione massima di 100 caratteri (incluso il terminatore).
- Terminare la stringa con il carattere `\0`.

Non é ammesso l'uso di oggetti di tipo `std::string` o altre librerie di manipolazione stringhe, pena annullamento dell'esercizio.

Si deve utilizzare un approccio ricorsivo per la ricerca del codice di ciascun carattere nell'albero. Per la copia della stringa risultante si possono usare iterazioni (e.g., for, while) o ricorsione, ma non si possono usare funzioni di libreria come `strcpy` o simili, pena annullamento dell'esercizio.

Non é consentito usare variabili globali o statiche, pena annullamento dell'esercizio.

Si puó usare solo la libreria standard `iostream` per input/output (pena annullamento dell'esercizio).

Si assuma che l'albero di Huffman sia giá costruito e fornito, non é richiesto implementare la sua costruzione.

La funzione `CodificaParola` assume solo che il risultato della stringa da condificare stia in un array di 100 caratteri (incluso terminatore), ma deve poter funzionare per qualsiasi albero di Huffman e qualsiasi parola da codificare (se la stringa eccede), si copia fino a completare con terminatore l'array di 100 caratteri.

Esempio: Se l'albero ha codici:
a $\rightarrow$ "0"
b $\rightarrow$ "10"
c $\rightarrow$ "11"
e la parola da codificare é "abc", allora la funzione deve stampare: 01011

Scaricare il file `lode.cpp`, modificare solo inserendo la definizione della funzione CodificaParola e delle eventuali funzioni ausiliarie necessarie, compilare e caricare il file risultante.

Note importanti.

- Scaricare i file `lode.cpp`.
- Modificare solo il file `lode.cpp`.

- Caricare il solo file `lode.cpp` per la valutazione.

lode.cpp

Information for graders:

Total of marks: 1