Cicli e iterazioni

Esercizi risolti

1 Esercizio: "Indovina cosa..."

Determinare il valore visualizzato dai seguenti programma nel caso in cui `num=4` e per i seguenti valori della variabile `conta`: `conta=5`, `conta=0`, `conta=1`, `conta= -5`.

```
int conta, num;
scanf("%d", &conta);
scanf("%d", &num);

while (conta != 0)
{
    num = num * 10;
    conta = conta - 1;
}
printf("%d\n", num) ;
```

```
int conta, num;
scanf("%d", &conta);
scanf("%d", &num);

while (conta > 0)
{
    num = num * 10;
    conta = conta - 1;
}
printf("%d\n", num) ;
```

Soluzione

Si nota innanzitutto come i due programmi siano identici tranne per la condizione `conta!=0`, che diviene `conta>0` nel secondo. Ciò significa che i due programmi si comporteranno in modo identico ogniqualvolta `conta` sarà un valore positivo o nullo (perché in tal caso le due condizioni `conta!=0` e `conta>0` si equivalgono), mentre si potranno comportare diversamente quando `conta<0`.

Analizzando il ciclo, si nota che l'operazione principale eseguita è `num=num*10`, che viene ripetuta `conta` volte. In pratica il programma calcola un valore finale pari a $num * 10^{conta}$.

In definitiva il valore calcolato (e stampato) sarà:

	Programma di sinistra	Programma di destra
`num=4, conta=5`	400000	400000
`num=4, conta=0`	4	4
`num=4, conta=1`	40	40
`num=4, conta=-5`	(*)	4

(*) in questo caso il programma esibisce un comportamento anomalo, dovuto ad un errore di programmazione (non ci si è "protetti" contro un dato errato, ossia negativo, inserito dall'utente). Il ciclo viene eseguito un'enormità di volte (dell'ordine di 2^{32} volte), finché il valore di `conta`, che parte da -5 e viene decrementato ripetutamente fino a quando la sottrazione non andrà in overflow, e poi nuovamente finché non arriverà a zero. In tal caso `num` viene moltiplicato per 10 un'enormità di volte, andando ripetutamente in overflow... il risultato ottenuto sarà quindi totalmente imprevedibile (e tra l'altro dipendente dall'implementazione degli **`int`** nel compilatore utilizzato). A titolo di esempio, nel caso del compilatore Dev-C++ su piattaforma Windows, dopo circa 20 secondi (durante i quali il programma decrementa `conta` all'impazzata) viene stampato il valore 0.

2 Esercizio "Conversione Binario-Decimale"

Si scriva un programma in linguaggio C che converta un numero binario in un numero decimale. Il numero binario è rappresentato su N bit, e il valore di N è inserito da tastiera. L'utente inserisce le cifre del numero binario un bit alla volta, partendo dal bit meno significativo (ossia dal bit di peso 2^0). Il programma visualizzerà il numero decimale corrispondente.

Suggerimento. Per calcolare le potenze di 2 utilizzare la funzione `pow`, includendo la libreria `math.h`. Ad esempio per calcolare 2^5, si scriverà `pow(2,5)`. In generale, data una base a, per calcolare $y = a^b$, si scrive `y = pow(a,b)` includendo la libreria `math.h`.

Soluzione

```
/* PROGRAMMAZIONE IN C */

/* File: bindec1.c */
/* Soluzione proposta esercizio "Conversione Binario-Decimale" */

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

#define BASE 2

int main(void)
{
    int N ;         /* numero di cifre binarie */
    int bit ;       /* cifra binaria */
    int peso ;      /* peso della cifra binaria */
    int numero ;    /* valore decimale del numero binario */

    /* INIZIALIZZA LE VARIABILI */
    peso = 0 ;   /* LA PRIMA CIFRA BINARIA LETTA AVRA' PESO ZERO */
    numero = 0 ; /* IL VALORE DECIMALE DEL NUMERO BINARIO E'
                    INIZIALIZZATO A ZERO */

    /* LEGGI IL NUMERO DI CIFRE BINARIE */
    printf("Immetti␣il␣numero␣di␣bit␣del␣numero␣binario:␣") ;
    scanf("%d", &N) ;

    /* LEGGI IL NUMERO BINARIO */
    printf("\nImmetti␣il␣numero␣binario␣partendo␣dal␣bit␣meno␣significativo␣\n") ;

    while ( peso < N )
    {
        /* LEGGI LA CIFRA BINARIA SUCCESSIVA */
        printf("Immetti␣la␣cifra␣binaria␣2^%d:", peso) ;
        scanf("%d", &bit) ;

        /* CALCOLA IL VALORE IN DECIMALE DELLA CIFRA BINARIA INSERITA
        E AGGIUNGILO ALLA CIFRA DECIMALE CALCOLATA FINO AD ORA*/
        numero = numero + bit * pow(BASE, peso) ;

        /* AGGIORNA IL PESO DELLA CIFRA BINARIA */
        peso = peso + 1 ;
    }

    /* STAMPA IL RISULTATO */
    printf("\n") ;
    printf("La␣cifra␣decimale␣calcolata␣e':␣%d\n", numero) ;
    exit(0) ;
}
```

Soluzione alternativa

Viene proposta una seconda soluzione, che non usa la funzione `pow` ma calcola la potenza mediante ripetute moltiplicazioni ed inoltre controlla se le cifre inserite sono corrette. Questa soluzione è "generalizzabile" facilmente ad altre basi pur di cambiare il valore della costante `BASE`.

```
1  /* PROGRAMMAZIONE IN C */
2
3  /* File: bindec2.c */
4  /* Soluzione proposta esercizio "Conversione Binario-Decimale" */
5  /* Versione 2 */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 #define BASE 2
11
12 int main(void)
13 {
14     int N ;         /* numero di cifre binarie */
15     int bit ;       /* cifra binaria */
16     int peso ;      /* peso della cifra binaria */
17     int potenza;    /* potenza BASE^peso */
18     int numero ;    /* valore decimale del numero binario */
19
20     /* INIZIALIZZA LE VARIABILI */
21     peso = 0 ;       /* LA PRIMA CIFRA BINARIA IMMESSA AVRA' PESO 0 */
22     numero = 0 ;     /* IL VALORE DECIMALE DEL NUMERO BINARIO E'
23                         INIZIALIZZATO A 0 */
24     potenza = 1 ;    /* POICHE' PESO=0, BASE^PESO E' UGUALE A 1 */
25
26     /* LEGGI IL NUMERO DI CIFRE BINARIE */
27     printf("Immetti␣il␣numero␣di␣bit␣del␣numero␣binario:␣") ;
28     scanf("%d", &N) ;
29
30     while ( peso < N )
31     {
32         /* LEGGI LA CIFRA BINARIA SUCCESSIVA */
33         printf("Immetti␣la␣cifra␣binaria␣2^%d:", peso) ;
34         scanf("%d", &bit) ;
35
36         /* CONTROLLA SE IL VALORE DELLA CIFRA BINARIA E' CORRETTO */
37         if( bit >= 0 && bit < BASE)
38         {
39           /* CALCOLA IL VALORE IN DECIMALE DELLA CIFRA BINARIA INSERITA
40           E AGGIUNGILO ALLA CIFRA DECIMALE CALCOLATA FINO AD ORA*/
41           numero = numero + bit*potenza ;
42
43            /* AGGIORNA IL PESO DELLA CIFRA BINARIA */
44             peso = peso + 1 ;
45
46             /* AGGIORNA LA POTENZA */
47             potenza = potenza * BASE ;
48         }
49         else
50             /* SE IL VALORE DELLA CIFRA BINARIA NON E' CORRETTO
51                STAMPA UN MESSAGGIO */
```

```
52              printf("Dato␣errato␣-␣reinseriscilo\n") ;
53      }
54
55      /* STAMPA IL RISULTATO */
56      printf("\n") ;
57      printf("La␣cifra␣decimale␣calcolata␣e':␣%d\n", numero) ;
58
59      exit(0) ;
60  }
```

3 Esercizio “Media dei numeri”

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.

