Esercizi di programmazione in C

Alessandro Pellegrini

Per motivi didattici, si consiglia agli studenti di utilizzare la toolchain di compilazione anche per osservare qual è il codice assembly generato dal compilatore per i programmi che verranno scritti per implementare le soluzioni agli esercizi. Si può utilizzare `cc -S` sul file sorgente o `objdump -d` su un file oggetto (generato con `cc -c`) o sul file eseguibile.

Nel file eseguibile sarà comunque presente più codice e più dati, legati al collegamento della libreria standard al programma.

Esercizio 1 : Si scriva un programma in linguaggio C che legga un numero qualsiasi di interi positivi e visualizzi la loro media aritmetica. Si può utilizzare lo zero come indicazione che non si intende acquisire più numeri.

Esercizio 2 : Si scriva un programma in linguaggio C capace di compiere le 4 operazioni (somma, sottrazione, moltiplicazione e divisione) tra due numeri reali inseriti da tastiera. Dopo che sono stati inseriti i due numeri, detti A e B, il programma dovrà visualizzare i quattro valori $A + B$, $A - B$, $A \cdot B$, A/B. Si ipotizzi che sia $B \neq 0$.

Esercizio 3 : Si scriva un programma in linguaggio C che, dato un numero reale D immesso da tastiera, calcoli e stampi:

1. l'area del quadrato di lato D
2. l'area del cerchio di diametro D
3. l'area del triangolo equilatero di lato D

Esercizio 4 : Si realizzi un programma in linguaggio C che acquisisca da tastiera un numero e stampi un messaggio che indichi se tale numero sia positivo oppure negativo.

Esercizio 5 : Si realizzi un programma in linguaggio C che acquisisca da tastiera un numero e stampi il valore assoluto di tale numero.

Esercizio 6 : Data l'equazione

$$ax + b = 0$$

con a e b inseriti da tastiera, scrivere un programma in linguaggio C per determinare il valore di x, se esiste, che risolve l'equazione.

Esercizio 7 : Dato un numero intero tra 1 e 12, che rappresenta il mese corrente, stampare il nome del mese per esteso ("Gennaio" ... "Dicembre").

Esercizio 8 : Si realizzi un programma in linguaggio C per risolvere equazioni di secondo grado. In particolare, data una generica equazione di secondo grado nella forma

$$ax^{+}bx + c = 0$$

dove a, b, c sono coefficienti reali noti e x rappresenta l'incognita, il programma determini le due radici x_1 ed x_2 dell'equazione data, ove esse esistano. Si identifichino tutti i casi particolari ($a = 0$, $\Delta \leq 0$, ...) e si stampino gli opportuni messaggi informativi.

Esercizio 9 : Si scriva un programma in linguaggio C che converta un numero binario in un numero decimale. Il numero binario è rappresentato su N bit, definito come costante nel programma. L'utente inserisce le cifre del numero binario un bit alla volta, partendo dal bit più significativo. Il programma visualizzerà il numero decimale corrispondente.

Esercizio 10 : Si scriva un programma in linguaggio C per calcolare la media aritmetica di una serie di numeri inseriti da tastiera. L'introduzione di un valore particolare pari a "0" indica il termine del caricamento dei dati.

Esercizio 11 : Si scriva un programma in linguaggio C per calcolare il valore massimo e minimo di un insieme di N numeri inseriti da tastiera. Il programma deve leggere il valore di N, ed in seguito deve leggere una sequenza di N numeri. A questo punto il programma deve stampare il massimo ed il minimo tra i numeri inseriti.

Esercizio 12 : Si scriva un programma in linguaggio C per poter analizzare una sequenza di numeri. Dati N numeri interi letti da tastiera si vogliono calcolare e stampare su schermo diversi risultati:

- quanti sono i numeri positivi, nulli e negativi
- quanti sono i numeri pari e dispari
- se la sequenza dei numeri inseriti è crescente, decrescente oppure né crescente né decrescente.

Esercizio 13 : Si scriva un programma in linguaggio C per calcolare il massimo comun divisore (MCD) di due numeri interi positivi. Il MCD è definito come il massimo tra i divisori comuni ai due numeri.

Esercizio 14 : Si scriva un programma in linguaggio C per calcolare il minimo comune multiplo (MCM) di due numeri interi positivi. Dati due numeri interi N1 e N2, il minimo comune multiplo è il più piccolo numero M che è divisibile (con resto pari a zero) sia per N1 che per N2.

Esercizio 15 : Scrivere un programma in linguaggio C per la rappresentazione del triangolo di Floyd. Il triangolo di Floyd è un triangolo rettangolo che contiene numeri naturali, definito riempiendo le righe del triangolo con numeri consecutivi e partendo da 1 nell'angolo in alto a sinistra.

