Fondamenti di Informatica - Ing. Civile/Edile - Dott. Penzo
Soluzione compito A - 11/06/2001

Esercizio 1
Per i passaggi intermedi vedere le dispense.

19 in base 2: 10011
0.7 in base 2: .101100110....

19.7 in base 2: 10011.101100110....
normalizzando: .10011101 mantissa e 00000101 esponente (err. di arrotondamento)
rappresentaz. : .00011101 mantissa e 00000101 esponente

23 in base 2: 10111
9 in base 2: 00001001
-9 in complemento a 2: 11110111

23 - 9 = 00001110 = 14 in base 10
14 in floating point normalizzato: .11100000 mantissa e 00000100 esponente
14 in rappresentazione interna: .01100000 mantissa e 00000100 esponente

14 incolonnato all'esponente maggiore:
.01110000 mantissa e 00000101 esponente

somma 19.7 + 14: .10011101
.01110000

1.00001101 mantissa e 00000101 esponente

normalizzando: .1000110 (errore di troncamento) mantissa e 00000110 esponente
ovvero: 100001.10 che in decimale vale 32 + 1 + 0.5 = 33.5 (invece di 33.7)

Esercizio 2
La prima printf stampa il valore restituito dalla funzione F. Il valore stampato è 6 perché il ciclo for nella funzione F termina quando l'indice i supera o eguaglia il valore 5. Poiché i assume solo valori pari (i += 2) il ciclo terminerà quando i assumerà valore 6 e tale valore verrà restituito al programma chiamante tramite l'istruzione return. Poiché il vettore V viene passato per indirizzo a F, la funzione lo modifica e il ciclo for del main stampa i valori modificati per gli indici che vanno da 4 a 1 compresi, ossia: 9, 4, 5, 2 (verticalmente).

Esercizio 3

```
#include<stdio.h>
#include<string.h>
#define MAX_PILOTI 50

typedef struct {
                char nome[20];
                char cognome[20];
                char nazionalità[20];
                char scuderia[20];
                } pilota;

typedef struct {
                char nome[20];
                char cognome[20];
                char scuderia[20];
                int punteggio;
                } classifica;

typedef struct {
                int numgp;
                char nome[20];
                char cognome[20];
                int posiz;
                } podio;


void AggiornaPunteggio(classifica V[], char n[], char c[], int pos);

int numelem = 0;

main()
{
pilota elem;
podio p;
int i;
classifica V[MAX_PILOTI];
FILE *fp;

fp = fopen("PILOTI.DAT","rb");
if (fp == NULL) printf("Errore di apertura file PILOTI.DAT");
else{
        while(!feof(fp)){
                fread(&elem, sizeof(pilota),1,fp);
                strcpy(V[numelem].nome, elem.nome);
                strcpy(V[numelem].cognome, elem.cognome);
                strcpy(V[numelem].scuderia, elem.scuderia);
                V[numelem++].punteggio = 0;
        }
```

```
		fclose(fp);
		fp = fopen("PODIO.DAT","ab");
		printf("Inserisci il numero del Gran Premio: ");
		scanf("%d",&p.numgp);
		printf("Inserisci i dati dei classificati sul podio: ");
		for (i=0;i < 3; i++){
			printf("Inserisci il nome del pilota: ");
			gets(p.nome);
			printf("Inserisci il cognome del pilota: ");
			gets(p.cognome);
			do{
				printf("Inserisci il posto sul podio: ");
				scanf("%d",&p.posiz);
			}while (p.posiz < 1 || p.posiz > 3);
			fwrite(&p, sizeof(podio),1,fp);
		}
		fclose(fp);
		fp = fopen("PODIO.DAT","rb");
		while(!feof(fp)){
			fread(&p, sizeof(podio),1,fp);
			AggiornaPunteggio(V, p.nome, p.cognome, p.posiz);
		}
		fclose(fp);
} /* else */
} /*main */

void AggiornaPunteggio(classifica V[], char n[], char c[], int pos)
{
int i = 0, trovato = 0;
while((i < numelem) && (!trovato))
{
		if (!strcmp(V[i].nome, n) && !strcmp(V[i].cognome, c))
		{
			trovato = 1;
			switch(pos){
				case 1:
					V[i].punteggio += 10;
					break;
				case 2:
					V[i].punteggio += 6;
					break;
				case 3:
					V[i].punteggio += 4;
					break;
				default:
					break;
			}
		}
		i++;
}
```

```
if (!trovato)
        printf("Pilota non presente nell'elenco");
return;
}
```