Soluzione

```
1   /* PROGRAMMAZIONE IN C */
2
3   /* File: media_numeri.c */
4   /* Soluzione proposta esercizio "Media dei numeri" */
5
6   #include <stdio.h>
7   #include <stdlib.h>
8
9   int main(void)
10  {
11      int numero ;         /* numero inserito */
12      int conta ;          /* conta quanti numeri sono inseriti */
13      float somma ;        /* somma dei numeri inseriti */
14      float media ;        /* media dei numeri inseriti */
15
16      /* "somma" e "media" sono di tipo float per calcolare la media
17      come valore decimale con la virgola*/
18
19      /* INIZIALIZZA LE VARIABILI */
20      somma = 0 ;
21      conta = 0 ;
22
23      /* LEGGI UN NUMERO */
24      printf("Inserire␣una␣serie␣di␣numeri.␣La␣condizione␣di␣terminazione␣"
25             "e'␣il␣numero␣zero.\n") ;
26      printf("Inserisci␣numero:␣") ;
27      scanf ("%d", &numero) ;
28
29       /* LEGGI UNA SERIE DI NUMERI, FINO A QUANDO NON E' INSERITO IL NUMERO 0 */
30       while ( numero != 0 )
31       {
32          /* AGGIORNA LA SOMMA DEI NUMERI INSERITI */
33          somma = somma + numero ;
34
35          /* INCREMENTA IL CONTATORE DEI NUMERI INSERITI FINO AD ORA */
36          conta = conta + 1 ;
37
```

```
38          /* LEGGI UN NUMERO */
39          printf("Inserisci numero: ") ;
40          scanf ("%d", &numero);
41       }
42
43      /* CALCOLA LA MEDIA DEI NUMERI INSERITI */
44      media = somma/conta ;
45
46      /* STAMPA IL RISULTATO */
47      printf("\n") ;
48      printf("Numeri inseriti %d, Somma %f, Media %f \n", conta, somma, media);
49      exit(0) ;
50  }
```

4 Esercizio “Massimo e minimo”

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.

Soluzione

```
1   /* PROGRAMMAZIONE IN C */
2
3   /* File: massimo_minimo.c */
4   /* Soluzione proposta esercizio "Massimo e minimo" */
5
6   #include <stdio.h>
7   #include <stdlib.h>
8
9   int main(void)
10  {
11      int numero ;        /* numero inserito */
12      int N ;             /* quanti numeri saranno inseriti */
13      int max, min ;      /* valore massimo e minimo tra i numeri inseriti */
14
15      /* LEGGI QUANTI NUMERI SARANNO INSERITI */
16      printf("Indica quanti numeri saranno inseriti: ") ;
17      scanf("%d", &N) ;
18
19      /* VERIFICA CHE LA SEQUENZA INSERITA CONTENGA ALMENO UN NUMERO*/
20       if ( N <= 0 )
21           printf("Errore: non sara' inserito nessun numero \n") ;
22       else
23       {
24           /* LEGGI UN NUMERO */
25           printf("Inserisci un numero: ") ;
26           scanf ("%d", &numero) ;
27
28           /* N VIENE DECREMENTATO POICHE' E' STATO INSERITO UN NUMERO */
29           N = N - 1 ;
30
31           /* INIZIALIZZA "max" e "min" CON IL PRIMO NUMERO INSERITO */
32           max = numero ;
33           min = numero ;
34
```

```
35          /* LEGGI GLI ALTRI NUMERI DELLA SEQUENZA */
36          while ( N > 0 )
37          {
38               /* LEGGI UN NUMERO */
39               printf("Inserisci un numero: ") ;
40               scanf ("%d", &numero) ;
41
42               /* AGGIORNA IL VALORE MASSIMO "max" */
43               if ( numero > max )
44                  max = numero ;
45               else
46               {
47                  /* AGGIORNA IL VALORE MINIMO "min" */
48                  if ( numero < min )
49                     min = numero ;
50               }
51
52              /* N VIENE DECREMENTATO POICHE' E' STATO INSERITO UN NUMERO */
53              N = N - 1 ;
54          }
55
56          /* STAMPA IL RISULTATO */
57          printf("\n") ;
58          printf("Valore massimo %d, Valore minimo %d\n", max, min) ;
59      }
60      exit(0) ;
61  }
```

5 Esercizio “Quadrati perfetti”

Si scriva un programma in linguaggio C per il calcolo dei quadrati perfetti per una sequenza di numeri. Il programma deve prima leggere un numero inserito da tastiera, e quindi stampare i primi quadrati perfetti sino al quadrato del numero.

Soluzione

```
1  /* PROGRAMMAZIONE IN C */
2
3  /* File: quadrati_perfetti.c */
4  /* Soluzione proposta esercizio "Quadrati perfetti" */
5
6  #include <stdio.h>
7  #include <stdlib.h>
8  #include <math.h>
9
10 int main(void)
11 {
12     int numero_finale ; /* numero inserito */
13     int N ;             /* numero compreso tra 0 e "numero_finale" */
14     int quadrato ;      /* quadrato del numero "N" */
15
16     /* LEGGI UN NUMERO */
17     printf("Inserisci un numero intero e positivo: ") ;
18     scanf("%d", &numero_finale) ;
19
20     /* VERIFICA CHE IL NUMERO INSERITO SIA POSITIVO */
```

```
21      if ( numero_finale < 0 )
22          printf("Errore: il numero deve essere positivo\n") ;
23      else
24      {
25          /* INIZIALIZZA IL NUMERO "N" CON IL VALORE 0 */
26          N = 0 ;
27
28          /* CONSIDERA TUTTI I NUMERI TRA 0 E "numero_finale"
29          E PER OGNI NUMERO CALCOLA IL QUADRATO */
30          while ( N <= numero_finale )
31          {
32              /* CALCOLA IL QUADRATO DEL NUMERO "N" */
33              quadrato = pow(N,2) ;
34
35              /* IN ALTERNTIVA E' POSSIBILE CALCOLARE IL
36              QUADRATO di "N" COME quadrato = N * N ; */
37
38              /* STAMPA IL RISULTATO */
39              printf("\n") ;
40              printf("Numero %d, Quadrato %d\n", N, quadrato) ;
41
42              /* INCREMENTA IL VALORE DEL NUMERO "N" */
43              N = N + 1 ;
44          }
45      }
46      exit(0) ;
47  }
```

6 Esercizio: “Fattoriale”

Si scriva un programma in linguaggio C che acquisisca un numero intero positivo N da tastiera e stampi il valore del fattoriale di N.
Suggerimento. Si ricorda che il fattoriale di un numero è il prodotto di tutti i numeri compresi tra 1 ed N.

$$N! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot (N-1) \cdot N$$

Inoltre $0! = 1$.

Soluzione

```
1   /* PROGRAMMAZIONE IN C */
2
3   /* File: fattoriale.c */
4   /* Soluzione proposta esercizio "Fattoriale" */
5
6   #include <stdio.h>
7   #include <stdlib.h>
8
9   int main(void)
10  {
11      int N ;             /* numero inserito */
12      int fattoriale ;    /* fattoriale del numero */
13
14      /* LEGGI UN NUMERO */
15      printf("Inserisci un numero intero positivo: ") ;
16      scanf("%d", &N) ;
```

```
17
18      /* VERIFICA CHE IL NUMERO INSERITO SIA POSITIVO */
19      if ( N < 0 )
20          printf("Errore: il numero inserito deve essere positivo\n") ;
21      else
22      {
23          /* INIZIALIZZA IL VALORE DEL FATTORIALE */
24          fattoriale = 1 ;
25
26          /* IL FATTORIALE E' CALCOLATO COME PRODOTTO
27            TRA TUTTI I NUMERI COMPRESI TRA "N" E 1 */
28          while( N > 1 )
29          {
30                  /* AGGIORNA IL VALORE DEL FATTORIALE */
31                  fattoriale = fattoriale * N ;
32
33                  /* DECREMENTA IL VALORE DI "N" */
34                  N = N - 1 ;
35          }
36
37          /* STAMPA IL RISULTATO */
38          printf("\n") ;
39          printf("Il fattoriale e' %d\n", fattoriale) ;
40       }
41       exit(0) ;
42  }
```

7 Esercizio: “Classificazione di sequenze”

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.

