Esercizi di preparazione al II Compitino
Fondamenti di Informatica A - Prof. Paola Mello (non sono inclusi esercizi su liste, seguirà altra dispensa esercizi su liste)

Il compito prevede:

- un esercizio di analisi;
- due esercizi di sintesi (o uno composto da più domande);
- una domanda.

Esercizio (analisi)
Dato il seguente programma C:

```
#include <stdio.h>
#define Dim 5
char  c = 'a';
char A[Dim]={'c','i','a','o','\0'};
char vet[Dim]={'e','e','e','e','\0'};

void sub(char vet1[], char vet2[]);
void stampa(char vet[]);

main()
{stampa(A);
 sub(A,vet);
 stampa(vet);
 stampa(A);
 printf("%c",c);}

void sub(char vet1[Dim], char vet2[Dim])
{int i; char c='b';
 for(i=0;i<Dim-1;i++)
    if (vet2[i]>vet1[i])
       vet1[i]=vet2[i];
       else vet1[i]=c;
 }

void stampa(char vet[])
{printf("Vettore:\n");
 printf("%s",vet);
 printf("\n");
}
```

Che cosa viene stampato dal programma (si motivi opportunamente la risposta)? Si dica inoltre se la variabile `i` definita nella procedura `sub()` è visibile dalla procedura `stampa()` e dal main.

Soluzione (analisi)
La prima stampa produce i valori ciao che corrispondono al vettore `A` inalterato. Dopo di che viene chiamata la procedura `sub` che modifica i valori di `vet` e di `A`. Se l'elemento di `vet` è maggiore alfabeticamente del corrispondente elemento di `A`, quest'ultimo elemento viene sovrascritto dall'elemento di `vet`, altrimenti viene sovrascritto con il carattere 'b'.

Quindi `vet` viene modificato in ebeb. Viene poi stampato il vettore `A`, producendo eeee. Poi viene stampato `vet` ebeb.
Infine viene stampato `c`, che è la variabile definita esternamente al main e non quella definita nella procedura `sub()`.

Quindi, il risultato stampato è:

```
Vettore:
ciao
Vettore:
eeee
Vettore:
ebeb
a
```

la variabile *i* definita nella procedura `sub()` non è visibile nella procedura `stampa()` e nel main.

Esercizio (sintesi)
Dato un file di testo **mesi.txt**, si supponga che contenga righe ciascuna contenente una stringa (nome del mese) ed un intero (numero di giorni). Ad esempio:

```
gennaio   31
febbraio  28
marzo     31
aprile    30
```

Si stampino a video i nomi dei mesi che hanno 31 giorni.

Esercizio (soluzione)

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

main() {
int giorni; FILE* f;
char nome[20];
if ((f=fopen("mesi.txt", "r"))==NULL) {
    printf("Il file non esiste!"); exit(1); }
 while(fscanf(f,"%s%d\n",  nome, &giorni) != EOF)
      if (giorni == 31)
      printf("%s\n", nome);
fclose(f);
}
```

Esercizio (sintesi)
Si scriva una procedura che legga da input una serie di numeri positivi (massimo 10) e termini con un numero negativo o nullo. Tale procedura restituisce in uscita (come parametri passato per riferimento) un vettore **V** (di dimensione fisica 10 dichiarato nel programma chiamante) contenente i numeri letti e il numero effettivo di elementi letti **N** (ossia la dimensione logica del vettore).

```
void leggi(int V[], int *N);
```

Si dia anche un esempio di chiamata.

Soluzione (sintesi)

```
#include <stdio.h>

void leggi(int V[], int *N)
{
    int i = 0;
    int Num;
    printf("inserisci un intero positivo (0 o neg per terminare): ");
    scanf("%d", &Num);
    while (Num > 0)
    { V[i] = Num;
      i++;
      printf("inserisci un intero positivo (0 o neg per terminare): ");
      scanf("%d", &Num);
    }
 *N = i;
}

chiamata:

main(){
int VETTORE[10];
int DimLogica;

leggi(VETTORE, &DimLogica);
printf("Numero effettivo di elementi nel vettore: %d",DimLogica);
}
```

Esercizio (domanda)

Supponiamo di dover assegnare due strutture identiche contenenti un intero e un float:

```
struct prova{ int A;
              float B;
              };
struct prova S1;
struct prova S2;
```

Quali delle seguenti istruzioni non è consentita per effettuare l'assegnamento ?

