

## Esame 20250909

### Esercizio lode

#### (1) Esercizio Lode

L'algoritmo di Huffman é una tecnica di compressione che utilizza un albero binario per rappresentare i simboli con codici binari di lunghezza variabile.

Una caratteristica fondamentale di questa codifica é che é priva di ambiguitá: nessun codice di un simbolo puó essere prefisso del codice di un altro simbolo.

Questo garantisce che la sequenza di bit ottenuta dalla codifica di un testo possa essere decodificata in modo univoco.

Grazie a questa proprietá, il procedimento di decodifica é semplice e sicuro:

Si parte dalla radice dell'albero di Huffman.

Si leggono i bit della stringa codificata uno alla volta.

Se il bit é 0 → ci si sposta sul figlio sinistro.

Se il bit é 1 → ci si sposta sul figlio destro.

Quando si raggiunge una foglia (cioé un nodo che rappresenta un carattere), significa che é stato individuato un simbolo della parola originale.

Si scrive questo carattere nella parola decodificata e si riparte dalla radice per continuare a leggere i bit rimanenti. Alla fine, la stringa decodificata é ricostruita senza errori nè ambiguitá.

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

```
void DecodificaParola(NodoHuffman* radice, const char* codiceBinario, char* parolaRisultato);
```

dove:

NodoHuffman é una struttura dati gi 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.

codiceBinario é un array di char '0' e '1' terminata da \0, che rappresenta il testo codificato. parolaRisultato é un array di char in cui salvare la parola decodificata, terminata da \0.

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

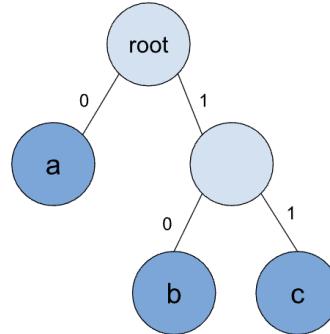
Si deve utilizzare un approccio ricorsivo per la ricerca del codice di ciascun carattere nell'albero. Per la copia della stringa risultante si possono usare iterazioni (e.g., for, while) o ricorsione, ma non si possono usare funzioni di libreria come strcpy o simili, pena annullamento dell'esercizio.

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

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

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

La funzione CodificaParola assume solo che il risultato della stringa da codificare 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.



Ad esempio, dato l'albero di Huffman:

a → "0"

b → "10"

c → "11"

e la parola da decodificare è "01011", allora la funzione deve stampare: abc

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*