Suggerimento. Una sequenza è crescente se ogni numero è maggiore del precedente, decrescente se ogni numero è minore del precedente, né crescente né decrescente in tutti gli altri casi.

Soluzione

```
1   /* PROGRAMMAZIONE IN C */
2
3   /* File: sequenzanumeri.c */
4   /* Soluzione proposta esercizio "Classificazione di sequenze" */
5
6   #include <stdio.h>
7   #include <stdlib.h>
8
9   int main(void)
10  {
11      int totale ;                    /* quanti numeri saranno inseriti */
12      int numero ;                    /* ultimo numero inserito */
13      int numero_precedente ;         /* penultimo numero inserito */
```

```
14      int N ;                          /* contatore per scandire i
15                                          numeri della sequenza */
16      int positivi, negativi, nulli; /* contatori numeri postivi, negativi,
17                                          o nulli */
18      int pari, dispari;               /* contatori numeri pari o dispari */
19      int crescente, decrescente ;     /* flag per indicare se la sequenza e'
20                                          crescente o decrescente */
21
22      /* LEGGI QUANTI NUMERI SARANNO INSERITI */
23      printf("Quanti␣numeri␣saranno␣inseriti?␣") ;
24      scanf("%d", &totale) ;
25
26      /* INIZIALIZZA A ZERO I CONTATORI DI NUMERI POSITIVI, NEGATIVI, NULLI,
27      PARI E DIPARI */
28      positivi = 0 ;
29      negativi = 0 ;
30      nulli = 0 ;
31      pari = 0 ;
32      dispari = 0 ;
33
34      /* INIZIALIZZA I FLAG PER INDICARE SE LA SEQUENZA E' CRESCENTE O DECRESCENTE
35      -- SE "crescente" E' UGUALE a 1: SEQUENZA CRESCENTE
36      -- SE "crescente" E' UGUALE a 0: SEQUENZA NON CRESCENTE
37      -- SE "decrescente" E' UGUALE a 1: SEQUENZA DECRESCENTE
38      -- SE "decrescente" E' UGUALE a 0: SEQUENZA NON DECRESCENTE
39      INIZIALIZZA AD 1 ENTRAMBI I FLAG. ALL'INTERNO DEL CICLO WHILE
40      ASSEGNA I FLAG A 0 SE VERIFICHI CHE LA SEQUENZA NON E' CRESCENTE O
41      NON E' DECRESCENTE */
42      crescente = 1 ;
43      decrescente = 1 ;
44
45      /* INIZIALIZZA IL CONTATORE DEI NUMERI GIA' INSERITI */
46      N = 0 ;
47
48      /* RIPETI IL SEGUENTE CICLO FINO A QUANDO NON SONO STATI INSERITI TUTTI
49      I NUMERI DELLA SEQUENZA */
50      while( N < totale )
51      {
52           /* LEGGI UN NUMERO */
53           printf("Inserisci␣il␣numero␣%d:␣", N+1) ;
54           scanf("%d", &numero) ;
55
56           /* SE IL NUMERO E' UGUALE A ZERO INCREMENTA IL CONTATORE "nulli" */
57           if ( numero == 0 )
58                  nulli = nulli + 1 ;
59           else
60           {
61                  /* IL NUMERO E' DIVERSO DA ZERO. SE NUMERO E' POSITIVO
62                  INCREMENTA IL CONTATORE "positivi" ALTRIMENTI INCREMENTA
63                  IL CONTATORE "negativi" */
64                  if ( numero > 0 )
65                    positivi = positivi + 1 ;
66                  else
67                    negativi = negativi + 1 ;
68           }
69
```

```
70          /* SE IL NUMERO E' PARI INCREMENTA IL CONTATORE "pari"
71          ALTRIMENTI INCREMENTA IL CONTATORE "dispari" */
72          if ( numero % 2 == 0 )
73                  pari = pari + 1 ;
74          else
75                  dispari = dispari + 1 ;
76
77          /* PER VERIFICARE SE LA SEQUENZA E' CRESCENTE O DECRESENTE
78          CONFRONTA IL NUMERO CORRENTE CON IL PENULTIMO NUMERO INSERITO.
79          LA VERIFICA PUO' ESSERE FATTA SOLO QUANDO SONO STATI INSERITI
80          ALMENO DUE NUMERI DELLA SEQUENZA, OSSIA N>1. INFATTI,
81          N==0 QUANDO VIENE INSERITO IL PRIMO NUMERO E N==1 QUANDO VIENE
82          INSERITO IL SECONDO NUMERO */
83
84          if ( N > 1 )
85          {
86                  /* SE IL NUMERO CORRENTE E' MAGGIORE DEL PRECEDENTE LA
87                  SEQUENZA NON E' DECRESCENTE */
88                  if ( numero > numero_precedente )
89                       decrescente=0;
90                  else
91                  {
92                       /* SE IL NUMERO CORRENTE E' MINORE DEL PRECEDENTE LA
93                       SEQUENZA NON E' CRESCENTE */
94                       if (numero < numero_precedente)
95                            crescente=0;
96                       else
97                       {
98                       /* SE IL NUMERO CORRENTE E' UGUALE AL PRECEDENTE LA
99                       SEQUENZA NON E' STRETTAMENTE CRESCENTE NE'
100                      STRETTAMENTE DECRESCENTE */
101                      crescente=0;
102                      decrescente=0;
103                      }
104                 }
105         }
106
107         /* IL NUMERO CORRENTE SARA' IL PENULTIMO NUMERO INSERITO NELLA PROSSIMA
108         ITERAZIONE DEL CICLO */
109         numero_precedente=numero;
110
111         /* INCREMENTA IL CONTATORE DEI NUMERI INSERITI */
112         N = N + 1 ;
113     }
114
115     /* STAMPA IL RISULTATO */
116     printf("Hai␣inserito:␣%d␣positivi,␣%d␣negativi,␣%d␣uguali␣a␣zero\n",
117     positivi, negativi, nulli) ;
118
119     printf("Hai␣inserito:␣%d␣numeri␣pari␣e␣%d␣numeri␣dispari\n",
120            pari, dispari) ;
121
122     if ( crescente == 1 )
123         printf("La␣sequenza␣e'␣crescente\n") ;
124     else
125     {
```

```
126         if ( decrescente == 1 )
127                 printf("La␣sequenza␣e'␣decrescente\n") ;
128         else
129                 printf("La␣sequenza␣non␣e'␣ne`␣crescente␣ne`␣decrescente\n") ;
130     }
131
132     exit(0) ;
133 }
```

8 Esercizio: "Divisori di un numero"

Sia dato un numero intero positivo N inserito da tastiera. Si scriva un programma in linguaggio C che calcoli i numeri interi che sono divisori (con resto uguale a zero) di N. Dire inoltre se N è un numero primo. *Suggerimento.*

- Un numero M è divisore di un numero N se il resto della divisione N/M è uguale a zero.
- Un numero è primo se è divisibile solo per 1 o per il numero stesso.

