Esercizi di Preparazione alla Prima Prova Intermedia
Fondamenti di Informatica L-A (Proff. Paola Mello e Paolo Bellavista)

La prima prova intermedia potrà includere:

- esercizi di analisi;
- esercizi di sintesi;
- esercizi su record di attivazione;
- piccole domande varie, ad esempio su rappresentazione dei numeri e grammatiche.

ESERCIZIO (Analisi)

Si indichino i valori stampati dal seguente programma C, motivando la risposta data. Indicare quali sono i blocchi in cui è visibile la variabile N (motivare la risposta).

```
#include <stdio.h>
#define L 7

int N=L;
void  P(char* VET[], int DIM);

main () {

     char* M[L] = {"pippo","pluto","paperino","qui","quo","qua",
          "minnie"};
     int  i;

     P(M,L-2);
     for (i = L-1; i-->0; ) printf("%s\n",*(M+i));
}

void  P(char* VET[], int DIM) {

     int i;
     N++;
     for (i=0; i<DIM-1; i=i+2)  strcpy(VET[i],VET[i+1]);
     return;
}
```

Soluzione
Il programma stampa:
qua
quo
qui
qui
pluto
pluto

La variabile N è visibile in tutti i blocchi, perché è definita nell'ambiente globale.

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
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 (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 (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 restituisce 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 (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 (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 (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.

Soluzione:

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

Esercizio (sintesi)
Si definisca una struttura mese, caratterizzata dai campi nome_mese (stringa, non più di 16 caratteri compreso il terminatore), e da lunghezza_mese (intero, rappresentante i giorni di tale mese).

Si scriva poi una funzione seleziona che, ricevuto come parametro di ingresso un array di strutture mese ed un array rappresentatne la lunghezza di tale array, stampi a video i nomi dei mesi che hanno 31 giorni.

Si definisca poi un programma main che chieda all'utente di inserire il nome di un mese ed il numero di giorni che lo compongono (chieda tali informazioni al massimo 12 volte), memorizzi le risposte in un array preparato precedentemente, e stampi a video i mesi di 31 giorni tramite la funzione seleziona.

Soluzione

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

struct mese {
    char nome_mese[16];
    int lunghezza_mese;
};


void seleziona(struct mese meseArray[], int lung) {
    int i;

    for (i=0; i< lung; i++)
      if (meseArray[i].lunghezza_mese == 31)
            printf("%s ha 31 giorni.\n", meseArray[i].nome_mese);
}


int main() {
    struct mese mesiLetti[12];
    int i;

    for (i=0; i<12; i++) {
      printf("Inserire primo mese: ");
      scanf("%s", mesiLetti[i].nome_mese);
      printf("Inserire giorni: ");
      scanf("%d", &(mesiLetti[i].lunghezza_mese) );
    }

    seleziona(mesiLetti, 12);
    return 0;
}
```

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 passati 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 mostri anche un esempio di chiamata della procedura.

Soluzione

```
#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 (sintesi) "La banca"

Una banca vuole realizzare un programma di simulazione di gestione conto-corrente.

A tal scopo il programma simula l'esistenza di un conto corrente, inizialmente vuoto, e offre un menù all'utente in cui egli può scegliere di compiere 4 operazioni diverse: **deposito, prelievo, visualizzazione del saldo, calcolo degli interessi composti**. L'utente sceglie l'operazione specificando un numero (o un carattere); sempre tramite un numero o un carattere segnala l'intenzione di terminare la simulazione. Dopo ogni operazione il programma ristampa a video l'elenco delle operazioni possibili ed attende un nuovo comando dall'utente.

- L'operazione di deposito consiste nel chiedere all'utente una somma, controllare che sia > 0, e depositarla nel conto.
- L'operazione di prelievo invece consiste sempre nel chiedere la somma all'utente, ma è ovviamente necessario controllare che sia > 0 e che ci siano fondi a sufficienza nel conto corrente.
- L'operazione di visualizzazione saldo consiste semplicemente nel visualizzare il saldo del conto corrente. Per comodità questa operazione deve essere eseguita automaticamente al termine delle operazioni di deposito e prelievo.
- L'operazione di calcolo degli interessi consiste invece nel calcolare a quanto ammonterebbe il capitale finale qualora si avesse il capitale iniziale pari ai soldi depositati sul conto, per **N** anni con un tasso di interesse pari a **r**. Il programma deve quindi chiedere all'utente il tasso di interesse ed il numero di anni su cui simulare. Ovviamente è necessario controllare che i dati inseriti corrispondano a valori plausibili (tasso d'interesse >= 0 e <= 100) (anni > 0).

