Laboratorio di Programmazione: Linguaggio C
Lezione 17 del 24 marzo 2014

Damiano Macedonio

Esercizio 1

Scrivere un programma C che richiede all'utente di inserire una stringa e ritorna:

- la lunghezza della stringa,
- il numero di vocali contenute nella stringa,
- il numero di consonanti contenute nella stringa.

Suggerimento: Per scorrere la stringa, utilizzare un ciclo while che itera finché non viene raggiunto il carattere nullo.

Esercizio 2

Scrivere un programma C che richiede all'utente due stringhe, crea una stringa che corrisponde alla loro concatenazione e la scrive a video.

Esercizio 3

Scrivere un programma C che richiede all'utente due stringhe `a` e `b`, e determina se `b` è contenuta in `a`.

Suggerimento:

- Determinare l'inizio di `b` in `a`: cercare la posizione `i` in `a` del primo carattere di `b`.
- Se `a` non contiene il primo carattere di `b`, allora sicuramente `b` non è contenuta in `a`.
- Altrimenti procedere dalla posizione `i` in `a` e controllare l'uguaglianza con gli altri caratteri di `b`, finché non si raggiunge la fine di una delle due stringhe o si trovano due caratteri diversi.
- Se si raggiunge la fine di `b`, allora tutta la stringa `b` è contenuta in `a`. Se si raggiunge la fine di `a` senza raggiungere la fine di `b`, allora `b` non è contenuta in `a`. Altrimenti si riprende dalla posizione `i + 1` e si cerca ancora la posizione del primo carattere di `b`...

Esercizio 4

Scrivere un programma C che richiede all'utente di inserire due stringhe e determina se sono sono uguali o diverse, *a prescindere da caratteri maiuscoli o minuscoli.*

Nel caso di stringhe diverse il programma deve visualizzarle in ordine lessicografico e, in subordine, in ordine di lunghezza. In altre parole: l'ordine tra i caratteri è determinato dalla tabella ASCII e, nel caso una stringa sia prefisso dell'altra, viene prima quella più corta. Per esempio: se le due stringhe sono `"ora"` e `"orario"`, si deve scrivere prima `"ora"` e poi `"orario"`.

Le seguenti funzioni devono essere dichiarate nel header file `my_functions.h` e definite nel file `my_functions.c`:

`int compareTo(char this [], char that [])` Considera l'ordine lessicografico, la lunghezza in subordine, e restituisce: un valore negativo se `this` precede `that`, un valore positivo se `this` segue `that`, il valore 0 se le due stringhe sono uguali.

`void copy_to_lower_cases(char this [], char that [])` Copia la stringa `this` nella stringa `that` utilizzando solo caratteri minuscoli. Per esempio, se `this` è `"LaTeX-2e"` allora `that` diventa `"latex-2e"`.

`bool equals_ignore_cases(char [], char [])` Verifica se due stringhe sono uguali, a prescindere da caratteri maiuscoli o minuscoli;

Nel file `main.c` deve essere definita la funzione `int main(void)` che richiama le precedenti funzioni per risolvere l'esercizio.

Esempio:

```
$ ./a.out
Prima stringa massimo 80 caratteri:   BaNaNa
Seconda stringa massimo 80 caratteri: bAnAnA
Le stringhe sono uguali, a prescindere da caratteri minuscoli/maiuscoli.
$ ./a.out
Prima stringa massimo 80 caratteri:   BaNaNa
Seconda stringa massimo 80 caratteri: bAndAnA
Le stringhe sono diverse,
eccole scritte in ordine alfabetico:  BaNaNa bAndAnA
$ ./a.out
Prima stringa massimo 80 caratteri:   bananA
Seconda stringa massimo 80 caratteri: banDaNa
Le stringhe sono diverse,
eccole scritte in ordine alfabetico:  banDaNa bananA
$ ./a.out
Prima stringa massimo 80 caratteri:   bAnDa
Seconda stringa massimo 80 caratteri: bAnDaNa
Le stringhe sono diverse,
eccole scritte in ordine alfabetico:  bAnDa bAnDaNa
```

Esercizio 5 [Stringhe Palindrome]

Diciamo che una stringa è *palindroma* se rimane identica quando letta a rovescio. Alcuni esempi: anna, radar, ottetto, ingegni, onorarono...

Scrivere un programma C composto dalle seguenti funzioni:

`leggi()` Richiede all'utente di inserire una stringa di al massimo 80 caratteri.

`palindroma()` Verifica se la stringa inserita è palindroma.

`main()` Richiama le due funzioni precedenti e stampa il risultato.

La funzione `palindroma()` può essere implementata in modo ricorsivo. Una stringa è palindroma se:

Caso base Se la stringa è vuota oppure ha un solo carattere, la stringa è palindroma.

Passo ricorsivo Se la stringa è composta da almeno 2 caratteri, allora essa è palindroma se e solo se il primo e l'ultimo carattere sono uguali ed è palindroma la stringa che va dal secondo carattere al penultimo.