Soluzione

```
1  /* PROGRAMMAZIONE IN C */
2
3  /* File: divisori.c */
4  /* Soluzione proposta esercizio "Divisori di un numero" */
5
6  #include <stdio.h>
7  #include <stdlib.h>
8
9  int main(void)
10 {
11     int numero ;   /* numero inserito */
12     int divisore ; /* divisore del numero. E' un contatore per scandire
13                       tutti i valori tra 1 e "numero"  */
14     int primo ;    /* flag per indicare se il numero inserito e' primo */
15
16     /* LEGGI UN NUMERO */
17     printf("Inserisci␣un␣numero␣intero␣positivo:␣") ;
18     scanf("%d", &numero) ;
19
20     /* CONTROLLA SE IL NUMERO E' POSITIVO */
21     if ( numero <= 0 )
22         printf("Errore:␣hai␣inserito␣un␣numero␣nullo␣o␣negativo\n") ;
23     else
24     {
25             /* PER CALCOLARE I DIVISORI CONSIDERA
26             TUTTI I NUMERI COMPRESI TRA 1 E "numero" */
27             divisore=1 ;
28
29            /* INIZIALIZZA IL FLAG "primo":
30             -- SE "primo" E' UGUALE a 1: "numero" E' PRIMO
31             -- SE "primo" E' UGUALE A 0: "numero" NON E' PRIMO.
32            IPOTIZZA CHE  "numero" SIA PRIMO ED INIZIALIZZA primo=1.
33            ALL'INTERNO DEL CICLO ASSEGNA primo=0 SE VERIFICHI CHE
34            "numero" NON E' PRIMO (OSSIA SE E' DIVISIBILE CON RESTO ZERO
35            ALMENO PER UN VALORE DIVERSO DA 1 E DA "numero") */
```

```
36              primo = 1 ;
37
38              /* IL CICLO ANALIZZA TUTTI I VALORI DI "divisore"
39              COMPRESI TRA 1 E "numero" */
40              while ( divisore <= numero )
41              {
42                   /* VERIFICA SE IL RESTO DELLA DIVISIONE E' UGUALE A ZERO */
43                   if ( numero%divisore == 0 )
44                   {
45                       /* STAMPA IL RISULTATO */
46                       printf("%d e' divisore di %d - Risultato divisione: %d \n",
47                              divisore, numero, numero/divisore) ;
48
49                       /* SE "divisore" E' DIVERSO SIA DA 1 CHE DA "NUMERO"
50                       ALLORA "numero" NON E' PRIMO*/
51                       if ( divisore != 1 && divisore != numero )
52                          primo=0;
53                   }
54
55                   /* INCREMENTA IL VALORE DEL POSSIBILE DIVISORE DI "numero" */
56                   divisore = divisore + 1 ;
57              }
58         }
59
60         /* STAMPA IL RISULTATO */
61         if ( primo == 1 )
62             printf("%d e' un numero primo \n", numero) ;
63         else
64             printf("%d non e' un numero primo \n", numero) ;
65
66         exit(0) ;
67   }
```

9 Esercizio: “Massimo comune divisore di 2 numeri”

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.
Suggerimento. Si considerino due numeri interi N1 e N2. Il MCD di N1 e N2 è il massimo tra i numeri che sono divisori (con resto uguale a zero) sia di N2 che di N1. In particolare, si supponga che sia N1 minore di N2. Il MCD è il massimo tra i numeri compresi tra 1 e N1 che sono divisori (con resto uguale a zero) sia di N1 che di N2.

Soluzione

```
1  /* PROGRAMMAZIONE IN C */
2
3  /* File: massimo_comun_divisore.c */
4  /* Soluzione proposta esercizio "Massimo comune divisore di 2 numeri" */
5
6  #include <stdio.h>
7  #include <stdlib.h>
8
9  int main(void)
10 {
11     int numero1, numero2 ;   /* numeri inseriti */
```

```
12      int minimo ;              /* valore minimo tra numero1 e numero2 */
13      int divisore ;            /* divisore del numero. E' un contatore per
14                                   scandire tutti i valori tra 1 e "minimo" */
15      int mcd ;                 /* massimo comun divisore */
16
17      /* LEGGI I DUE NUMERI */
18      printf("Inserisci␣il␣primo␣numero:␣") ;
19      scanf("%d", &numero1) ;
20
21      printf("Inserisci␣il␣secondo␣numero:␣") ;
22      scanf("%d", &numero2) ;
23
24      /* CONTROLLA SE ENTRAMBI I NUMERI SONO POSITIVI */
25      if ( numero1 <= 0 || numero2 <= 0 )
26          printf("Errore:␣hai␣inserito␣un␣numero␣nullo␣o␣negativo\n") ;
27      else
28      {
29          /* CALCOLA IL VALORE INFERIORE TRA I DUE NUMERI INSERITI*/
30          if ( numero1 < numero2 )
31              minimo = numero1 ;
32          else
33              minimo = numero2 ;
34
35          /* PER CALCOLARE IL MASSIMO COMUN DIVISORE CONSIDERA
36          TUTTI I NUMERI COMPRESI TRA 1 E "minimo". IL MASSIMO COMUN DIVISORE
37          E' IL MASSIMO TRA I VALORI COMPRESI TRA 1 e "minimo" CHE E' DIVISORE
38          SIA DI "numero1" CHE DI "numero2" */
39          divisore=1;
40          mcd=1;
41
42          while ( divisore <= minimo )
43          {
44              /* VERIFICA SE IL NUMERO RAPPRESENTATO IN "divisore"
45              E' DIVISORE, CON RESTO UGUALE A 0, SIA DI "numero1" CHE
46              DI "numero2" */
47              if ( numero1%divisore == 0  && numero2%divisore == 0 )
48              {
49                  /* POICHE' IL RESTO E' UGUALE A 0, IL VALORE DI "divisore"
50                  E' UN POSSIBILE MASSIMO COMUN DIVISORE. AGGIORNA IL VALORE
51                  DEL MASSIMO COMUM DIVISORE */
52                  mcd = divisore ;
53                  printf("%d␣e'␣divisore␣\n", mcd) ;
54              }
55              /* INCREMENTA IL VALORE DI "divisore" */
56              divisore = divisore + 1 ;
57          }
58
59          /* STAMPA IL RISULTATO */
60          printf("\n") ;
61          printf("Il␣massimo␣comun␣divisore␣per␣i␣numeri␣%d␣e␣%d␣e'␣%d\n",
62                 numero1, numero2, mcd) ;
63      }
64      exit(0) ;
65  }
```

10 Esercizio: “Minimo comune multiplo di 2 numeri”

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.
Suggerimento. Si considerino due numeri interi N1 e N2. Sia N1 più grande di N2. Il MCM è il primo multiplo di N1 che è divisibile (con resto uguale a zero) per N2.

Soluzione

```c
/* PROGRAMMAZIONE IN C */

/* File: minimo_comune_multiplo.c */
/* Soluzione proposta esercizio "Minimo comune multiplo di 2 numeri" */

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

int main(void)
{
    int numero1, numero2 ; /* numeri inseriti */
    int massimo, minimo ;  /* valore massimo e minimo tra numero1 e numero2 */
    int conta ;            /* contatore per generare i multipli di "massimo" */
    int fine ;             /* flag per indicare che e' stato trovato
                              il minimo comune multiplo */
    int mcm ;              /* valore del minimo comune multiplo */

    /* LEGGI I DUE NUMERI */
    printf("Inserisci␣il␣primo␣numero:␣") ;
    scanf("%d", &numero1) ;

    printf("Inserisci␣il␣secondo␣numero:␣") ;
    scanf("%d", &numero2) ;

    /* CONTROLLA SE ENTRAMBI I NUMERI SONO POSITIVI */
    if ( numero1<=0 || numero2<=0 )
        printf("Errore:␣hai␣inserito␣un␣numero␣nullo␣o␣negativo\n") ;
    else
    {
        /* CALCOLA IL VALORE MAGGIORE E INFERIORE TRA I DUE NUMERI INSERITI*/
        if ( numero1 > numero2 )
        {
            massimo = numero1 ;
            minimo = numero2 ;
        }
        else
        {
            massimo = numero2 ;
            minimo = numero1 ;
        }

        /* INIZIALIZZA "conta" e "mcm" */
        conta=1;
        mcm=0;

        /* INIZIALIZZA IL FLAG "fine" A 0. LA RICERCA TERMINA QUANDO "fine"
```

```
47          ASSUME IL VALORE 1 */
48          fine = 0 ;
49
50          while ( fine == 0 )
51          {
52              /* CALCOLA IL SUCCESSIVO MULTIPLO DI "massimo". QUESTO VALORE E'
53              UN CANDIDATO MINIMO COMUNE MULTIPLO */
54              mcm = conta * massimo ;
55
56              /* VERIFICA SE "minimo" E' DIVISORE DI "mcm" */
57              if ( mcm % minimo == 0 )
58              {
59                  /* LA RICERCA E' TERMINATA. AGGIORNA IL FLAG "fine" */
60                  fine = 1 ;
61              }
62              else
63              {
64                  /* INCREMENTA LA VARIABILE "conta" */
65                  conta = conta + 1 ;
66              }
67          }
68
69          /* STAMPA IL RISULTATO */
70          printf("\n") ;
71          printf("Il␣MCM␣per␣%d␣e␣%d␣e'␣%d*%d=%d\n",
72                 numero1, numero2, conta, massimo, mcm);
73
74      }
75      exit(0) ;
76  }
```