```
1.   S1 = S2;
2.   S1 == S2;
3.   S1.A = S2.A; S1.B = S2.B;
```

Soluzione

2. Non è un operatore di assegnamento ma di confronto.

Esercizio (domanda)

Si descrivano sinteticamente i componenti della macchina di Von Neumann

Soluzione

(si consulti la dispensa relativa)

Esercizio (analisi)

Dato il seguente programma C:

```
#include <stdio.h>
#define Dim 5

int f(int *V, int k)
{int i, s=Dim; for(i=Dim-1;i>=0; i-=k) {    V[i]=i;        s+=i;
     }
 return s;
}

main()
{ int A[Dim]={1,1,1,1,1};
     int i;
        for (i=0; i<Dim; i+=2)
          A[i]-=i;
     printf("%d\n", f(A,3));
     for(i=0; i<Dim; i++)
          printf("%d\t", A[i]);
}
```

Qual è l'uscita del programma? La risposta deve essere opportunamente motivata.

Soluzione:

Il ciclo for nel main modifica le componenti del vettore A di indice pari. Il vettore diventa:
{1,1,-1,1,-3}
Il main chiama poi la funzione f, passando il vettore A per indirizzo e il valore 3.
La funzione, a passi di 3 (k=3), assegna alle componenti del vettore V (che rappresenta l'indirizzo del primo elemento di A) di indice i=4 e i=1 il valore dell'indice stesso.
Il vettore A diventa:
{1,1,-1,1,4}
e somma alla variabile s (inizialmente uguale a 5) tali indici, restituendo il valore di s in uscita (s=5+4+1=10).
Nel main viene stampato il valore restituito dalla funzione:
10
e il vettore A:
1 1 -1 1 4

Esercizio (domanda)

Qual è la relazione tra vettori e puntatori nel linguaggio C? È lecito scrivere:

```
int V[10], *punt;
punt=V;
```

Motivare la risposta.

Soluzione:
Un puntatore è una variabile che ha come valore l'indirizzo di un altro dato. Un vettore rappresenta un indirizzo costante (quello del primo elemento del vettore). La differenza è che tale indirizzo è costante e non se ne può variare il valore.
L'istruzione punt=V è lecita e assegna al puntatore l'indirizzo del primo elemento del vettore (rappresentato dal nome V).

Esercizio (analisi)

Dato il seguente programma C:

```
#include <stdio.h>
#define D 4
float V[D]={1.5, 2.5, 3.5, 4.5};
float A[D]={0,0,1,1};

float Fun(float V[], int k)
{int i;
 for(i=0;i<D; i+=k)
          V[i]=i;
 return i;
}

main()
{int i;
 printf("%f\n", Fun(A,2));
 for(i=0; i<D; i++)
      printf("%f\t", V[i]);
 printf("\n");
 for(i=0; i<D; i++)
      printf("%f\t", A[i]);
}
```

Che cosa stampa il programma (si motivi opportunamente la risposta)?

Soluzione
Il programma main chiama la funzione Fun, passando il vettore A per indirizzo e il valore 2.
La funzione, a passi di 2 (k=2), assegna alle componenti del parametro formale V (al quale è stato assegnato l'indirizzo del vettore A) di indice i=0,2 il valore dell'indice stesso, e al termine restitisce il valore dell'indice i (=4).
Tale risultato viene stampato dalla printf nel programma main:
4.000000
Terminata la funzione, il main stampa il contenuto del vettore V:
1.500000 2.500000 3.500000 4.500000
e il vettore A:
0.000000 0.000000 2.000000 1.000000
Si noti che il vettore V dichiarato nella parte delle dichiarazioni globali non è stato modificato dalla chiamata della funzione Fun. Infatti a tale funzione viene trasferito per indirizzo il vettore A (modificato). La funzione Fun accede alle componenti di A attraverso il parametro formale V che non è il vettore dichiarato in precedenza, ma un nome (identico al precedente) con il quale si riferisce – all'interno della funzione Fun – il parametro attuale A passato per indirizzo.

Esercizio (sintesi)

Si scriva un programma C che tramite tre funzioni, *leggi*, *media*, *stampa*:

a) Legga da terminale una prima sequenza di numeri terminati dal valore 0 (un numero su ogni linea) e li inserisca in un vettore A;

b) Legga da terminale una seconda sequenza di numeri terminati dal valore 0 e li inserisca in un altro vettore B;

c) Sia in grado di calcolare la media degli elementi di un vettore con la funzione *media*;

d) Stampi a video il vettore (A oppure B) la cui media è maggiore.

NOTA: Si ipotizzi una dimensione massima di 10 per i vettori

Esempio:

Vettore A:	3	5	7	8	2		Media=25/5=5
Vettore B:	2	6	10	2	3	15	Media=38/6=6.333
Vettore Max:	2	6	10	2	3	15	

Soluzione:

```
#include<stdio.h>
#define MAX 10

int leggi(int vet[],char nome);
float media(int vet[],int lung);
void stampa(int vet[],int lung);

main()
{int A[MAX],B[MAX],a,b;
a=leggi(A,'A');
b=leggi(B,'B');
if(media(A,a)<media(B,b))stampa(B,b);
   else if(media(A,a)==media(B,b))printf("I vettori hanno la stessa
media!\n");
      else stampa(A,a);
}

int leggi(int vet[],char nome)
{int i=0;
printf("\nScrivi gli elementi del vettore %c\n",nome);
do
   {printf("Elemento %d: ",i+1);
   scanf("%d",&vet[i]);
   i++;
   }
while(vet[i-1]!=0&&i<MAX);
return i-1;
}
```

```
float media(int vet[],int lung)
{int i;
float sum=0;
for(i=0;i<lung;i++)sum+=vet[i];
return sum/lung;
}

void stampa(int vet[],int lung)
{int i;
printf("Il vettore con media maggiore è:\n");
for(i=0;i<lung;i++)printf("%d ",vet[i]);
}
```

Esercizio (analisi)
Dato il seguente programma C:

```
#include<stdio.h>
#define Dim 4
char  single = 'n';
char Primo[Dim]={'a','b','c','\0'};
char Secondo[Dim]={'b','b','b','\0'};

int change(char string1[], char string2[]);
void print(char string[]);

main()
{int N;
 print(Primo);
 N = change(Primo,Secondo);
 print(Secondo);
 print(Primo);
 printf("%c,%d",single,N);}

int change(char string1[], char string2[])
{int j; int i=4; char single='f';
 for(j=0;j<Dim-1;j++)
    if (string2[j]>string1[j])
       {string1[j]=string2[j];
        i++;}
        else Primo[j]=single;
 return i;}

void print(char string[])
{printf("Vettore:\n");
 printf("%s",string);
 printf("\n");
}
```

Che cosa viene stampato dal programma? La risposta deve essere opportunamente motivata. Si dica inoltre se la variabile N definita nel main è visibile anche dalle funzioni/procedure **`change`** e **`print`**.

Soluzione
La prima stampa produce i valori `abc` che corrispondono al vettore **`Primo`** inalterato. Dopo di che viene chiamata la procedura **`change`** che modifica i valori di **`Primo`** e di **`Secondo`**. Se l'elemento di **`Secondo`** è maggiore alfabeticamente del corrispondente elemento di **`Primo`**, quest'ultimo elemento viene sovrascritto dall'elemento di **`Secondo`**, altrimenti il j-esimo elemento di **`Primo`** viene sovrascritto con il carattere 'f' (la definizione di **`single`**, interna alla procedura **`change`**, nasconde la definizione data come variabile globale). Il valore restituito da **`change`** è il valore iniziale di ***i*** più il numero di volte in cui si è verificato il caso **`Secondo[j]`** > **`Primo[j]`**, cioè 4+1.

Quindi **`Primo`** diventa bff. Viene poi stampato il vettore **`Secondo`**, producendo bbb. Poi viene stampato **`Primo`**: bff.
Infine viene stampato **`single`**, che è la variabile definita esternamente al main e non quella definita nella procedura **`change`** e il valore restituito da **`change`**.

Quindi, il risultato stampato è:

```
Vettore:
abc
Vettore:
bbb
Vettore:
bff
n,5
```

la variabile ***N*** definita nel main non è visibile nelle procedure/funzioni.

Esercizio (sintesi)
Dato un file di testo **estratti.txt**, si supponga che sia costituito da righe ciascuna contenente una ruota (nome della ruota del lotto) ed un intero (numero estratto). Ad esempio:

```
napoli    31
genova    28
napoli    60
```

Si scriva un programma C che prenda in ingresso il nome di una ruota e stampi tutti i numeri estratti su quella ruota.

Soluzione

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

main() {
int numero; FILE* f;
char ruota[20],miaruota[20];

printf("Inserire ruota: ");
scanf("%s",miaruota);

if ((f=fopen("estratti.txt", "r"))==NULL) {
    printf("Il file non esiste!"); exit(1); }
 while(fscanf(f,"%s %d\n", ruota, &numero) != EOF)
      if (strcmp(ruota,miaruota)==0)
      printf("%d\t", numero);
fclose(f);
}
```

Esercizio (domanda)

Qual è la differenza tra un parametro formale passato per indirizzo e uno passato per valore ad una procedura:

A. Se modificato all'interno della procedura, il parametro passato per indirizzo non comporta modifica sul parametro attuale. Viceversa avviene per il parametro passato per valore;
B. Se modificato all'interno della procedura, il parametro passato per valore non comporta modifica sul parametro attuale. Viceversa avviene per il parametro passato per indirizzo;
C. I parametri passati per indirizzo sono solo vettori, mentre tutti gli altri non possono essere passati che per valore.

Soluzione:
La risposta corretta è la B.

Esercizio (domanda)
Di che tipo è la variabile **f** restituita dalla funzione fopen (se ne mostri anche la dichiarazione):

```
f = fopen("esame.txt", "r");
```

Soluzione
La variabile **f** e' un puntatore a file

```
FILE* f;
f = fopen("esame.txt", "r");
```

Esercizio (sintesi, due esercizi collegati)

Si scriva un programma C che legga due serie di dati e li memorizzi in due vettori di strutture.

Nel primo vettore FILM (di dimensione 3) vengono memorizzate strutture (**struct film**) del tipo:
- titolo (stringa di lunghezza 20 non contenente spazi bianchi)
- codice film (intero)
- costo (intero)
- numero giorni di riprese

Nel secondo vettore ATTORI (di dimensione 5) vengono memorizzate strutture (**struct attore**) del tipo:
- nome (stringa di lunghezza 20)
- codice film (intero)
- costo al giorno (intero)

Si scriva un programma che:
- tramite due procedure **leggifilm** e **leggiattore** legga a terminale i dati da inserire nei due vettori;

```
void leggifilm(int n, struct film F[]);
void leggiattore(int n, struct attore A[]);
```

- tramite la procedura **aggiornacosto** aggiorni il costo del film sommando a questo il costo di ciascun attore che partecipa al film. Il costo dell'attore viene calcolato come costo al giorno per il numero di giorni delle riprese.

```
void aggiornacosto(struct film F[], int n, struct attore A[], int
m);
```

dove **n** è la dimensione del vettore **F[]** e **m** è la dimensione del vettore **A[]**

