Programmazione I e laboratorio: prova pratica
Tempo a disposizione: 2 ore

Corso B

Appello straordinario - 02/04/2019

Implementare il programma richiesto usando il linguaggio standard ANSI C e sottomettere la soluzione sulla piattaforma di esame.

Il codice consegnato sarà valutato solo se supera almeno il 51% dei test case. La valutazione terrà conto della correttezza del programma, strutture dati usate, uso efficiente di memoria, efficienza dell'implementazione algoritmica, stile di programmazione, controlli dell'input.

Non è consentito l'uso di nessun materiale o strumento tecnologico oltre il computer del laboratorio. Si può usare qualsiasi IDE installato sul computer.

Non è consentita la collaborazione tra studenti.

Se si desidera abbandonare l'esame occorre dirlo esplicitamente ai docenti per evitare la consegna del codice.

Compilazione Si ricorda di compilare con l'opzione ANSI C e che per abilitare i messaggi di diagnostica del compilatore, bisogna compilare il codice usando le opzioni `-g -Wall` di gcc:

```
gcc -std=c89 -Wall -g sorgente.c -o eseguibile
```

Provare la propria soluzione in locale. Valutare la correttezza della soluzione sulla propria macchina accertandosi che rispetti gli input/output contenuti nel TestSet. I file di input e output per i test sono nominati secondo lo schema:
`input0.txt output0.txt`
`input1.txt output1.txt`
...
Per effettuare le vostre prove potete utilizzare il comando del terminale per la redirezione dell'input. Ad esempio
`./eseguibile < input0.txt`
effettua il test del vostro codice sui dati contenuti nel primo file di input, assumendo che `eseguibile` sia il file ottenuto dalla compilazione del vostro codice. Per effettuare un controllo automatico sul primo file input `input0.txt` e trovare le differenze fra l'output prodotto dal vostro programma e quello corretto potete eseguire la seguente sequenza di comandi

```
./eseguibile < input0.txt | diff - output0.txt
```

Esercizio

Scrivere un programma che riceve da standard input una stringa e verifica se la stringa è palindroma. Si ricorda che una stringa è palindroma se letta al contrario rimane invariata ("abcdgdcba", "12aa21" sono palindrome, "1234323" non è palindroma). Il programma deve stampare il messaggio "Stringa palindroma" se la stringa è palindroma altrimenti "Stringa non palindroma".

Il programma deve rispettare le seguente restrizioni:

- Non si possono usare array. Le soluzioni implementate con array **non** saranno accettate.
- Si deve usare una struttura dati di tipo Pila, implementata usando liste concatenate. Si ricorda che una pila e una lista di elementi dove gli elementi vengono sempre inseriti in testa e rimossi dalla testa, quindi seguendo l'ordine LIFO (last in first out, ultimo inserito primo rimosso). Il programma deve definire le `struct` necessarie e due funzioni: `pop` che rimuove un elemento dalla pila e lo restituisce e `push` che inserisce un nuovo elemento nella pila. Le soluzioni che non usano una pila **non** saranno accettate.
- la verifica della palindromia della stringa deve essere fatta usando una funzione ricorsiva. Le soluzioni che **non usano una funzione ricorsiva** per la verifica della palindromia saranno automaticamente **penalizzate di 10 punti**, quindi il voto massimo che si può ottenere è di 14 punti.

L'input è così composto:

- Il primo valore immesso è un numero intero che corrisponde alla lunghezza della stringa n. Il programma deve controllare che questo numero sia un intero non negativo. Nel caso contrario il programma esce col messaggio "Input non corretto".
- Segue una stringa di dimensione n seguita da un'interlinea.

Si ricorda che non vi deve essere spreco di memoria nell'allocazione della memoria. Sarà valutata anche l'organizzazione del codice: uso delle funzioni, commenti, indentazione, semplicità.

Esempio

Input

```
 7
ingegni
```

Output

```
Stringa palindroma
```

Input

```
 7
ingegno
```

Output

```
Stringa non palindroma
```

Input

```
 a
informatica
```

Output

```
Input non corretto
```

Input

```
 -6
casa
```

Output

```
Input non corretto
```