11 Esercizio: “Disegno figure geometriche”

1. Si realizzi un programma in linguaggio C che legga un numero intero N e visualizzi un quadrato di asterischi di lato N (vedi esempio con N = 5).
2. Si realizzi una variante del programma per visualizzare solo i lati del quadrato (vedi esempio con N = 5).
3. Si realizzi una variante del programma per visualizzare un triangolo isoscele rettangolo di lato N (vedi esempio con N = 5).
4. Si realizzi una variante del programma per visualizzare un quadrato di lato N come nell’esempio del caso 4 (con N = 5).

```
Caso 1      Caso 2      Caso 3      Caso 4

*****       *****       *           *++++
*****       *   *       **          **+++
*****       *   *       ***         ***++
*****       *   *       ****        ****+
*****       *****       *****       *****
```

Soluzione Caso 1

```c
/* PROGRAMMAZIONE IN C */

/* File: quadasterisco.c */
/* Soluzione proposta esercizio "Disegno figure geometriche (Caso 1)" */

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

int main(void)
{
    int lato ;           /* lato del quadrato */
    int riga, colonna ;  /* riga e colonna del quadrato */

    /* LEGGI IL LATO DEL QUADRATO */
    printf("Inserisci il lato del quadrato: ") ;
    scanf("%d",&lato) ;

    /* CONTROLLA SE IL LATO DEL QUADRATO E' UN NUMERO MAGGIORE DI 0 */
    if ( lato <= 0 )
        printf("Errore, il lato deve essere maggiore di zero\n") ;
    else
    {
        /* IL CICLO PIU' ESTERNO SCANDISCE LA RIGHE DEL QUADRATO */

        /* INIZIALIZZA LA VARIABILE PER LA SCANSIONE DELLE RIGHE DEL QUADRATO */
        riga = 0 ;

        while ( riga < lato )
        {
            /* IL CICLO PIU' INTERNO SCANDISCE LE COLONNE DEL QUADRATO */
            /* PER OGNI RIGA STAMPA "*" PER OGNI COLONNA */

            /*INIZIALIZZA LA VARIABILE PER LA SCANSIONE DELLE COLONNE
            DEL QUADRATO */
            colonna = 0 ;

            while ( colonna < lato )
            {
                /* STAMPA "*" senza andare a capo */
                printf("*") ;

                /* INCREMENTA "colonna" PER PASSARE ALLA COLONNA SUCCESSIVA */
                colonna = colonna + 1 ;
            }

            /* TERMINATA LA STAMPA DI UNA RIGA SI DEVE RIPORTARE IL CURSORE
            AL MARGINE SINISTRO DELLO SCHERMO */
            printf("\n");

            /* INCREMENTA "riga" PER PASSARE ALLA RIGA SUCCESSIVA */
            riga = riga + 1 ;
        }
    }
    exit(0) ;
}
```

Soluzione Caso 2

```c
/* PROGRAMMAZIONE IN C */

/* File: quadasterisco2.c */
/* Soluzione proposta esercizio "Disegno figure geometriche (Caso 2)" */

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

int main(void)
{
    int lato ;           /* lato del quadrato */
    int riga, colonna ;  /* riga e colonna del quadrato */

    /* LEGGI IL LATO DEL QUADRATO */
    printf("Inserisci il lato del quadrato: ") ;
    scanf("%d",&lato) ;

    /* CONTROLLA SE IL LATO DEL QUADRATO E' UN NUMERO MAGGIORE DI 0 */
    if ( lato <= 0 )
        printf("Errore, il lato deve essere maggiore di zero\n") ;
    else
    {
        /* IL CICLO PIU' ESTERNO SCANDISCE LA RIGHE DEL QUADRATO */

        /* INIZIALIZZA LA VARIABILE PER LA SCANSIONE DELLE RIGHE DEL QUADRATO */
        riga = 0 ;

        while ( riga < lato )
        {
            /* IL CICLO PIU' INTERNO SCANDISCE LE COLONNE DEL QUADRATO */

            /*INIZIALIZZA LA VARIABILE PER LA SCANSIONE DELLE COLONNE
            DEL QUADRATO */
            colonna = 0 ;

            while ( colonna < lato )
            {
                /* PER LA PRIMA E L'ULTIMA RIGA STAMPA "*" PER OGNI COLONNA */
                if ( riga == 0 || riga == (lato-1) )
                    printf("*") ;
                else
                {
                    /* PER LE ALTRE RIGHE STAMPA "*" SOLO PER LA PRIMA
                    E L'ULTIMA COLONNA */
                    if ( colonna == 0 || colonna == (lato-1) )
                        printf("*") ;
                    else
                        /* IN TUTTI GLI ALTRI CASI STAMPA UNO SPAZIO*/
                        printf(" ") ;
                }

                /* INCREMENTA "colonna" PER PASSARE ALLA COLONNA SUCCESSIVA */
                colonna = colonna + 1 ;
            }

```

```
56              /* TERMINATA LA STAMPA DI UNA RIGA SI DEVE RIPORTARE IL CURSORE
57              AL MARGINE SINISTRO DELLO SCHERMO */
58              printf("\n") ;
59
60              /* INCREMENTA "riga" PER PASSARE ALLA RIGA SUCCESSIVA */
61              riga = riga + 1 ;
62          }
63      }
64      exit(0) ;
65  }
```

Soluzione Caso 3

```
1   /* PROGRAMMAZIONE IN C */
2
3   /* File: triangasterisco.c */
4   /* Soluzione proposta esercizio "Disegno figure geometriche (Caso 3)" */
5
6   #include <stdio.h>
7   #include <stdlib.h>
8
9   int main(void)
10  {
11      int lato ;           /* lato del triangolo */
12      int riga, colonna ;  /* riga e colonna del triangolo */
13
14      /* LEGGI IL LATO DEL TRIANGOLO */
15      printf("Inserisci␣il␣lato␣del␣triangolo:␣") ;
16      scanf("%d", &lato) ;
17
18      /* CONTROLLA SE IL LATO DEL TRIANGOLO E' UN NUMERO MAGGIORE DI 0 */
19      if ( lato <=0 )
20          printf("Errore,␣il␣lato␣deve␣essere␣maggiore␣di␣zero\n") ;
21      else
22      {
23          /* IL CICLO PIU' ESTERNO SCANDISCE LA RIGHE DEL TRIANGOLO */
24
25          /* INIZIALIZZA LA VARIABILE PER LA SCANSIONE DELLE RIGHE DEL
26          TRIANGOLO */
27          riga = 0 ;
28
29          while ( riga < lato )
30          {
31              /* IL CICLO PIU' INTERNO SCANDISCE LE COLONNE DEL TRIANGOLO */
32              /* PER OGNI RIGA STAMPA "*" SOLO SE  colonna <= riga */
33
34              /*INIZIALIZZA LA VARIABILE PER LA SCANSIONE DELLE COLONNE DEL
35              TRIANGOLO */
36              colonna = 0 ;
37
38              while ( colonna <= riga )
39              {
40                  /* STAMPA "*" senza andare a capo */
41                  printf("*") ;
42
43                  /* INCREMENTA "colonna" PER PASSARE ALLA COLONNA SUCCESSIVA */
44                  colonna = colonna + 1 ;
```

```
45              }
46
47              /* TERMINATA LA STAMPA DI UNA RIGA SI DEVE RIPORTARE IL CURSORE
48              AL MARGINE SINISTRO DELLO SCHERMO */
49              printf("\n") ;
50
51              /* INCREMENTA "riga" PER PASSARE ALLA RIGA SUCCESSIVA */
52              riga = riga + 1 ;
53          }
54      }
55      exit(0) ;
56  }
```

