

Laboratorio 28/03/03

strutture dati:
vettori
record e union

Caricare un vettore

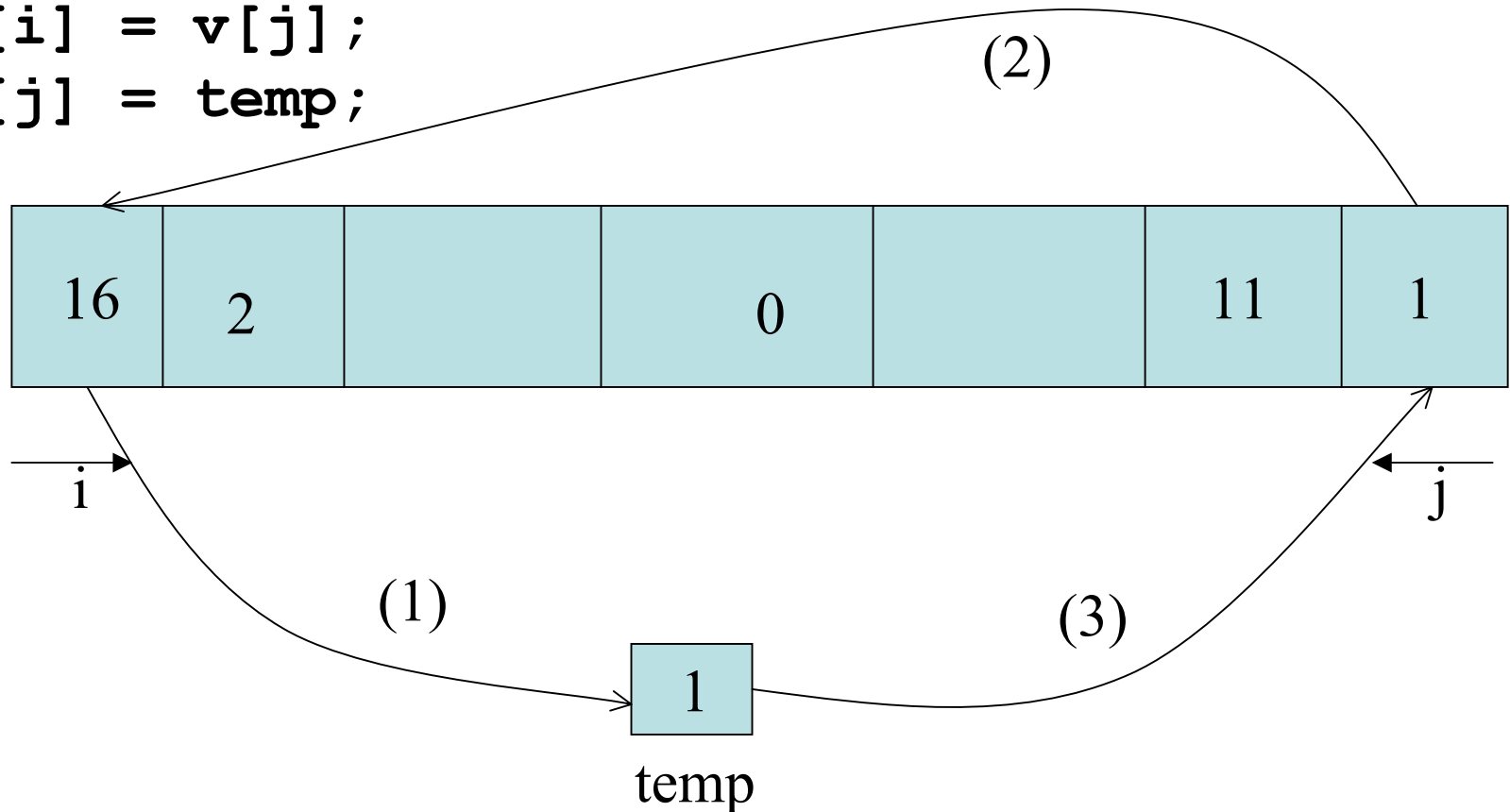
```
#include <stdio.h>
#define DIM 7
typedef double VtDouble[DIM];
int main()
{ int i;  VtDouble VD;
  for(i=0;i<DIM;i++)
    scanf("%lf",&VD[i]);
  for(i=0;i<DIM; i++)
    printf("%6.3f\n",VD[i]);
  return(0);
}
```

Esercizio

- Scrivere il programma che:
 - carica un vettore v di N elementi interi
 - scambia il primo elemento con l'ultimo, il secondo con il penultimo e così via
 - stampa il vettore

Esempio per la soluzione

```
temp = v[i];  
v[i] = v[j];  
v[j] = temp;
```



Soluzione

```
#include <stdio.h>
#define N 4
typedef int VT[N];

main()
{ VT v; int i,j,temp;
  for(i=0;i<N;i=i+1) /*carica il vettore*/
    scanf("%d",&v[i]);

  /*scambio posizioni*/
  i=0;j=N-1;
  while (i < j)
    {temp=v[i]; v[i] = v[j]; v[j] = temp;
      i++; j--;}

  for(i=0;i<N;i++) /*stampa*/
    printf("%d\t",v[i]);}
```

Ordinare un vettore

- Idea del metodo di selezione diretta:
 - scegliere l'elemento minore del vettore e metterlo nella prima posizione
 - considerare il vettore a partire dalla seconda posizione e scegliere l'elemento minore, porlo in seconda posizione
 - considerare il vettore a partire dalla terza posizione e scegliere l'elemento minore, porlo in terza posizione
 - ...

