Università di Catania
Dipartimento di Matematica e Informatica
Corso di Studio in Informatica, A.A. 2022-2023
Programmazione 1 e Laboratorio
Prova d'esame di laboratorio
3 Febbraio 2023
B

Descrizione del programma

Si scriva un programma C che:

- **A** chieda all'utente di inserire un intero L (si assuma n<256), una stringa Z, una stringa W, e un carattere K da tastiera. Si verifichi che entrambe le stringhe Z e W siano di lunghezza L. In caso contrario, si stampi un errore su standard error e si termini il programma con un opportuno codice di terminazione;
- **B** costruisca una nuova stringa "s3" ottenuta sostituendo tutte le occorrenze del carattere "c" in "s1" con i caratteri che si trovano in "s2" nelle posizioni corrispondenti;
- **C** definisca una nuova stringa "s4" ottenuta invertendo l'ordine dei caratteri in "s2";
- **D** concateni le stringhe "s3" e "s4" in una nuova stringa "s5" e la ordini in ordine lessicografico ascendente usando un algoritmo di ordinamento a scelta;
- **E** stampi a schermo la stringa ordinata. I caratteri i cui codici numerici relativi (fare cast dei caratteri a int) siano dispari, vanno sostituiti con "*".

Specifiche

Il programma potrà essere articolato in un unico file sorgente, ma dovrà contenere almeno le seguenti funzioni con opportuni parametri formali:

- **readInput**: funzione che permette di leggere gli input da tastiera e restituisce opportunamente i valori letti al chiamante. La funzione deve gestire correttamente gli errori relativi a input non corretti e gestire la terminazione del programma in caso di errori.
- **replaceChar**: funzione che riceve come input le stringhe "s1", "s2" e il carattere "c" e restituisce la stringa con le occorrenze di "c" in "s1" sostituite dai corrispondenti valori in "s2. Ad esempio, se

s1=”abcacba”, s2=”fhsuika” e c=’a’, allora il risultato sarà “fbcucba”;

- **invertString**: funzione che costruisce una stringa a partire da una stringa in input invertendono i caratteri;
- **sort**: funzione che permetta di ordinare una stringa in ordine lessicografico;
- **printResult**: funzione che stampa a schermo la stringa ottenuta come specificato nel punto E.

Note

Durata della prova: 120 minuti

Generazione di numeri pseudocasuali:

- Si consideri la seguente funzione get_random() per la generazione di numeri pseudo-casuali interi positivi (qualora necessaria):

```
// Scaricabile da: https://pastebin.com/f6eAKNQy
unsigned int get_random() {
    static unsigned int m_w = 123456;
    static unsigned int m_z = 789123;
    m_z = 36969 * (m_z & 65535) + (m_z >> 16);
    m_w = 18000 * (m_w & 65535) + (m_w >> 16);
    return (m_z << 16) + m_w;
}
```

- NB: Ai fini della generazione di numeri in virgola mobile, si faccia uso della costante UINT_MAX (<limits.h>) unitamente alla funzione get_random().

È VIETATO usare variabili globali.

Output di controllo

Eseguendo il programma con i seguenti input:

```
15
jsudbehdnbvsju9
kduw76_lposhndb
s
```

Il programma dovrà stampare il seguente output:

```
6***bbbdddddd*hhhjj*lnn*p****v*
```

CONSEGNA:

- al file sorgente che contiene lo svolgimento dell'esercizio va assegnato il seg. nome: [MATRICOLA].c;
- All'interno del file sorgente, in un commento, inserire nome e cognome e matricola;
- il file sorgente va lasciato all'interno della home directory;
- il file sorgente sara' prelevato automaticamente allo scadere dei 120 minuti.