Soluzione Caso 4

```
1   /* PROGRAMMAZIONE IN C */
2
3   /* File: quadasterisco3.c */
4   /* Soluzione PROPOSTA esercizio "Disegno figure geometriche (Caso 4)" */
5
6   #include <stdio.h>
7   #include <stdlib.h>
8
9   int main(void)
10  {
11      int lato ;           /* lato del quadrato */
12      int riga, colonna ;  /* riga e colonna del quadrato */
13
14      /* LEGGI IL LATO DEL QUADRATO */
15      printf("Inserisci␣il␣lato␣del␣quadrato:␣") ;
16      scanf("%d",&lato) ;
17
18      /* CONTROLLA SE IL LATO DEL QUADRATO E' UN NUMERO MAGGIORE DI 0 */
19      if ( lato <= 0 )
20          printf("Errore,␣il␣lato␣deve␣essere␣maggiore␣di␣zero\n") ;
21      else
22      {
23          /* IL CICLO PIU' ESTERNO SCANDISCE LA RIGHE DEL QUADRATO */
24
25          /* INIZIALIZZA LA VARIABILE PER LA SCANSIONE DELLE RIGHE DEL QUADRATO */
26          riga = 0 ;
27
28          while ( riga < lato )
29          {
30              /* IL CICLO PIU' INTERNO SCANDISCE LE COLONNE DEL QUADRATO */
31
32              /*INIZIALIZZA LA VARIABILE PER LA SCANSIONE DELLE COLONNE
33              DEL QUADRATO */
34              colonna = 0 ;
35
36              while ( colonna < lato )
37              {
38                  /* SE colonna <= riga  STAMPA "*" ALTRIMENTI STAMPA "+" */
39                  if ( colonna <= riga )
40                      printf("*") ;
41                  else
42                      printf("+") ;
```

```
43
44                  /* INCREMENTA "colonna" PER PASSARE ALLA COLONNA SUCCESSIVA */
45                  colonna = colonna + 1 ;
46              }
47
48              /* TERMINATA LA STAMPA DI UNA RIGA SI DEVE RIPORTARE IL CURSORE
49              AL MARGINE SINISTRO DELLO SCHERMO */
50              printf("\n") ;
51
52              /* INCREMENTA "riga" PER PASSARE ALLA RIGA SUCCESSIVA */
53              riga = riga + 1 ;
54          }
55      }
56      exit(0) ;
57  }
```

12 Esercizio: “Rappresentazione del triangolo di Floyd”

Scrivere un programma in linguaggio C per la rappresentazione del triangolo di Floyd. Il programma riceve da tastiera un numero interno N. Il programma visualizza le prima N righe del triangolo di Floyd.

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

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

Suggerimento. Si osserva che il numero di valori in ogni riga corrisponde all’indice della riga: 1 valore sulla prima riga, 2 sulla seconda, 3 sulla terza.

Soluzione

```
1  /* PROGRAMMAZIONE IN C */
2
3  /* File: TriangoloFloyd.c */
4  /* Soluzione proposta esercizio "Rappresentazione del triangolo di Floyd" */
5
6  #include <stdio.h>
7  #include <stdlib.h>
8
9  int main(void)
10 {
11     int numero ;         /* numero inserito */
12     int riga, colonna ;  /* riga e colonna del triangolo */
13     int cifra ;          /* numero da stampare nel triangolo di Floyd */
14
15     /* LEGGI UN NUMERO */
16     printf("Inserisci␣il␣numero␣") ;
17     scanf("%d",&numero) ;
18
19     /* CONTROLLA SE IL NUMERO E' MAGGIORE DI 0 */
20     if ( numero <=0 )
21         printf("Errore,␣il␣lato␣deve␣essere␣maggiore␣di␣zero\n") ;
22     else
23     {
```

```
24          /* IL CICLO PIU' ESTERNO SCANDISCE LA RIGHE DEL TRIANGOLO */
25
26          /* INIZIALIZZA LA VARIABILE PER LA SCANSIONE DELLE RIGHE DEL
27          TRIANGOLO */
28          riga = 0 ;
29
30          /* LA PRIMA CIFRA DA STAMPARE NEL TRIANGOLO E' 1 */
31          cifra=1;
32
33          while ( riga < numero )
34          {
35              /* IL CICLO PIU' INTERNO SCANDISCE LE COLONNE DEL TRIANGOLO */
36              /* PER OGNI RIGA STAMPA IL VALORE IN "cifra" SOLO SE
37              colonna <= riga */
38
39              /*INIZIALIZZA LA VARIABILE PER LA SCANSIONE DELLE COLONNE DEL
40              TRIANGOLO */
41              colonna = 0 ;
42
43              while ( colonna <= riga )
44              {
45                  /* STAMPA "cifra" */
46                  printf("%d␣", cifra) ;
47
48                  /* INCREMENTA "colonna" PER PASSARE ALLA COLONNA SUCCESSIVA */
49                  colonna = colonna + 1 ;
50
51                  /* INCREMENTA "cifra" */
52                  cifra=cifra+1;
53              }
54
55              /* TERMINATA LA STAMPA DI UNA RIGA SI DEVE RIPORTARE IL CURSORE
56              AL MARGINE SINISTRO DELLO SCHERMO */
57              printf("\n") ;
58
59              /* INCREMENTA "riga" PER PASSARE ALLA RIGA SUCCESSIVA */
60              riga = riga + 1 ;
61          }
62      }
63      exit(0) ;
64  }
```

13 Esercizio: "Calcolo dell'opposto di un numero binario rappresentato in complemento a 2 su N bit"

Scrivere un programma in linguaggio C che riceva in ingresso un numero binario rappresentato in complemento a 2 su N bit. Inizialmente l'utente inserisce il numero N di bit. Quindi inserisce le cifre del numero binario un bit alla volta, partendo dal bit meno significativo. Il programma calcola l'opposto del numero binario ricevuto in ingresso. Tale numero sarà visualizzato partendo dalla cifra meno significativa.
Suggerimento. Per poter effettuare il calcolo del risultato, utilizzare il metodo secondo il quale si considerano le cifre del numero binario in complemento a due a partire dalla meno significativa alla più significativa (ossia da destra verso sinistra). Si ricopiano in uscita tutti gli zeri fino al primo 1 compreso. Dopo si invertono i restanti bit.

Soluzione

```c
/* PROGRAMMAZIONE IN C */

/* File: opposto_ca2.c */
/* Soluzione proposta esercizio "Calcolo dell'opposto di un numero binario
rappresentato in complemento a 2 su N bit" */

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

int main(void)
{
    int N ;          /* numero di cifre del numero binario */
    int bit ;        /* cifra binaria del numero binario */
    int opposto ;    /* cifra binaria dell'opposto del numero binario */
    int inverti ;    /* flag per indicare se le cifre binarie devono essere
                        invertite */
    int num_bits ;   /* contatore per scandire le cifre binarie */

    /* LEGGI IL NUMERO DI CIFRE BINARIE */
    printf("Quanti bit saranno inseriti? ") ;
    scanf("%d", &N) ;

    /* INIZIALIZZA IL FLAG "inverti":
    -- SE "inverti" E' UGUALE a 1: si invertono tutti i bit inseriti
       successivamente
    -- SE "inverti" E' UGUALE A 0: si ricopiano in uscita i bit inseriti
       successivamente
    "inverti" E' INIZIALIZZATO A 0 ED ASSEGNATO A 1 QUANDO VIENE INSERITO
    IL PRIMO BIT UGUALE A 1 */
    inverti = 0 ;

    /* LEGGI LE CIFRE DEL NUMERO BINARIO A PARTIRE DAL BIT MENO SIGNIFICATIVO */
    printf("Inserisci il numero binario dal bit meno significativo\n");

    /* INIZIALIZZA "num_bits" A 0*/
    num_bits = 0 ;

    while ( num_bits < N )
    {
        /* LEGGI LA CIFRA BINARIA */
        printf("Inserisci il bit di peso %d: ", num_bits) ;
        scanf("%d", &bit) ;

        /* CALCOLA IL VALORE OPPOSTO */
        if ( inverti == 0 )
        {
            /* RICOPIA IN USCITA LA CIFRA BINARIA INSERITA */
            opposto = bit ;

            /* SE HAI TROVATO LA PRIMA CIFRA BINARIA AD 1, AGGIORNA "inverti" */
            if ( bit == 1 )
                inverti = 1 ;
        }
        else
        {
```

```
56              /* RICOPIA IN USCITA L'INVERSO DELLA CIFRA BINARIA INSERITA */
57              if ( bit == 1 )
58                  opposto = 0 ;
59              else
60                  opposto = 1 ;
61          }
62
63          /* STAMPA IL RISULTATO */
64          printf("Risultato␣%d\n", opposto) ;
65
66          /* INCREMENTA IL CONTATORE "num_bits" */
67          num_bits = num_bits + 1 ;
68      }
69      exit(0) ;
70  }
```

14 Esercizio: "Somma di numeri binari"

Si considerino due numeri binari rappresentati in binario puro su N bit. Il valore di N viene inserito da tastiera. 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.

