Esame 20250730

Esercizio 2

(1) **Esercizio 2 v1**

ESSAY | marked out of 10 | penalty 0 | File picker

In molti casi pratici ci si trova a dover trasformare del testo perché questo possa essere visualizzato su dispositivi con al massimo un numero finito di caratteri. Inoltre, a volte per rendere il testo meno leggibile si utilizzano codifiche che trasformano il testo cambiando i caratteri.

Si assuma data una mappa che associa ad ogni lettera dell'alfabeto indipendentemente dal maiuscolo/minuscolo una altra lettera (e.g., 'A' -> 'Z', 'B' -> 'Y', ..., 'Z' -> 'A'), si vuole scrivere un programma che, legga un testo, memorizzi il contenuto del testo in una lista concatenata di `char`, e poi elabori il testo memorizzato nella lista concatenata stampando a video il carattere alfabetico memorizzato in ogni nodo della lista concatenata trasformato secondo la mappa data. Se il carattere memorizzato non é una lettera dell'alfabeto allora la stampa cosí come é.

Assumendo che la mappa dei caratteri sia memorizzata in un array di elementi di dimensione pari a 'Z'-'A'+1, dove ogni elemento in posizione i contiene il carattere in cui deve essere mappato (Non importa quale sia l'encoding di conversione utilizzato, che é opaco e definito solo dalla mappa, e non importa se il carattere é maiuscolo o meno, la mappa é definita solo sui caratteri maiuscoli). Si scriva una funzione ricorsiva `elabora` che prende come primo argomento la lista concatenata (`Testo`), come secondo argomento un array di caratteri che rappresenta la mappa dei caratteri, e come terzo argomento un intero che rappresenta la dimensione di ogni riga. La funzione `elabora` deve stampare a video il testo trasformato secondo la mappa data, e quando il numero di caratteri stampati eccede la dimensione della riga specificata andare a capo riga e continuare con il resto del testo seguendo lo stesso schema.

I caratteri del testo devono essere stampati nell'ordine in cui sono stati inseriti nella lista concatenata, e devono essere trasformati secondo la mappa data. Non si deve creare una copia della lista concatenata, ma si deve utilizzare solo la lista concatenata originale. Non sono consentiti usi di costanti intere per rappresentare i caratteri, ma si deve utilizzare solo l'aritmetica dei caratteri vista a lezione (pena annullamento dell'elaborato).

Il programma in `esercizio2.cpp` prende come argomento a) una stringa che rappresenta il nome di un file da codificare, e la dimensione della riga (un intero positivo).

Il `main` del programma é già implementato e non deve essere modificato, e chiama la funzione `elabora` **(da definire)** che prende gli argomenti sopra specificati. **La funzione `elabora` stampa a video la decodifica della stringa rispetto alla mappa data**.

La funzione `elabora` **deve essere ricorsiva** e **NON deve contenere iteratori** espliciti (`for`, `while`, `do-while`). La funzione `elabora` può ovviamente contenere codice sequenziale o condizionale. Sono consentite (se ritenute necessarie) chiamate a funzioni ricorsive ausiliarie che a loro volta **non contengano iterazioni esplicite** (`for`, `while`, `do-while`).

Il file `esercizio2.cpp` contiene tutto quanto necessario tranne la dichiarazione e la definizione della procedura `elabora`.

Di seguito sono riprotati due esempi di esecuzione del programma.

```
computer > ./a.out vuoto.txt 30

computer > ./a.out testo.txt 30
MVO NVAAL WVO XZNNRM WR MLHGIZ
ERGZ NR IRGILEZR KVI FMZ HVOEZ
LHXFIZ: XSV' OZ WRIRGGZ ERZ VIZ
 HNZIIRGZ.
V JFZMGL Z WRI JFZO
```

```
VIZ V' XLHZ WFIZ VHGZ HVOEZ HVO
EZTTRZ VG ZHKIZ V ULIGV, XSV MV
O KVMHRVI IRMLEZ OZ KZFIZ!
GZMG
'V' ZNZIZ XSV KLXL V' KRF' NLIG
V; NZ, KVI GIZGGZI WVO YVM XS'R
L ER GILEZR, WRIL' WVOO'ZOGIV X
LHV XS'RL E'L' HXLIGV.
```

Note:

- Scaricare il file `esercizio2.cpp`, modificarlo per inserire il codice necessario per rispondere a questo esercizio. **Caricare il file sorgente risultato delle vostre modifiche a soluzione di questo esercizio** nello spazio apposito.
- Si può assumere che la stringa di input contenga solo caratteri numerici e spazi.
- All'interno di questo programma **non è ammesso** l'utilizzo di variabili globali o di tipo `static` e di funzioni di libreria al di fuori di quelle definite in `iostream`, `fstream`, `cstdlib`, `cstdio`.
- Si ricorda che, gli esempi di esecuzione sono puramente indicativi, e la soluzione proposta NON deve funzionare solo per l'input fornito, ma deve essere robusta a variazioni compatibili con la specifica riportata in questo testo.
- Si ricorda di inserire solo nuovo codice e di **NON MODIFICARE** il resto del programma (pena annullamento dell'esercizio).
- Si ricorda che la soluzione deve essere implementata in C++ **NON usando `string` e altri elementi della C++ standard template library, anche se il file compila senza cambiare gli header**!

esercizio2.cpp

testo.txt

vuoto.txt

Information for graders:

(2) **Esercizio 2 v2**

ESSAY | marked out of 10 | penalty 0 | File picker

In molti casi pratici ci si trova a dover trasformare del testo perché questo possa essere visualizzato su dispositivi con al massimo un numero finito di caratteri. Inoltre, a volte per rendere il testo meno leggibile si utilizzano codifiche che trasformano il testo cambiando i caratteri.