Soluzione

```
#include <stdio.h>
#define DIMA 4
#define DIMF 2

struct film{
       char titolo[20];
       int codice;
       int costo;
       int giorni;};

struct attore{
        char nome[20];
       int codicefilm;
       int costogiorno;};

void leggifilm(int n, struct film F[]);
void leggiattore(int n, struct attore A[]);
void aggiornacosto(struct film F[], int n, struct attore A[], int m);

main()
{int i;
 struct film FILM[DIMF];
 struct attore ATTORI[DIMA];
 leggifilm(DIMF,FILM);
 leggiattore(DIMA,ATTORI);
 aggiornacosto(FILM,DIMF,ATTORI,DIMA);
 }

void leggifilm(int n, struct  film Vet[])
{int i;
 for(i=0;i<n;i++){
     printf("Inserisci Codice, Titolo, Costo e Numero giorni \n");
     scanf("%d",&Vet[i].codice);
     scanf("%s",Vet[i].titolo);
     scanf("%d",&Vet[i].costo);
     scanf("%d",&Vet[i].giorni);
 }
}

void leggiattore(int n, struct attore Vet[]) {
  int i;
  for(i=0;i<n;i++){
    printf("Inserisci nome, codicefilm, costo al giorno\n");
    scanf("%s",Vet[i].nome);
    scanf("%d",&Vet[i].codicefilm);
    scanf("%d",&Vet[i].costogiorno);
 }
}
```

```
void aggiornacosto(struct film F[], int n, struct attore A[], int m){
  int i,j;

  for (i = 0; i < n; i++)
    for (j = 0; j < m; j++)
      if (A[i].codicefilm == F[j].codice){
            F[j].costo = F[j].costo + A[i].costogiorno*F[j].giorni;
  printf("\nCosto aggiornato del film %s: %d", F[j].titolo, F[j].costo);
      }
}
```

Esercizio (analisi)

Dato il seguente programma C:

```
#include <stdio.h>
#define Dim 5

int calc(int *N, int p)
{int i, k=Dim;
 for(i=0 ;i<Dim; i=i+p)
        {       N[i]=p-N[i];
                k=k-i;
        }
p = p + 1;
return k;
}

main()
{ int Quad[Dim]={1,4,9,16,25};
  int i, j = 2;
  for (i=0; i<Dim; i++)
      Quad[i] = i+j - Quad[i];
  printf("%d\n", calc(Quad,j));
  for(i=0; i<Dim; i++)
        printf("%d\t", Quad[i]);
  printf("\n%d\n", j);
}
```

dire che cosa viene stampato dal programma, con le opportune motivazioni.
Si dica poi se la variabile `k` è visibile dal `main()`.

Soluzione

Dopo il primo ciclo for del main, il vettore Quad contiene i seguenti valori

`{1, -1, -5, -11, -19}`

Dopo di che viene chiamata la funzione calc che riceve come parametro attuale (per indirizzo) il vettore Quad e la variabile intera j = 2. Nel ciclo for della funzione calc il vettore viene modificato nel seguente modo:

- Al primo passo l'elemento di indice 0 viene sottratto al valore 2 e la variabile k rimane uguale a 5
- Al secondo passo, l'elemento di indice 2 viene sottratto a 2 e la variabile k diventa uguale a 3
- Al terzo passo, l'elemento di indice 4 viene sottratto a 2 e la variabile k diventa uguale a -1.

k viene restituita dalla funzione calc e quindi la prima stampa del programma main fornisce in uscita **-1**. Viene poi stampato il vettore Quad modificato (perché passato per indirizzo) ed infine j che non viene modificata dalla funzione ric.

Risultato stampato dal programma

```
-1
1    -1   7    -11  21
2
```

Esercizio (domanda)

Si dicano le analogie e differenze tra le funzioni **`fscanf`** e **`scanf`**.

Soluzione

Entrambe sono funzioni che permettono di leggere dati da rispettivamente un file di testo (passato come parametro alla fscanf) e la seconda da standard input. Entrambe inoltre hanno come parametri una stringa di formato e i valori da leggere.

Esercizio (domanda)

Se s1 e s2 sono due stringhe e si scrive s1=s2; cosa succede?

A. Tutto il contenuto di s2 viene copiato in s1
B. Si ottiene un errore di compilazione
C. Il primo elemento di s2 viene ricopiato nel primo elemento di s1

Soluzione

B. Si ottiene un errore di compilazione

Esercizio (sintesi)

Si vuole realizzare un programma che data da input una sequenza di N parole (di, al massimo, 20 caratteri ciascuna), li memorizzi in una struttura dati dinamica e poi stampi la loro lunghezza.

