

Programmazione  
II Compitino (Versione A)

17 gennaio 2013

Cognome **Fatto** Nome **Affare**  
Matricola **666999**

1. Scrivete un programma che chieda ripetutamente all'utente di inserire degli interi, e che termini quando l'utente inserisce il valore 0; potete assumere che l'utente non inserisca più di 100 interi. Al termine deve stampare una sequenza di numeri, in cui l'*i*-esimo elemento è il numero di volte per cui l'*i*-esimo numero è comparso. Ad esempio, se l'utente inserisce *61 300 506 61 601 300 300 503 0* il programma deve stampare *2 3 1 2 1 3 3 1* (infatti, il 61 compare 2 volte, il 300 compare 3 volte, il 506 compare 1 volta, il 61 compare 2 volte ecc.).

```
#include <stdio.h>
#define MAX 100

int main() {
    int a[MAX], n, i, j, conta;

    for ( n = 0; n < MAX; n++ ) {
        scanf( "%d", &a[n] );
        if ( a[n] == 0 ) break;
    }
    for ( i = 0; i < n; i++ ) {
        conta = 0;
        for ( j = 0; j < n; j++ )
            if ( a[i] == a[j] ) conta++;
        printf( "%d_", conta );
    }
    printf( "\n" );
    return 0;
}
```

2. Scrivete una funzione che, data una stringa e un carattere, restituisca il numero di volte che il carattere compare nella stringa *alla fine di una parola*; con questa espressione, intendiamo che il carattere deve o comparire alla fine della stringa, o appena prima di uno spazio. Ad esempio, se la stringa fosse “la mamma ha fatto la manza” e il carattere fosse 'a', la funzione dovrebbe restituire 5.

```
int f( char *s, char c ) {
    int i, n, conta;
    n = strlen( s );
    for ( conta = i = 0; i < n; i++ )
        if ( s[i] == c && ( i == n - 1 || s[i+1] == ' ' ) )
            conta++;
    return conta;
}
```

3. Un magazzino contiene alcune casse, ciascuna caratterizzata da:
- il nome del prodotto contenuto nella cassa (una stringa di al più 100 caratteri);
  - le misure (lunghezza, larghezza, altezza), in metri, della cassa (dei valori float);
  - il peso, in grammi, della cassa (un valore float);
  - il numero di unità di quel prodotto presenti nella cassa.

Definite una struttura C, di nome `prodotto`, che rappresenti un prodotto.

```
#define MAXSTR 100
typedef struct {
    char nome[MAXSTR+1];
    float lung , larg , alt ;
    float peso;
    int numero;
} prodotto;
```

Ora scrivete una funzione che, dato un array di prodotti e un numero  $n$  (il numero di elementi dell'array effettivamente utilizzati), calcoli e restituisca il peso specifico più alto<sup>1</sup>.

```
float f( prodotto p[], int n ) {
    float max, curr;
    int i;
    max = 0;
    for ( i = 0; i < n; i++ ) {
        curr = p[i].peso / ( p[i].lung * p[i].larg * p[i].alt );
        if ( curr > max ) max = curr;
    }
    return max;
}
```

---

<sup>1</sup>Il peso specifico è il rapporto fra il peso e il volume della cassa.

4. Scrivete un programma che, ricevendo sulla riga di comando una sequenza di numeri interi, calcoli e stampi la loro media (come valore floating point). Ad esempio, se supponete che il programma si chiami `media`, un tipico esempio di esecuzione potrebbe essere

```
[gnagna]>./media 5 15 3
7.666666666666666
[gnagna]>
```

```
#include <stdio.h>
#include <stdlib.h>

int main( int argc, char **argv ) {
    int i;
    float tot;
    for ( i = 1; i < argc; i++ ) tot += atoi( argv[i] );
    printf( "%f\n", tot / ( argc - 1 ) );
    return 0;
}
```

5. Scrivete una funzione con tre argomenti: il primo è un intero ( $x$ ) il secondo e il terzo sono un puntatori a intero ( $y$  e  $z$ ). La funzione deve operare diversamente a seconda che  $y$  e  $z$  si riferiscano o no alla stessa zona di memoria; se sì, la funzione deve copiare il valore di  $x$  nei due interi (ovvero in uno di essi), mentre in caso contrario deve mettere nella zona puntata da  $y$  il valore  $x$  e nella zona puntata da  $z$  il *doppio* di  $x$ .

```
void f( int x, int *y, int *z ) {  
    if ( y == z )  
        *y = x;  
    else {  
        *y = x;  
        *z = 2 * x;  
    }  
}
```

6. Supponete di avere dichiarato un tipo `persona` come segue:

```
typedef struct {
    char nome[ 500 ], cognome[ 500 ];
    char sesso; /* vale 'm' oppure 'f' */
    int anno_di_nascita;
} persona;
```

Scrivete una funzione che, dato un array di persone e la sua lunghezza, determini e restituisca l'anno di nascita del maschio più vecchio e l'anno di nascita della femmina più vecchia, e memorizzi questi valori in due variabili intere i cui indirizzi sono passati come argomenti.

```
void f( persona p[], int n, int *anno_m, int *anno_f ) {
    int i;
    *anno_m = INT_MAX;
    *anno_f = INT_MAX;
    for ( i = 0; i < n; i++ ) {
        if ( p[i].sesso == 'm' && p[i].anno_di_nascita < *anno_m )
            *anno_m = p[i].anno_di_nascita;
        if ( p[i].sesso == 'f' && p[i].anno_di_nascita < *anno_f )
            *anno_f = p[i].anno_di_nascita;
    }
}
```

7. Data la definizione di `prodotto` (vedi sopra), scrivete una funzione che riceve un array di prodotti e la sua lunghezza, il nome di un prodotto, e restituisce quante unità di quel prodotto sono presenti in magazzino (notate che lo stesso prodotto può comparire in più casse).

```
int f( prodotto p[], int n, char *s ) {
    int i, c;
    for ( c = i = 0; i < n; i++ )
        if ( strcmp( p[i].nome, s ) == 0 )
            c+=p[i].numero;
    return c;
}
```