Corso di Laurea in Matematica

LABORATORIO DI INFORMATICA 2002-2003

Compito Totale - 21 Marzo 2003

Soluzione degli Esercizi

Esercizio 1. (versione 1)
i) Dato il numero binario 1010101111, dire qual è la sua rappresentazione nei sistemi decimale, ottale ed esadecimale.
ii) Convertire il numero decimale 227 nella sua rappresentazione binaria.
iii) Dato il numero 227_8 nel sistema ottale, dire qual è la sua rappresentazione nei sistemi decimale, binario ed esadecimale.
iv) Convertire il numero decimale -39 nella sua rappresentazione binaria in complemento a due.
v) Convertire il numero frazionario 0.52 nella sua rappresentazione binaria utilizzando 6 cifre.
Soluzione
i) Decimale: 687, ottale: 1257, esadecimale: 2AF.
ii) 11100011.
iii) Decimale: 151, binario: 10010111, esadecimale: 97.
iv) 1011001.
v) 0.100001.

Esercizio 1. (versione 2)
i) Dato il numero binario 1010111101, dire qual è la sua rappresentazione nei sistemi decimale, ottale ed esadecimale.
ii) Convertire il numero decimale 231 nella sua rappresentazione binaria.
iii) Dato il numero 231_8 nel sistema ottale, dire qual è la sua rappresentazione nei sistemi decimale, binario ed esadecimale.
iv) Convertire il numero decimale -43 nella sua rappresentazione binaria in complemento a due.
v) Convertire il numero frazionario 0.49 nella sua rappresentazione binaria utilizzando 6 cifre.
Soluzione
i) Decimale: 701, ottale: 1275, esadecimale: 2BD.
ii) 11100111.
iii) Decimale: 153, binario: 10011001, esadecimale: 99.
iv) 1010101.
v) 0.011111.

Esercizio 2. Si consideri il seguente frammento di programma in linguaggio macchina, memorizzato a partire dalla locazione di memoria di indirizzo 100:

```
100 READ 11
101 READ 12
102 LOADA 12
103 LOADB 10
104 DIF
105 JUMPZ 114
106 LOADB 11
107 DIF
108 JUMPZ 112
109 LOADA 12
110 STOREA 11
111 JUMP 101
112 WRITE 12
113 JUMP 115
114 WRITE 10
115 HALT
```

Si supponga che la cella di memoria di indirizzo 10 contenga il valore 0. Data una sequenza di numeri interi $n_1, n_2, \ldots, n_k$ in ingresso, dire che cosa fa il programma e che cosa viene stampato in uscita. Dire che cosa viene stampato nel caso in cui in ingresso si abbiano le sequenze -2 4 4 7 0 e 3 -1 3 0.

Soluzione
Il programma confronta ogni numero della sequenza (a partire dal secondo) con il numero che lo precede e, non appena trova due numeri adiacenti uguali, stampa tale numero, altrimenti stampa 0. Data in ingresso la sequenza -2 4 4 7 0, il programma stampa 4 e poi termina. Data in ingresso la sequenza 3 -1 3 0, il programma stampa 0 e poi termina.

Esercizio 3. (versione 1)
Scrivere una procedura in C che, data una matrice quadrata $n \times n$ di caratteri, stampa i caratteri presenti nel triangolo in alto a destra inclusa la diagonale. Ad esempio, data la matrice

```
o p q
r s t
u v z
```

la procedura stampa i caratteri `opqstz`.

Soluzione

```
#define n 3
typedef char Matrice[n][n];

void triangolo_ad (Matrice m) {
   int i,j;
   for (i=0; i<n; i++)
     for (j=i; j<n; j++)
       printf("%c", m[i][j]);
}
```

Nella versione 2 dell'Esercizio 3 si richiede una procedura che stampa i caratteri presenti nel triangolo in basso a sinistra inclusa la diagonale. Ad esempio, data la stessa matrice della versione 1, la procedura stampa i caratteri `orsuvz`.