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

typedef char parola[20];

main()
{     parola  *p;
      int i,  N;

      printf("Quante parole? ");
      scanf("%d", &N);
      /* allocazione del vettore */
      p=(parola *)malloc(N*sizeof(parola));
      /* lettura della sequenza */
      for(i=0; i<N; i++) scanf("%s", p[i]);
      for(i=0; i<N; i++) printf("\n%d", strlen(p[i]));
      free(p); /* deallocazione */
}
```

Esercizio (sintesi, 3 esercizi in uno)

Sia dato il file di testo "*dati.txt*" contenente i dati relativi agli studenti immatricolati al primo anno della Facolta' di Ingegneria.

In particolare, le informazioni sono memorizzate nel file "*dati.txt*" come segue:

ognuna delle linee del file contiene i dati relativi ad un nuovo studente ed in particolare:

- **Matricola**: un intero che indica il numero di matricola dello studente;
- **CdL**: un intero che indica il corso di laurea (CdL) dello studente (es: 2145);

Sia dato un secondo file binario "*indirizzi.bin*" che contiene, invece, l'indirizzo di ogni studente, e in particolare:

- **Matricola**: il numero di matricola dello studente;
- **Nome**: il nome dello studente;
- **Cognome**: il cognome dello studente;
- **Via**: una stringa che riporta la via di residenza dello studente;
- **Citta`**: una stringa che riporta la citta` di residenza dello studente;
- **CAP**: un intero che rappresenta il codice di avviamento postale dello studente.

Si scriva un programma in linguaggio C che:

1. A partire dai file "dati.txt" e "indirizzi.bin" costruisca una tabella T contenente, per ogni studente, Matricola, Nome, Cognome, Via, Citta`, CAP e CdL.

2. A partire dalla tabella T, e dato da input un intero C che rappresenta un CdL, stampi la percentuale di studenti (rispetto al numero totale delle matricole) iscritti al corso C. [Ad esempio, se il numero totale delle matricole e` 1000, e quello degli studenti iscritti a C e` 200, il programma stampera` "20%"]

3. Scriva su un terzo file di testo "bologna.txt", nome, cognome e numero di matricola di tutti gli studenti che abitano a Bologna.

Soluzione:

```
#include <stdio.h>
#include <string.h>
/* tipi di dato */
typedef struct {      unsigned int matr;
                              unsigned CDL;
               }dati;

typedef struct {unsigned int matr;
                              char nome[20];
                              char cognome[30];
                              char via[30];
                              char citta[30];
                              unsigned int CAP;
                         } indirizzo;

typedef struct {    unsigned int matr;
                              char nome[20];
                              char cognome[30];
                              char via[30];
                              char citta[30];
                              unsigned int CAP;
                              unsigned int CDL;
                         } elemento;

typedef  elemento tabella[10];

elemento riempiel( dati d, indirizzo i);

/* le seguenti funzioni servono solo per predisporre e visualizzare il
file di  indirizzi:*/

void creafile(char *b);
void vedifile(char *b);
/* fine funzioni file */

main()
{     dati D;
      indirizzo I;
      elemento E;
      tabella T;
      FILE *f1, *f2;
      int i, trovato,ins=0, totC;
      unsigned int C;

      /*non necessario: creaz. del file binario */
      printf("creare il file (0/1)??");
      scanf("%d", &i);
      if (i==1)
            creafile("indirizzi.bin");
      else vedifile("indirizzi.bin");


/*domanda 1: costruzione della tabella */

      f1=fopen("dati.txt", "r");
      f2=fopen("indirizzi.bin", "rb");


      while (fscanf(f1,"%u%u", &D.matr, &D.CDL)>0)
      {     trovato=0;
            rewind(f2);
            while(fread(&I,sizeof(indirizzo),1,f2)>0
                        && !trovato)
                  if (I.matr==D.matr) /*ho trovato
                                                l'indirizzo
                                                   dello stud. D */
                  {     trovato=1;
                        E=riempiel(D, I);
                        T[ins]=E;
                        ins++;
                  }
      }

      fclose(f1);
      fclose(f2);

      /*domanda 2: stampa della percentuale degli
      iscritti a un corso dato*/
      printf("Inserire il corso C: ");
      scanf("%u", &C);
      totC=0;
      for(i=0; i<ins; i++)
            if(T[i].CDL==C)
                  totC++;

      printf("\n Iscritti al corso %u: %f \%\n",
                  C, (float)totC*100/ins);


