1 Esercizio 1

Scrivere nel file `esercizio1.cc` un programma che, dati in input due file, un messaggio e un elenco di parole, generi un terzo file che contenga tutte le parole del messaggio censurando quelle **presenti** all'interno dell'elenco. Le parole devono essere sostiuite nel testo da una sequenza di caratteri **X** lunga quanto la parola da censurare.

Il programma dovrà accettare tre parametri da riga di comando: il file con il messaggio, il file con l'elenco delle parole e il file dove salvare l'output, in questo ordine. Il programma dovrà anche implementare dei controlli sul numero di argomenti passati da riga di comando e sull'apertura degli stream. Dovrà inoltre avvisare l'utente in caso di errori.

Procediamo ora con un esempio. Dati in input i file `messaggio_segreto.txt` e `vocabolario.txt` contenenti le seguenti parole:

messaggio_segreto.txt	**vocabolario.txt**
Eravamo effettivamente in possesso di un disco volante La copertura fu orchestrata molto bene da Washington Tenente Walter Haut Roswell New Mexico 1947	disco volante roswell walter haut washington

se l'eseguibile è `a.out`, allora il comando

```
./a.out messaggio_segreto.txt vocabolario.txt output
```

genererà un file chiamato `output` che conterrà le seguenti parole:

```
Eravamo effettivamente in possesso di un XXXXX XXXXXXX La copertura fu orchestrata
molto bene da XXXXXXXXXX Tenente XXXXXX XXXX XXXXXXX XXX XXXXXX 1947
```

Note

- Per semplicità, considerate come una *parola* qualsiasi stringa di caratteri compresa tra due spazi bianchi, caratteri di tabulazione e/o a capo. Non sono presenti segni di punteggiatura.
- Il confronto tra una parola del testo e una dell'elenco è **case-insensitive**. Ad esempio, "Roswell" e "roswell" sono da considerarsi identiche ai fini dell'esercizio.
- Non è consentito assumere un numero massimo di parole per il messaggio. Il vocabolario invece può contenere fino a 1000 parole formate da un massimo 100 caratteri ciascuna.
- E' consentito l'utilizzo della funzione `strlen` della libreria `<cstring>`. Non è invece consentito l'utilizzo di librerie, funzioni, e/o strutture dati "particolari" diverse da quelle specificate pena l'**annullamento dell'esercizio**.
- Il programma deve poter funzionare con qualsiasi codifica (non solo ASCII). Non è consentito utilizzare tabelle, `if-then-else`, oppure `switch`, uno per ogni carattere.
- E' consentito definire ed implementare funzioni ausiliarie che possano aiutarvi nella soluzione del problema.

2 Esercizio 1

Scrivere nel file `esercizio1.cc` un programma che, dati in input due file, un messaggio e un elenco di parole, generi un terzo file che contenga tutte le parole del messaggio censurando quelle **non presenti** all'interno dell'elenco. Le parole devono essere sostiuite nel testo da una sequenza di caratteri **X** lunga quanto la parola da censurare.

Il programma dovrà accettare tre parametri da riga di comando: il file con il messaggio, il file con l'elenco delle parole e il file dove salvare l'output, in questo ordine. Il programma dovrà anche implementare dei controlli sul numero di argomenti passati da riga di comando e sull'apertura degli stream. Dovrà inoltre avvisare l'utente in caso di errori.

Procediamo ora con un esempio. Dati in input i file `messaggio_segreto.txt` e `vocabolario.txt` contenenti le seguenti parole:

messaggio_segreto.txt	**vocabolario.txt**
Eravamo effettivamente in possesso di un disco volante La copertura fu orchestrata molto bene da Washington Tenente Walter Haut Roswell New Mexico 1947	disco volante roswell walter haut washington

se l'eseguibile è `a.out`, allora il comando

```
./a.out messaggio_segreto.txt vocabolario.txt output
```

genererà un file chiamato `output` che conterrà le seguenti parole:

```
XXXXXXX XXXXXXXXXXXXXX XX XXXXXXXX XX XX disco volante XX XXXXXXXXX XX XXXXXXXXXXX
XXXXX XXXX XX Washington XXXXXXX Walter Haut Roswell New Mexico XXXX
```

Note

- Per semplicità, considerate come una *parola* qualsiasi stringa di caratteri compresa tra due spazi bianchi, caratteri di tabulazione e/o a capo. Non sono presenti segni di punteggiatura.
- Il confronto tra una parola del testo e una dell'elenco è **case-insensitive**. Ad esempio, "Roswell" e "roswell" sono da considerarsi identiche ai fini dell'esercizio.
- Non è consentito assumere un numero massimo di parole per il messaggio. Il vocabolario invece può contenere fino a 1000 parole formate da un massimo 100 caratteri ciascuna.
- E' consentito l'utilizzo della funzione `strlen` della libreria `<cstring>`. Non è invece consentito l'utilizzo di librerie, funzioni, e/o strutture dati "particolari" diverse da quelle specificate pena l'**annullamento dell'esercizio**.
- Il programma deve poter funzionare con qualsiasi codifica (non solo ASCII). Non è consentito utilizzare tabelle, `if-then-else`, oppure `switch`, uno per ogni carattere.
- E' consentito definire ed implementare funzioni ausiliarie che possano aiutarvi nella soluzione del problema.