Soluzione

```
void triangolo_bs (Matrice m) {
   int i,j;
   for (i=0; i<n; i++)
     for (j=0; j<=i; j++)
       printf("%c", m[i][j]);
}
```

Esercizio 4. (versione 1)
Definire una struttura C che rappresenta una *data* ed una struttura C che riflette i dati di un'*auto*: marca (es. Fiat, Alfa Romeo, ...), tipo (es. punto, bravo, stilo, ...), una provincia (es. RI, MI, ...), una targa (unica per ogni auto e supposta intera) e la data di immatricolazione. Supponendo che:
- esiste un vettore dove sono contenute tutte le auto,
- l'elemento i-esimo del vettore non contiene i dati di un'auto se la targa è pari a zero,
scrivere le seguenti procedure o funzioni che:
- confrontano due date verificando se una è piú recente dell'altra;
- stampano i dati di tutte le auto immatricolate dopo una data d o un messaggio di avviso

se non vi sono auto nel vettore.

Soluzione

```
typedef char string[10];

typedef struct{
        int giorno;
        int mese;
        int anno;
} data;

typedef struct {
        char provincia[2];
        int targa;
        string marca;
        string tipo;
        data imm;
} automobile;

int compara_date(data d,data d1) {
  if (d.anno > d1.anno) return 1;                    /*d successiva a d1*/
  if (d.anno == d1.anno)
    if (d.mese > d1.mese) return 1;                  /*d successiva a d1*/
  if (d.anno == d1.anno)
    if (d.mese == d1.mese)
      if (d.giorno > d1.giorno) return 1;            /*d successiva a d1*/
  if (d.anno == d1.anno)
    if (d.mese == d1.mese)
      if (d.giorno == d1.giorno) return 0;           /* date uguali */
  return -1;                        /*in tutti gli altri casi d precede d1*/
}

void recenti(automobile v[], data d, int dim) {
  int i;
  for (i=0; (i<dim)&&(v[i].targa == 0); i++);
  if (i == dim){
    printf("Non sono presenti auto nel database\n");
    return;
   }
   for (i=0; i<dim; i++){
     if (compara_date(v[i].imm,d) > 0){
       printf("%s %s\n", v[i].marca, v[i].tipo);
       printf("%s %d\n", v[i].provincia, v[i].targa);
       printf("%d/%d/%d\n", v[i].imm.giorno, v[i].imm.mese, v[i].imm.anno);
     }
   }
}
```

Nella versione 2 dell'Esercizio 4 si richiede di stampare i dati di tutte le auto immatricolate *prima* di una data d, per cui basta sostituire il confronto
`(compara_date(v[i].imm,d) > 0)` con `(compara_date(v[i].imm,d) < 0)`.

Esercizio 5. (versione 1)
Scrivere una funzione ricorsiva in C che, dato un array di interi a, restituisce *true* se il valore assoluto della differenza tra due elementi adiacenti di a è non crescente, e restituisce *false* altrimenti. Ad esempio, dato l'array $\{7,\ 2,\ -2,\ -6,\ -4\}$, la funzione restituisce *true* poiché $5 > 4 \geq 4 > 2$.

Soluzione

```
#define n 5
typedef int Array[n];
typedef enum {false,true} boolean;

boolean non_cresc (Array a) {
   int abs (int x);
   boolean diff_non_cresc (Array a, int d, int i);
   return diff_non_cresc(a,abs(a[0]-a[1]),1);
}

boolean diff_non_cresc (Array a, int d, int i) {
   int abs (int x);
   int d1;
   if (i == n-1) return true;
   d1 = abs(a[i]-a[i+1]);
   if (d1 > d) return false;
   return diff_non_cresc(a,d1,i+1);
}

int abs (int x) {
  if (x < 0) return -x;
  return x;
}
```

Nella versione 2 dell'Esercizio 5 si richiede una funzione ricorsiva che, dato un array di interi a, restituisce *true* se il valore assoluto della differenza tra due elementi adiacenti di a è non decrescente, e restituisce *false* altrimenti. Nella soluzione data sopra basta sostituire il confronto `(d1 > d)` con `(d1 < d)`.

Esercizio 6. Scrivere una procedura in C (*non* una funzione) che, dati in ingresso una stringa s ed un carattere c, restituisca in un terzo parametro intero il numero di occorrenze di c all'interno di s. Ad esempio, il carattere `c` occorre 5 volte in `"abccbccc"`. Scrivere inoltre un `main` in cui si utilizza la procedura precedentemente dichiarata.

Soluzione

```
void conta_occorrenze(char s[], char c, int *count) {
    int i, val;
    val = 0;
    for (i=0; s[i]!='\0'; i++)
      if (s[i] == c) val = val+1;
    *count = val;
    return;
}
```

```
main() {
    int val;
    conta_occorrenze("abccbccc",'c',&val);
    printf("%d",val);
    return 0;
}
```