Esame di Fondamenti di Informatica
Ingegneria Meccanica (A–O)
Appello del 19/4/2002

Esercizio 1

Un elaboratore adotta per i **numeri interi** una rappresentazione in complemento a due su un byte e per i **numeri reali** una rappresentazione in virgola mobile con un byte per la mantissa normalizzata in segno/modulo (si usi il primo bit della mantissa per rappresentare il segno) ed un byte per l'esponente in complemento a due.
Si consideri l'espressione:

61/13 – 11.13

Indicare il risultato dell'espressione ottenuto eseguendo il calcolo con l'elaboratore dato. Mostrare i passaggi intermedi eseguiti dall'elaboratore con particolare riferimento alle operazioni ed alla rappresentazione interna in binario, nonché i relativi eventuali errori compiuti. Scrivere il risultato finale in codifica decimale e confrontarlo con il risultato atteso (suggerimento: si approssimi 61/13 = 4.6923).

Esercizio 2

Sia dato il seguente programma C:

```
#include <stdio.h>

const int n=4;

int F(int *V, int i) {
  int k=*V;
  if(i<n-1) return k+F(++V, ++i);
  else return k;
}

main() {
  int V[n];
  int i;

  for(i=0; i<n; i++) V[i]=n-i;
  printf("%d\n", F(V, 0));
  for(i=0; i<n; i++) printf("%d\t", V[i]);
  printf("\n");
}
```

Scrivere l'output prodotto dal programma ed illustrare la dinamica dei record di attivazione sullo stack. Motivare opportunamente le risposte.

Esercizio 3

L'agenzia di viaggi "AliVorno" permette la prenotazione di viaggi in partenza dal capoluogo di provincia toscano. A tal scopo, le informazioni su tutti i voli sono memorizzate all'interno di un file binario chiamato "VOLI.DAT". Ogni record del file contiene il nome della compagnia aerea che gestisce il volo, il codice del volo, il nome della città di arrivo e l'orario di partenza del volo. Si supponga che, all'interno del file, i record siano ordinati per orario di partenza.
Si scriva un programma C per l'agenzia di viaggi "AliVorno" che:

1. Legga il contenuto del file "VOLI.DAT" e lo memorizzi all'interno di un vettore `V`. Si supponga che il massimo numero di voli sia 200.
2. Legga da tastiera il nome di una città e provveda ad inserire in un secondo vettore `V1` le informazioni sui soli voli destinati in tale città (si supponga che esista sempre almeno un tale volo). A tal scopo, si supponga l'esistenza di una procedura `inserisci` che, dato un vettore di voli, un intero rappresentante una posizione all'interno del vettore e le informazioni su un volo, provveda a copiare le informazioni sul volo nella posizione specificata nel vettore. Per il confronto tra stringhe si utilizzi la funzione `strcmp`, dichiarata all'interno del file "string.h", che, date due stringhe, restituisce 0 se e solo se le due stringhe sono identiche.
3. Letto da tastiera un orario, provveda a stampare le informazioni sul primo volo disponibile dopo tale orario all'interno del vettore `V1` (si presti attenzione al caso in cui l'orario dell'ultimo volo sia precedente all'orario dato).
4. Si scriva il codice della procedura `inserisci` (a tal scopo si utilizzi la funzione `strcpy`, dichiarata all'interno del file "string.h", che, date due stringhe, copia la seconda nella prima).

Esercizio 4

Definizione e proprietà di un algoritmo.

Soluzione Esercizio 1

Rappresentazione **61**:
$(61)_{10} = (111101)_2$
rappr. interna = **0011 1101**

Rappresentazione **13**:
$(13)_{10} = (1101)_2$
rappr. interna = **0000 1101**

Risultato **61 / 13**:

```
111101 –  | 1101
1101      | 100.10110
 010
 010010 –
  1101
  010100 –
   1101
   001110 –
    1101
    00010
```

Errore di troncamento!
Risultato = 100.10110 = $(4.6875)_{10}$

Rappresentazione **11.13**:
$(11.13)_{10} = (1011.0010)_2$
Errore di troncamento!
Normalizzando: 0.10110010×2^4
byte esponente = **0000 0100**
byte mantissa = **0011 0010**

Risultato **4.6875 – 11.13**:
Normalizzazione del risultato: -0.01001011×2^4
Nessun errore di incolonnamento
$0.1011\ 0010 \times 2^4 -$
$\underline{0.0100\ 1011} \times 2^4 =$
$0.0110\ 0111 \times 2^4$
Normalizzando: -0.11001110×2^3
Errore di cancellazione!
byte esponente = **0000 0011**
byte mantissa = **1100 1110**
risultato: $(-110.0111)_2 = (-6.4375)_{10}$
risultato atteso: $(-6.4377)_{10}$

Soluzione Esercizio 2

Il risultato del programma è il seguente:

10
4 3 2 1

Infatti, ogni invocazione della funzione `F` produce, come risultato, la somma dell'`i`-esimo elemento del vettore `V` e del valore restituito dalla funzione `F` chiamata con un valore di `i` incrementato di 1. La ricorsione ha termine quando `i` assume il valore 3. Di conseguenza, l'effetto della funzione è quello di sommare i valori contenuti nel vettore. Il valore restituito dalla funzione è pertanto 10, mentre i valori del vettore sono immutati.

Soluzione Esercizio 3

```
#include <stdio.h>
#include <string.h>
#define DIM 200

 typedef struct {
  char compagnia[30], codice[10], citta[20];
  int ore, min;
} volo;

void inserisci(volo V[], int i, volo e);

main() {
  volo V[DIM], V1[DIM], e;
  FILE *fp=fopen("VOLI.DAT", "rb");
  char c[20];
  int n, n1=0, h, m;

  n=fread(V, sizeof(volo), DIM, fp);
  fclose(fp); /* fine domanda 1. */
  scanf("%s", c);
  for(i=0; i<n; i++)
    if(strcmp(V[i].citta, c))
      inserisci(V1, n1++, V[i]);
  /* fine domanda 2. */
  scanf("%d %d", &h, &m);
  for(i=0; i<n; i++)
    if(V1[i].ore*60+V1[i].min>=h*60+m) {
      printf("%s %s %d %d\n", V1[i].compagnia, V1[i].codice,
        V1[i].ore, V1[i].min);
      break;
    }
  if(i==n)
    printf("%s %s %d %d\n", V1[0].compagnia, V1[0].codice,
      V1[0].ore, V1[0].min);
  /* fine domanda 3. */
}

void inserisci(volo V[], int i, volo e) {
  strcpy(V[i].compagnia, e.compagnia);
  strcpy(V[i].codice, e.codice);
  strcpy(V[i].citta, e.citta);
  V[i].ore=e.ore;
  V[i].min=e.min;
} /* fine domanda 4. */
```