Si consideri ad esempio il caso $N = 5$. Il triangolo di Floyd è il seguente:

```
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

Il programma riceve da tastiera un numero intero N. Il programma visualizza le prime N righe del triangolo di Floyd.

Esercizio 16 : Scrivere un programma in linguaggio C che riceva in ingresso un numero binario rappresentato in complemento a 2 su N bit, con N costante definita all'interno del programma. Le cifre sono inserite a partire dal bit più significativo. Il programma calcola e stampa l'opposto del numero binario ricevuto in ingresso.

Esercizio 17 : Si considerino due numeri binari rappresentati in binario puro su N bit, con N costante definita all'interno del programma. I due numeri sono inseriti da tastiera un bit alla volta a partire dal bit meno significativo (LSB). Si scriva un programma in linguaggio C per eseguire la somma dei due numeri. Il programma deve visualizzare il risultato delle somma, ed indicare se si è verificata la condizione di overflow.

Esercizio 18 : Scrivere un programma in linguaggio C che converta un numero decimale D in un numero binario rappresentato su N bit, con N costante definita all'interno del programma. Il programma visualizzerà i bit che compongono il numero binario partendo dal bit più significativo. Il programma segnalerà un errore se il numero N di bit inserito dall'utente non è sufficiente per rappresentare il numero decimale.

Esercizio 19 : Scrivere un programma in linguaggio C che riceve in ingresso una sequenza di numeri interi positivi e li memorizzi in un vettore. Terminato l'inserimento della sequenza di numeri, l'utente inserisce un valore di riferimento. Il programma deve indicare se tale valore di riferimento è contenuto nel vettore. Non è noto a priori il numero di valori che l'utente inserirà. L'inserimento termina quando l'utente inserisce una stringa vuota. In caso di inserimento errato, il programma scarta il valore non corretto.

Esercizio 20 : Scrivere un programma in linguaggio C che riceve in ingresso una sequenza di N numeri interi di dimensione byte, con N costante definita all'interno del programma. Il programma genera un *istogramma* dei valori inseriti, ossia conta le occorrenze di ciascun valore all'interno del vettore. Ad esempio, se viene inserita la sequenza 1,2,3,3,5,3,6,6,8,2,1, il programma stamperà:

```
1: **
2: **
3: ***
```

```
5: *
6: **
8: *
```

I valori con numero di occorrenze pari a zero non vengono stampati.

Esercizio 21 : Siano dati due vettori di interi. Si scriva un programma in linguaggio C che generi un terzo vettore di interi che contiene l'intersezione tra i due vettori. Tale vettore deve contenere i numeri presenti in entrambi i vettori dati.

Esercizio 22 : Scrivere un programma in linguaggio C che, data una stringa, verifichi se essa è palindroma.

Esercizio 23 : Scrivere un programma in linguaggio C che, date due stringhe, verifichi se la più breve delle due è una sottostringa dell'altra.

Esercizio 24 : Si scriva un programma in linguaggio C che, dato un percorso ad un file di testo passato come argomento di programma, calcoli:

1. il numero totale di righe che compongono il file;
2. il numero totale di caratteri che compongono il file (esclusi spazi);
3. il numero totale di caratteri alfanumerici che compongono il file;
4. il numero totale di parole che compongono il file;
5. il numero massimo e medio di caratteri in una riga del file;
6. la lunghezza della riga più lunga incontrata nel file;
7. il contenuto della riga più lunga incontrata nel file.

Esercizio 25 : Si scriva una funzione in C, denominata iniziali, che valuti quanti caratteri iniziali sono in comune tra due stringhe date. La funzione riceve due parametri, entrambi di tipo stringa, e restituisce il numero intero.

Esercizio 26 : Si scriva una funzione in C, denominata `alltoupper`, che converta in maiuscolo tutti i caratteri della stringa passata come parametro

Esercizio 27 : Un'azienda ha dotato i propri dipendenti di un sensore wireless che emette un codice numerico ogni volta che un dipendente attraversa la porta d'ingresso/uscita dell'azienda o ne transita nelle vicinanze. L'azienda ha meno di 1000 dipendenti. Ad ogni attraversamento, il sensore registra ora e minuti del passaggio, insieme al codice del dipendente (un codice alfanumerico di max 10 caratteri).

Si desidera sviluppare un programma in linguaggio C per il calcolo delle ore lavora- tive dei dipendenti dell'azienda. Il programma riceve sulla linea di comando un primo parametro, che rappresenta il nome del file contenente gli attraversamenti, ed un secondo parametro (opzionale), che rappresenta il codice numerico di un dipendente.

Il file è relativo ai passaggi di una sola giornata, ed è composto da una serie di righe, ciascuna delle quali corrisponde ad un passaggio, ed è composta da tre campi:

```
ora minuti codice_dipendente
```

Se il programma viene invocato con due parametri sulla linea di comando, allora dovrà stampare, per il dipendente specificato, il numero totale di minuti lavorati. Per determinare il numero di minuti lavorati occorre confrontare l'orario del primo passaggio con l'orario dell'ultimo passaggio per quel dipendente.

Se invece il programma viene invocato con un solo parametro sulla linea di comando, allora il programma dovrà stampare il numero totale di dipendenti diversi che hanno lavorato in quel giorno (ossia che sono passati almeno una volta dalla porta). Ad esempio, dato il seguente file di testo `passaggi.txt`:

```
8 30 abc222
8 30 abc123
8 31 azx112
9 10 abc123
12 10 abc123
```

il programma si dovrà comportare nel modo seguente:

```
$ ./orario passaggi.txt
Ci sono 3 dipendenti diversi.
$ ./orario passaggi.txt abc123
Il dipendente abc123 ha lavorato per 220 minuti.
```

Esercizio 28 : Si desidera sviluppare un programma in linguaggio C per la modifica di un file di testo. La modifica consiste nel sostituire, scambiandoli tra loro, due caratteri alfabetici dati. In particolare, tutte le occorrenze del primo carattere dovranno essere sostituite dal secondo e viceversa. La sostituzione deve avvenire mantenendo la forma (maiuscola o minuscola) della lettera originaria.

Il programma riceve tre parametri a riga di comando: il nome del file di testo da elaborare, il nome di un secondo file di testo nel quale salvare il risultato ed una stringa di 2 caratteri che specifica i caratteri da scambiare. Il file di testo è composto da un numero arbitrario di linee.

Ad esempio, se il programma venisse attivato con la seguente riga di comando:

```
./scambia testo.txt modificato.txt ae
```

ed il file `testo.txt` contenesse i seguenti dati:

```
QUEL RAMO del lago di Como, che volge a mezzogiorno,
tra due CATENE non interrotte di MONTI, tutto a seni e a golfi,
a seconda dello sporgere E DEL RIENTRARE di quelli, vien, quasi
```

allora il programma dovrebbe produrre il seguente file `modificato.txt`:

```
QUAL REMO dal lego di Como, cha volga e mazzogiorno,
tre dua CETANA non intarrotta di MONTI, tutto e sani a e golfi,
e saconde dallo sporgara A DAL RIANTRERE di qualli, vian, quesi
```

Esercizio 29 : Data una stringa A letta da tastiera, si realizzi un programma in linguaggio C che calcoli una seconda stringa B ottenuta dalla prima cancellando tutti i caratteri che compaiono più di una volta. La stringa risultante deve dunque contenere i caratteri della prima stringa, nello stesso ordine, ma senza ripetizioni. Esempio: `ACCIDENTI AL TRAFFICO` diviene `ACIDENT LRFO`.

Esercizio 30 : Si scriva un programma `entab` che rimpiazza sequenze di spazi in un file con il numero minimo di tab (`¬`) e spazi per ottenere la stessa spaziatura. Il risultato deve essere scritto in un nuovo file con un nome differente. Si consideri che un tab equivale ad 8 spazi.

Esercizio 31 : Scrivere un programma per mandare a capo le righe di input troppo lunghe in due o più righe più corte dopo l'ultimo carattere non vuoto che si presenta prima della n-esima colonna di input. Assicuratevi che il vostro programma faccia qualcosa di intelligente con righe molto lunghe e se non ci sono spazi vuoti o tabulazioni prima della colonna specificata.

Esercizio 32 : Scrivere un programma per rimuovere tutti i commenti dal sorgente di un programma C. Non dimenticatevi di gestire le stringhe e le costanti di tipo carattere in maniera appropriata. Nel C, i commenti non sono annidati.

Esercizio 33 : Scrivere un programma per verificare se un programma C ha dei semplici errori di sintassi, come parentesi non bilanciate. Considerate anche virgolette (singole e doppie) e considerate commenti e caratteri di escape. Questo è un programma difficile da realizzare se cercate di renderlo generale.

Esercizio 34 : Implementare una funzione `squeeze(char *s1, char *s2)` che elimini tutte le occorrenze in `s1` di ciascun carattere presente in `s2`.

Esercizio 35 : Implementare una funzione `invert(unsigned x, int p, int n)` che restituisce `x` con gli `n` bit partendo dalla posizione `p` invertiti, lasciando invariati gli altri.

Esercizio 36 : In un sistema che utilizza la rappresentazione in complemento a due, `x &= (x-1)` azzera il bit meno significativo impostato a 1 in `x`. Spiegate perché e sfruttate questa proprietà per scrivere una funzione `bitcount` che conta il numero di bit impostati a 1 in un intero.

Esercizio 37 : Scrivere un programma `expr` che valuti un'espressione in notazione polacca inversa passata a riga di comando, in cui ciascun operatore o operando è un argomento differente. Ad esempio:

```
./expr 2 3 4 + *
```

calcola $2 \times (3 + 4)$.

Esercizio 38 : Scrivere un programma `tail` che stampa le ultime n linee del testo passato in input. Di default n è 10, ma può essere cambiato utilizzando un argomento addizionale `-n`. Ad esempio:

```
./tail -3
```

stamperà le ultime tre linee. Il programma deve comportarsi in maniera razionale indipendentemente da quanto irragionevole è il valore n.

Esercizio 39 : Scrivere un programma che stampa una lista di tutte le parole in un documento e, per ciascuna parola, una lista di numeri di riga in cui la parola compare.

Esercizio 40 : Scrivere un programma che stampi le parole distinte fornite in input, ordinate in ordine di frequenza di occorrenza decrescente. Fate precedere ciascuna parola dal suo numero di occorrenze.

Esercizio 41 : Scrivere un programma per confrontare due file: esso stamperà la prima linea in cui sono differenti.