Soluzione

```
1   /* PROGRAMMAZIONE IN C */
2
3   /* File: somma_binario.c */
4   /* Soluzione proposta esercizio "Somma di numeri binari" */
5
6   #include <stdio.h>
7   #include <stdlib.h>
8
9   int main(void)
10  {
11      int N ;             /* numero di cifre binarie */
12      int bit_numero1 ;   /* cifra binaria del primo numero */
13      int bit_numero2 ;   /* cifra binaria del secondo numero */
14      int bit_risultato ; /* cifra binaria risultato dell'operazione di somma */
15      int riporto ;       /* riporto */
16      int num_bits ;      /* contatore per scandire le cifre binarie */
17
18      /* LEGGI IL NUMERO CIFRE BINARIE */
19      printf("Inserisci␣il␣numero␣di␣bit:␣") ;
20      scanf("%d", &N) ;
21
22      /* INIZIALIZZA IL RIPORTO A 0 */
23      riporto = 0;
24
25      /* LEGGI LE CIFRE BINARIE A PARTIRE DAL BIT MENO SIGNIFICATIVO */
26      printf("\nInserisci␣i␣due␣numeri␣binari␣partendo␣dal␣bit␣meno␣significativo\n");
27
28      /* INIZIALIZZA "num_bits" A 0 */
29      num_bits = 0 ;
30
31      while ( num_bits < N )
```

```
32      {
33          /* LEGGI LA CIFRA BINARIA DEL PRIMO NUMERO */
34          printf("\n");
35          printf ("Inserisci la cifra %d di peso 2^%d del primo numero: ",
36                  num_bits+1, num_bits) ;
37          scanf("%d", &bit_numero1) ;
38
39          /* LEGGI LA CIFRA BINARIA DEL SECONDO NUMERO */
40          printf ("Inserisci la cifra %d di peso 2^%d del secondo numero: ",
41                  num_bits+1, num_bits) ;
42          scanf("%d", &bit_numero2) ;
43
44          /* SOMMA LE DUE CIFRE BINARIE */
45          bit_risultato = bit_numero1 + bit_numero2 + riporto ;
46
47          /* VERIFICA CHE IL RISULTATO DELLA SOMMA SIA 0 O 1 */
48          /* ASSEGNA IL RIPORTO A 1 SE IL RISULTATO DELLA SOMMA E' DIVERSO
49          DA 0 O 1, ASSEGNA IL RIPORTO A ZERO ALTRIMENTI */
50          if ( bit_risultato >= 2 )
51          {
52              bit_risultato = bit_risultato - 2 ;
53              riporto = 1 ;
54          }
55          else
56              riporto = 0 ;
57
58          /* STAMPA IL RISULTATO */
59          printf("Il risultato per la cifra %d di peso %d e' %d e il riporto e' %d\n",
60                 num_bits+1, num_bits, bit_risultato, riporto) ;
61
62          /* INCREMENTA IL CONTATORE "num_bits" */
63          num_bits = num_bits + 1 ;
64      }
65
66      /* STAMPA L'INFORMAZIONE SULLA CONDIZIONE DI OVERFLOW */
67      printf("\n") ;
68      if ( riporto == 1 )
69          printf("La somma ha generato overflow\n") ;
70      else
71          printf("La somma non ha generato overflow\n") ;
72
73      exit(0) ;
74  }
```

Soluzione alternativa

```
1   /* PROGRAMMAZIONE IN C */
2
3   /* File: somma_binario2.c */
4   /* Soluzione proposta esercizio "Somma di numeri binari" */
5
6   #include <stdio.h>
7   #include <stdlib.h>
8
9   int main(void)
10  {
11      int N ;             /* numero di cifre binarie */
```

```
12      int bit_numero1 ;   /* cifra binaria del primo numero */
13      int bit_numero2 ;   /* cifra binaria del secondo numero */
14      int bit_risultato ; /* cifra binaria risultato dell'operazione di somma */
15      int riporto ;       /* riporto */
16      int num_bits ;      /* contatore per scandire le cifre binarie */
17
18      /* LEGGI IL NUMERO DI CIFRE BINARIE */
19      printf("Inserisci␣il␣numero␣di␣bit:␣") ;
20      scanf("%d", &N) ;
21
22      /* INIZIALIZZA IL RIPORTO A 0 */
23      riporto = 0;
24
25      /* LEGGI LE CIFRE BINARIE A PARTIRE DAL BIT MENO SIGNIFICATIVO */
26      printf("\nInserisci␣i␣due␣numeri␣binari␣partendo␣dal␣bit␣meno␣significativo\n");
27
28      /* INIZIALIZZA "num_bits" A 0 */
29      num_bits = 0 ;
30
31      while ( num_bits < N )
32      {
33          /* LEGGI LA CIFRA BINARIA DEL PRIMO NUMERO */
34          printf("\n");
35          printf ("Inserisci␣la␣cifra␣%d␣di␣peso␣%d␣del␣primo␣numero:␣",
36                  num_bits+1, num_bits) ;
37          scanf("%d", &bit_numero1) ;
38
39          /* LEGGI LA CIFRA BINARIA DEL SECONDO NUMERO */
40          printf ("Inserisci␣la␣cifra␣%d␣di␣peso␣%d␣del␣secondo␣numero:␣",
41                  num_bits+1, num_bits) ;
42          scanf("%d", &bit_numero2) ;
43
44          /* SOMMA LE DUE CIFRE BINARIE */
45
46          /* CASO 1: IL RIPORTO OTTENUTO DALLA SOMMA DEllE DUE CIFRE BINARIE
47          PRECEDENTI E' 0 */
48          if ( riporto == 0 )
49          {
50              /* VERIFICA SE LE DUE CIFRE BINARIE SONO DIVERSE
51              (1 e 0 oppure 0 e 1) */
52              if ( bit_numero1 != bit_numero2 )
53              {
54                  /* SE LE DUE CIFRE BINARIE SONO DIVERSE LA SOMMA
55                  E' 1 E IL RIPORTO E' 0 */
56                  bit_risultato = 1 ;
57                  riporto = 0 ;
58              }
59              else
60              {
61                  /* SE LE DUE CIFRE BINARIE SONO UGUALI (ENTRAMBE 1 OPPURE 0)
62                  LA SOMMA E' 0 */
63                  bit_risultato = 0 ;
64
65                  /* SE LE DUE CIFRE BINARIE SONO UGUALI A 1 IL RIPORTO E' 1 */
66                  if ( bit_numero1 == 1 ) /* OPPURE bit_numero2 == 1 */
67                      riporto = 1 ;
```

```
68                  else
69                      /* SE LE DUE CIFRE BINARIE SONO UGUALI A 0 IL RIPORTO E' 0 */
70                      riporto = 0 ;
71              }
72          }
73          else
74          {
75              /* CASO 2: IL RIPORTO OTTENUTO DALLA SOMMA DELLE DUE CIFRE
76              BINARIE PRECEDENTI E' 1 */
77
78              /* VERIFICA SE LE DUE CIFRE BINARIE SONO DIVERSE
79              (1 e 0 oppure 0 e 1) */
80              if (bit_numero1 != bit_numero2 )
81              {
82                  /* SE LE DUE CIFRE BINARIE SONO DIVERSE
83                  LA SOMMA E' 0 E IL RIPORTO E' 1 */
84                  bit_risultato = 0 ;
85                  riporto = 1 ;
86              }
87              else
88              {
89                  /* SE LE DUE CIFRE BINARIE SONO UGUALI (ENTRAMBE 1 OPPURE 0)
90                  LA SOMMA E' 1 */
91                  bit_risultato = 1 ;
92
93                  /* SE LE DUE CIFRE BINARIE SONO UGUALI 1 IL RIPORTO E' 1 */
94                  if ( bit_numero1 == 1 ) /* oppure bit_numero2 == 1 */
95                      riporto = 1 ;
96                  else
97                      /* SE LE DUE CIFRE BINARIE SONO UGUALI A 0 IL RIPORTO E' 0 */
98                      riporto = 0 ;
99              }
100         }
101
102         /* STAMPA IL RISULTATO */
103         printf("Il␣risultato␣per␣la␣cifra␣%d␣di␣peso␣%d␣e'␣%d␣e␣il␣riporto␣e'␣%d\n",
104                num_bits+1, num_bits, bit_risultato, riporto) ;
105
106         /* INCREMENTA IL CONTATORE "num_bits" */
107         num_bits = num_bits + 1 ;
108     }
109
110     /* STAMPA L'INFORMAZIONE SULLA CONDIZIONE DI OVERFLOW */
111     printf("\n") ;
112     if ( riporto == 1 )
113         printf("La␣somma␣ha␣generato␣overflow\n") ;
114     else
115         printf("La␣somma␣non␣ha␣generato␣overflow\n") ;
116     exit(0) ;
117 }
```