      /*domanda 3: scrittura di "bologna.txt" */
      f1=fopen("bologna.txt", "w");
      for (i=0; i<ins; i++)
            if (strcmp("bologna", T[i].citta)==0)
                  fprintf(f1, "%s %s %u\n",
                  T[i].nome, T[i].cognome, T[i].matr);
      fclose(f1);

}
```

```
elemento riempiel(dati d, indirizzo i)
{  elemento e;
     /*copia in e il contenuto di d e di i*/
     e.matr=d.matr;
     e.CDL=d.CDL;
     strcpy(e.nome, i.nome);
     strcpy(e.cognome, i.cognome);
     strcpy(e.via, i.via);
     strcpy(e.citta, i.citta);
     e.CAP=i.CAP;
   return e;
}
void creafile(char *v)

{ FILE *f; indirizzo e;int fine=0;

f=fopen(v, "wb");

printf("creazione di %s...\n", v);

while (!fine)
     {    printf("matricola");
          scanf("%u", &e.matr);
      printf("\nCAP ? " );
          scanf("%u", &e.CAP);
          printf("\nCognome ? ");
          scanf("%s", &e.cognome);
          printf("\nNome ? ");
          scanf("%s", &e.nome);
          printf("\nCitta`? ");
          scanf("%s", &e.citta);
          printf("\nVia ? ");
          scanf("%s", &e.via);
          fflush(stdin);
          fwrite(&e, sizeof(indirizzo), 1, f);
          printf("\nFine (SI=1, NO=0) ? ");
          scanf("%d", &fine);
     } fclose(f); }
void vedifile(char *v)
{FILE *f; indirizzo e;int fine=0;
f=fopen(v, "rb");
printf("Lettura di %s:\n", v);
fread(&e, sizeof(indirizzo), 1, f);
while (!feof(f))
     {    printf("%u\t", e.matr);
          printf("%s\t", e.cognome);
          printf("%s\t", e.nome);
          printf("%s\t", e.via);
          printf("%s\n", e.citta);
          printf("%u\t", e.CAP);
          fread(&e, sizeof(indirizzo), 1, f);


     } fclose(f); }
```

Esercizio (analisi)

Dato il seguente programma:

```
#include <stdio.h>
#define DIM 6

int p(int a)
{ if (a%2==0)
     return 0;
  else return a+1;
}

int f(int *a, int b)
{
     if (a[b]!=0)
          return a[b]=5;
     else return p(b+1)+b;
}



main()
{    int A[DIM]={0,0,0,0,0,0};
     int i;
     for(i=0; i<DIM; i+=2)
          A[i]=i;
     printf("%d\n", f(A,0));
     for(i=0; i<DIM; i++)
          printf("%d\t",A[i]);
}
```

Si indichino, nel giusto ordine, i valori stampati dal programma, motivando la risposta data.

Soluzione:

```
2
0          0          2          0          4          0
```

Esercizio (domanda)

Supponiamo che sia **int x = 11;**

Che differenza c'è fra

```
fprintf(file,"%d", x);
```

e

```
fwrite(&x, sizeof(int), 1, file);
```

se il codice ASCII del carattere 1 e' 00110001?

Soluzione

`fprintf` emette due bytes, ciascuno corrispondente al codice ASCII del carattere 1: `00110001 00110001`
`fwrite` emette (su 16 bit) due bytes che rappresentano 11 in notaziobe binaria: `00000000 00001011`

Esercizio (sintesi)
Dato un file di binario **`mesi.dat`**, si supponga che contenga (in rappresentazione interna) strutture cosi' configurate: una stringa (nome del mese) ed un intero (numero di giorni). Ad esempio:

```
gennaio   31
febbraio  28
marzo     31
aprile    30
```

Si memorizzi il contenuto del file in un vettore di strutture e si stampino a video i nomi dei mesi che hanno 31 giorni.

Esercizio (soluzione)

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

main() {
int i;
struct mese {int giorni; char nome[20];} v[12];
FILE* f;
if ((f=fopen("mesi.dat", "rb"))==NULL) {
   printf("Il file non esiste!"); exit(1); }
 while(fread(&v[i],sizeof(struct mese),1,f)>0){
           if (v[i].giorni == 31) printf("%s\n", v[i].nome);
           i++;
           }

   fclose(f);
}
```