Si assuma data una mappa che associa ad ogni lettera dell'alfabeto indipendentemente dal maiuscolo/minuscolo una altra lettera (e.g., 'A' -> 'Z', 'B' -> 'Y', ..., 'Z' -> 'A'), si vuole scrivere un programma che, legga un testo, memorizzi il contenuto del testo in una lista concatenata di `char`, e poi elabori il testo memorizzato nella lista concatenata stampando a video il carattere alfabetico memorizzato in ogni nodo della lista concatenata trasformato secondo la mappa data. Se il carattere memorizzato non é una lettera dell'alfabeto allora la stampa cosí come é.

Assumendo che la mappa dei caratteri sia memorizzata in un array di elementi di dimensione pari a 'Z'-'A'+1, dove ogni elemento in posizione i contiene il carattere in cui deve essere mappato (Non importa quale sia l'encoding di conversione utilizzato, che é opaco e definito solo dalla mappa, e non importa se il carattere é maiuscolo o meno, la mappa é definita solo sui caratteri maiuscoli). Si scriva una funzione ricorsiva `elabora` che prende come primo argomento la lista concatenata (`Testo`), come secondo argomento un array di caratteri che rappresenta la mappa dei caratteri, e come terzo argomento un intero che rappresenta la dimensione di ogni riga. La funzione `elabora` deve stampare a video il testo trasformato secondo la mappa data, e quando il numero di caratteri stampati eccede la dimensione della riga specificata andare a capo riga e continuare con il resto del testo seguendo lo stesso schema.

I caratteri del testo devono essere stampati nell'ordine inverso all'ordine in cui sono stati inseriti nella lista concatenata, e devono essere trasformati secondo la mappa data. Non si deve creare una copia della lista concatenata, ma si deve utilizzare solo la lista concatenata originale. Non sono consentiti usi di costanti intere per rappresentare i caratteri, ma si deve utilizzare solo l'aritmetica dei caratteri vista a lezione (pena annullamento dell'elaborato).

Il programma in `esercizio2.cpp` prende come argomento a) una stringa che rappresenta il nome di un file da codificare, e la dimensione della riga (un intero positivo).

Il `main` del programma é già implementato e non deve essere modificato, e chiama la funzione `elabora` **(da definire)** che prende gli argomenti sopra specificati. **La funzione `elabora` stampa a video la decodifica della stringa rispetto alla mappa data**.

La funzione `elabora` **deve essere ricorsiva** e **NON deve contenere iteratori** espliciti (`for`, `while`, `do-while`). La funzione `elabora` può ovviamente contenere codice sequenziale o condizionale. Sono consentite (se ritenute necessarie) chiamate a funzioni ricorsive ausiliarie che a loro volta **non contengano iterazioni esplicite** (`for`, `while`, `do-while`).

Il file `esercizio2.cpp` contiene tutto quanto necessario tranne la dichiarazione e la definizione della procedura `elabora`.

Di seguito sono riprotati due esempi di esecuzione del programma.

```
computer > ./a.out vuoto.txt 30

computer > ./a.out testo.txt 30
.VGILXH 'L'E LR'SX VHL
X VIGOZ'OOVW 'LIRW ,RZELIG RE L
R'SX MVY OVW IZGGZIG IVK ,ZN ;V
GILN 'FRK 'V LXLK VSX ZIZNZ 'V'
GMZG
!ZIFZK ZO ZELMRI IVRHMVK O
VM VSX ,VGILU V ZIKHZ GV ZRTTZE
OVH ZEOVH ZGHV ZIFW ZHLX 'V ZIV
 OZFJ IRW Z LGMZFJ V
.ZGRIIZNH
```

```
ZIV ZRE ZGGRIRW ZO 'VSX :ZIFXHL
 ZEOVH ZMF IVK RZELIGRI RN ZGRE
 ZIGHLM RW MRNNZX OVW LAAVN OVM
```

Note:

- Scaricare il file `esercizio2.cpp`, modificarlo per inserire il codice necessario per rispondere a questo esercizio. **Caricare il file sorgente risultato delle vostre modifiche a soluzione di questo esercizio** nello spazio apposito.
- Si può assumere che la stringa di input contenga solo caratteri numerici e spazi.
- All'interno di questo programma **non è ammesso** l'utilizzo di variabili globali o di tipo `static` e di funzioni di libreria al di fuori di quelle definite in `iostream`, `fstream`, `cstdlib`, `cstdio`.
- Si ricorda che, gli esempi di esecuzione sono puramente indicativi, e la soluzione proposta NON deve funzionare solo per l'input fornito, ma deve essere robusta a variazioni compatibili con la specifica riportata in questo testo.
- Si ricorda di inserire solo nuovo codice e di **NON MODIFICARE** il resto del programma (pena annullamento dell'esercizio).
- Si ricorda che la soluzione deve essere implementata in C++ **NON usando `string` e altri elementi della C++ standard template library, anche se il file compila senza cambiare gli header**!

esercizio2.cpp

testo.txt

vuoto.txt

Information for graders:

Total of marks: 20