3 Esercizio 1

Scrivere nel file `esercizio1.cc` un programma che, dati in input due file, un messaggio e un elenco di parole, generi un terzo file che contenga tutte le parole del messaggio censurando quelle **presenti** all'interno dell'elenco. Le parole devono essere sostiuite nel testo da una sequenza di caratteri * lunga quanto la parola da censurare.

Il programma dovrà accettare tre parametri da riga di comando: il file con il messaggio, il file con l'elenco delle parole e il file dove salvare l'output, in questo ordine. Il programma dovrà anche implementare dei controlli sul numero di argomenti passati da riga di comando e sull'apertura degli stream. Dovrà inoltre avvisare l'utente in caso di errori.

Procediamo ora con un esempio. Dati in input i file `messaggio_segreto.txt` e `vocabolario.txt` contenenti le seguenti parole:

messaggio_segreto.txt	**vocabolario.txt**
Eravamo effettivamente in possesso di un disco volante La copertura fu orchestrata molto bene da Washington Tenente Walter Haut Roswell New Mexico 1947	disco volante roswell walter haut washington

se l'eseguibile è `a.out`, allora il comando

```
./a.out messaggio_segreto.txt vocabolario.txt output
```

genererà un file chiamato `output` che conterrà le seguenti parole:

```
Eravamo effettivamente in possesso di un ***** ******* La copertura fu orchestrata
molto bene da ********** Tenente ****** **** ******* *** ****** 1947
```

Note

- Per semplicità, considerate come una *parola* qualsiasi stringa di caratteri compresa tra due spazi bianchi, caratteri di tabulazione e/o a capo. Non sono presenti segni di punteggiatura.
- Il confronto tra una parola del testo e una dell'elenco è **case-insensitive**. Ad esempio, "Roswell" e "roswell" sono da considerarsi identiche ai fini dell'esercizio.
- Non è consentito assumere un numero massimo di parole per il messaggio. Il vocabolario invece può contenere fino a 1000 parole formate da un massimo 100 caratteri ciascuna.
- E' consentito l'utilizzo della funzione `strlen` della libreria `<cstring>`. Non è invece consentito l'utilizzo di librerie, funzioni, e/o strutture dati "particolari" diverse da quelle specificate pena l'**annullamento dell'esercizio**.
- Il programma deve poter funzionare con qualsiasi codifica (non solo ASCII). Non è consentito utilizzare tabelle, `if-then-else`, oppure `switch`, uno per ogni carattere.
- E' consentito definire ed implementare funzioni ausiliarie che possano aiutarvi nella soluzione del problema.

4 Esercizio 1

Scrivere nel file `esercizio1.cc` un programma che, dati in input due file, un messaggio e un elenco di parole, generi un terzo file che contenga tutte le parole del messaggio censurando quelle **non presenti** all'interno dell'elenco. Le parole devono essere sostiuite nel testo da una sequenza di caratteri * lunga quanto la parola da censurare.

Il programma dovrà accettare tre parametri da riga di comando: il file con il messaggio, il file con l'elenco delle parole e il file dove salvare l'output, in questo ordine. Il programma dovrà anche implementare dei controlli sul numero di argomenti passati da riga di comando e sull'apertura degli stream. Dovrà inoltre avvisare l'utente in caso di errori.

Procediamo ora con un esempio. Dati in input i file `messaggio_segreto.txt` e `vocabolario.txt` contenenti le seguenti parole:

messaggio_segreto.txt	**vocabolario.txt**
Eravamo effettivamente in possesso di un disco volante La copertura fu orchestrata molto bene da Washington Tenente Walter Haut Roswell New Mexico 1947	disco volante roswell walter haut washington

se l'eseguibile è `a.out`, allora il comando

```
./a.out messaggio_segreto.txt vocabolario.txt output
```

genererà un file chiamato `output` che conterrà le seguenti parole:

```
******* ************** ** ******** ** ** disco volante ** ********* ** ***********
***** **** ** Washington ******* Walter Haut Roswell New Mexico ****
```

Note

- Per semplicità, considerate come una *parola* qualsiasi stringa di caratteri compresa tra due spazi bianchi, caratteri di tabulazione e/o a capo. Non sono presenti segni di punteggiatura.
- Il confronto tra una parola del testo e una dell'elenco è **case-insensitive**. Ad esempio, "Roswell" e "roswell" sono da considerarsi identiche ai fini dell'esercizio.
- Non è consentito assumere un numero massimo di parole per il messaggio. Il vocabolario invece può contenere fino a 1000 parole formate da un massimo 100 caratteri ciascuna.
- E' consentito l'utilizzo della funzione `strlen` della libreria `<cstring>`. Non è invece consentito l'utilizzo di librerie, funzioni, e/o strutture dati "particolari" diverse da quelle specificate pena l'**annullamento dell'esercizio**.
- Il programma deve poter funzionare con qualsiasi codifica (non solo ASCII). Non è consentito utilizzare tabelle, `if-then-else`, oppure `switch`, uno per ogni carattere.
- E' consentito definire ed implementare funzioni ausiliarie che possano aiutarvi nella soluzione del problema.