15 Esercizio: "Conversione Decimale-Binario su un numero fisso di bit"

Scrivere un programma in linguaggio C che converta un numero decimale in un numero binario rappresentato su N bit. L'utente inserisce un numero decimale intero positivo e il numero N di bit su cui il numero

decimale deve essere rappresentata. Il programma visualizzerà i bit che compongono il numero binario partendo dal bit meno significativo. Il programma segnalerà un errore se il numero N di bit inserito dall'utente non è sufficiente per rappresentare il numero decimale.

Suggerimento. Per effettuare la conversione usare il metodo delle divisioni successive. Ad esempio, per convertire il numero decimale 19 su 7 bit, si avrà:

numero da dividere	19	9 (19/2)	4 (9/2)	2 (4/2)	1 (2/2)	0 (1/2)	0 (0/2)
resto	1 (19%2)	1 (9%2)	0 (4%2)	0 (2%2)	1 (1%2)	0 (0%2)	0 (0%2)

cifra binaria	1	1	0	0	1	0	0
peso della cifra binaria	0	1	2	3	4	5	6

Nota: nell'applicazione del metodo delle divisioni successive, l'iterazione termina quando è stato assegnato un valore a ciascuno dei 7 bit.

Soluzione

```c
/* PROGRAMMAZIONE IN C */

/* File: Decimale_Binario_FixedBits.c */
/* Soluzione proposta esercizio "Conversione Decimale-Binario su un numero fisso di bit" */

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

#define DIVIDENDO 2

int main(void)
{
    int numero_decimale ;   /* numero decimale */
    int N ;                 /* numero di cifre binarie */
    int bit ;               /* cifra binaria */
    int num_bits ;          /* contatore per scandire le cifre binarie */

    /* LEGGI IL NUMERO DECIMALE */
    printf("Inserire␣il␣numero␣decimale␣da␣convertire:␣") ;
    scanf("%d", &numero_decimale) ;

    /* LEGGI IL NUMERO DI BIT */
    printf("Inserisci␣il␣numero␣di␣bit:␣") ;
    scanf("%d", &N) ;

    /* VERIFICA CHE IL NUMERO DI BIT SIA SUFFICIENTE PER RAPPRESENTARE
    IL NUMERO DECIMALE */
    if ( pow(DIVIDENDO,N) - 1 < numero_decimale )
        printf("Errore:␣il␣numero␣di␣bit␣e'␣insufficiente\n");
    else
    {
        /* INIZIALIZZA "num_bits" A 0 */
        num_bits = 0 ;

        /* IL CICLO CALCOLA LE CIFRE BINARIE PER RAPPRESENTARE IL NUMERO
        DECIMALE, PARTENDO DALLA CIFRA BINARIA MENO SIGNIFICATIVA (LSB) */
        while ( num_bits < N )
        {
```

```
40              /* CALCOLA LA CIFRA BINARIA DI PESO "num_bits" */
41              bit = numero_decimale % DIVIDENDO ;
42
43              /* CALCOLA IL NUMERO DECIMALE DA DIVIDERE PER "dividendo"
44              ALLA PROSSIMA ESECUZIONE DEL CICLO */
45              numero_decimale = numero_decimale/DIVIDENDO ;
46
47              /* STAMPA IL RISULTATO */
48              printf("Cifra␣binaria␣di␣peso␣2^%d:␣%d\n", num_bits, bit) ;
49
50              /* INCREMENTA IL CONTATORE "num_bits" */
51              num_bits = num_bits + 1 ;
52          }
53      }
54      exit(0) ;
55  }
```

16 Esercizio: “Numeri di Fibonacci”

Scrivere un programma in linguaggio C che calcoli e stampi i primi N numeri della serie di Fibonacci, con N inserito da tastiera. La serie di Fibonacci inizia con 1, 1 ed ogni numero successivo è dato dalla somma dei due precedenti: 1, 1, 2, 3, 5, 8, 13, 21 . . .

Soluzione

```
1   /* PROGRAMMAZIONE IN C */
2
3   /* File: fibonacci.c */
4   /* Soluzione proposta esercizio "Numeri di Fibonacci" */
5
6   #include <stdio.h>
7   #include <stdlib.h>
8
9   int main(void)
10  {
11      int N ;              /* numero di termini della serie */
12      int nuovo_termine;   /* nuovo termine della serie */
13      int prec_1, prec_2 ; /* due termini precedenti nella serie */
14      int num_termini;     /* contatore per scandire i termini della serie */
15
16      /* LEGGI IL NUMERO TERMINI DELLA SEQUENZA */
17      printf("Inserisci␣il␣numero␣di␣termini␣della␣serie␣di␣Fibonacci:␣") ;
18      scanf("%d", &N) ;
19
20      /* INIZIALIZZA A 1 I PRIMI DUE TERMINI DELLA SERIE */
21      prec_1 = 1 ;
22      prec_2 = 1 ;
23
24      /* INIZIALIZZA A 1 IL PRIMO VALORE DELLA SERIE */
25      nuovo_termine = 1 ;
26
27      /* INIZIALIZZA A 0 IL CONTATORE CHE SCANDISCE I TERMINI DELLA SERIE */
28      num_termini = 0 ;
29
30      while ( num_termini < N )
31      {
```

```
32          /* I PRIMI DUE TERMINI DELLA SERIE SONO UGUALI A 1.
33          I TERMINI SUCCESSIVI SONO CALCOLATI COME SOMMA DEI DUE TERMINI PRECEDENTI */
34          if ( num_termini >= 2 )
35          {
36              /* CALCOLA IL NUOVO TERMINE DELLA SERIE */
37              nuovo_termine = prec_1 + prec_2 ;
38
39              /* AGGIORNA IL VALORE DEI DUE TERMINI PRECEDENTI NELLA SERIE */
40              prec_2 = prec_1 ;
41              prec_1 = nuovo_termine ;
42          }
43
44          /* STAMPA UN NUOVO TERMINE DELLA SERIE */
45          printf("%d ", nuovo_termine) ;
46
47          /* INCREMENTA IL CONTATORE "num_termini" */
48          num_termini = num_termini + 1 ;
49      }
50
51      /* RIPORTA A CAPO IL CURSORE AL TERMINE DELLA STAMPA DELLA SERIE */
52      printf("\n");
53      exit(0) ;
54  }
```