La formula per il calcolo degli interessi composti, dove r è il tasso di rendimento e N è il numero di anni a cui gli interessi sono applicati, è la seguente:

$$Cfin = Cin * \left(1 + \frac{r}{100}\right)^{N}$$

Soluzione

```
#include <stdio.h>

#define DEPOSITO 1
#define PRELIEVO 2
#define SALDO 3
#define INTERESSI 4
#define FINE 0

void stampa_menu();
void f_deposito(float * soldi);
void f_prelievo(float * soldi);
void f_saldo(float soldi);
void f_interessi(float soldi);
float interessi(float c_in, float tasso, int anni);
```

```
main() {
   int op;
   float soldi = 0;

   printf("Programma di simulazione conto corrente\n\n");
   stampa_menu();

   printf("Scegliere operazione: ");
   scanf("%d", &op);
   while (op != FINE) {
      switch (op) {
         case DEPOSITO:
            f_deposito(&soldi); break;
         case PRELIEVO:
            f_prelievo(&soldi); break;
         case SALDO:
            f_saldo(soldi); break;
         case INTERESSI:
            f_interessi(soldi); break;
         default:
            printf("Operazione non consentita");
      }
      stampa_menu();
      printf("Scegliere operazione: ");
         scanf("%d", &op);
   }
}

void stampa_menu() {
      printf("Operazioni disponibili:\n");
      printf("%d.\tDeposito\n", DEPOSITO);
      printf("%d.\tPrelievo\n", PRELIEVO);
      printf("%d.\tSaldo\n", SALDO);
      printf("%d.\tCalcolo interessi\n", INTERESSI);
      printf("%d.\tFine\n", FINE);
}

void f_deposito(float * soldi) {
      float cash;

      printf("Digitare l'ammontare del deposito: ");
      scanf("%f", &cash);
      while (cash < 0) {
            printf("Ammontare negativo.\n");
            printf("Digitare l'ammontare del deposito: ");
            scanf("%f", &cash);
      }
      *soldi = *soldi + cash;
      f_saldo(*soldi);
}

void f_prelievo(float * soldi) {
      float cash;

      printf("Digitare l'ammontare del prelievo: ");
      scanf("%f", &cash);
      while (cash < 0) {
            printf("Ammontare negativo.\n");
```

```
            printf("Digitare l'ammontare del prelievo: ");
            scanf("%f", &cash);
      }
      while (cash > *soldi) {
            printf("Nel conto non ci sono cosi' tanti soldi.\n");
            printf("Digitare l'ammontare del prelievo: ");
            scanf("%f", &cash);
      }
      *soldi = *soldi - cash;
      f_saldo(*soldi);
}

void f_saldo(float soldi) {
      printf("Disponibilita' nel conto corrente: %6.2f\n", soldi);
}

void f_interessi(float soldi) {
      float tasso, c_fin;
      int anni;

      printf("Inserire il tasso d'interesse: ");
      scanf("%f", &tasso);
      while ((tasso<0) || (tasso>100)) {
            printf("Tasso errato\n");
            printf("Inserire il tasso d'interesse: ");
            scanf("%f", &tasso);
      }
      printf("Inserire anni: ");
      scanf("%d", &anni);
      while (anni<0) {
            printf("Anni negativi.\n");
            printf("Inserire anni: ");
            scanf("%d", &anni);
      }

      c_fin = interessi(soldi, tasso, anni);
      printf("Interessi simulati, con:\n");
      printf("Capitale iniziale: %6.2f\n", soldi);
      printf("Tasso d'interesse: %6.2f\n", tasso);
      printf("Anni di applicazione degli interessi: %d\n", anni);
      printf("Capitale finale: %6.2f\n", c_fin);
}

float interessi(float c_in, float tasso, int anni)
{
      float tasso_perc, tasso_tot, result=1;
      int i;

      tasso_perc = tasso/100;
      tasso_tot = 1 + tasso_perc;

      for (i=0; i<anni; i++)
            result=result*tasso_tot;

      return c_in*result;
}
```

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 (sintesi)
Si scriva un programma C che legga due serie di dati e le 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[j].codicefilm == F[i].codice){
           F[i].costo = F[i].costo + A[j].costogiorno*F[i].giorni;
  printf("\nCosto aggiornato del film %s: %d", F[i].titolo,
F[i].costo);
      }
}
```

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)
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 (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)
Siano date due stringhe char s1[]="Pippo", s2[20];
Se si scrive s1=s2; che 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.