Ordinare vettore (sol.: 1)

```
#include <stdio.h>
#define DIM 7
typedef int VtInt[DIM];
int main()
{ int i;
  VtInt Vt;
  /*caricare il vettore*/
  /*ordinare il vettore*/
  /*stampare il vettore*/
  return(0);
}
```

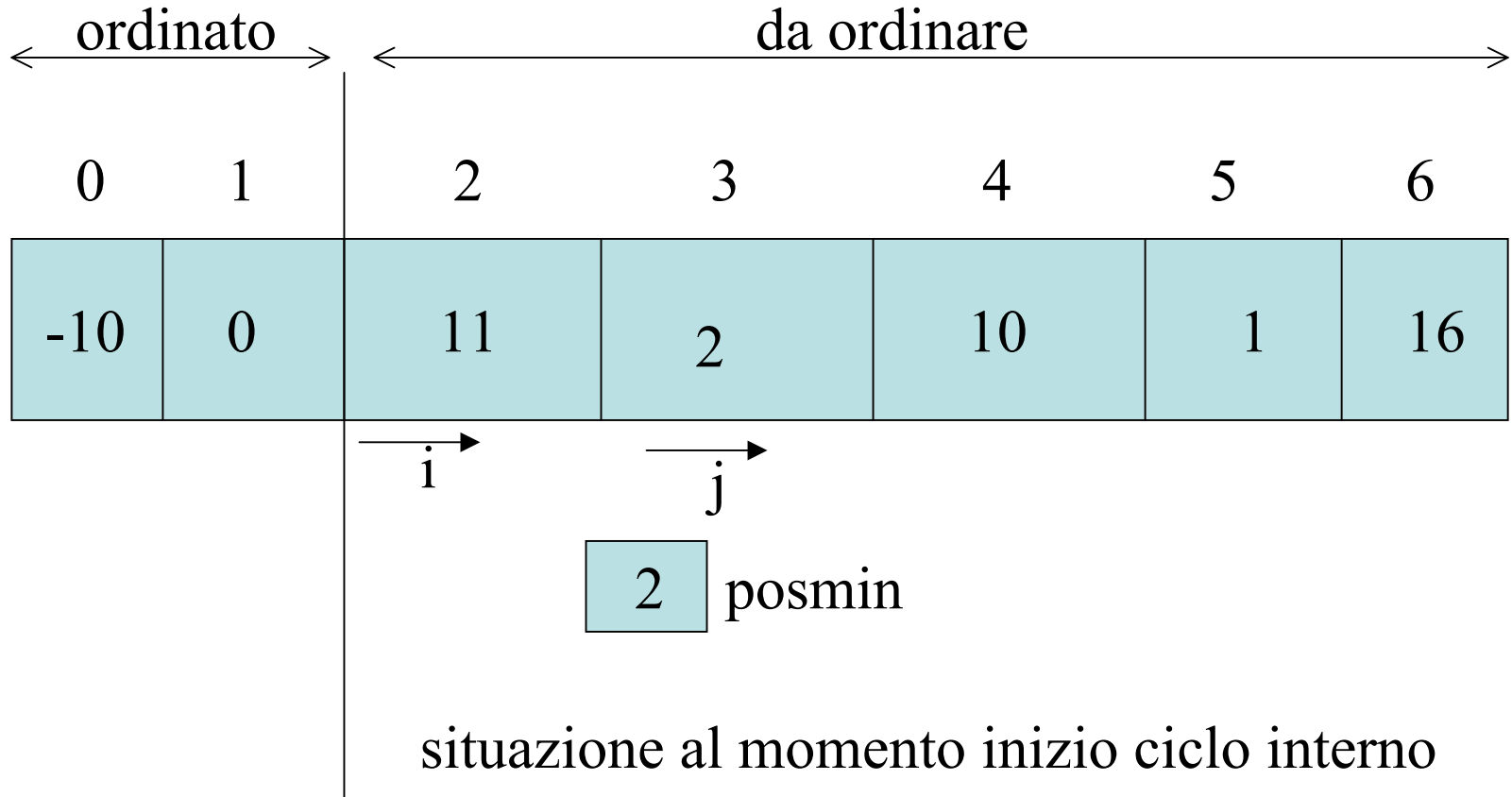
Ordinare vettore (sol.: 2)

```
#include <stdio.h>
#define DIM 7
typedef int VtInt[DIM];
int main()
{ int i;
  VtInt Vt;
  for(i=0;i<DIM;i++) /*caricare*/
    scanf("%d",&Vt[i]);
  /*ordinare*/
  for(i=0;i<DIM; i++) /*stampare*/
    printf("%d\t",Vt[i]);
  printf("\n"); return(0);}
```


Ordinare vettore (sol.: 3)

```
/*ordinare: aggiungere le variabili mancanti*/
for(i=0; i<DIM-1; i++)
    {posMin = i; /*posizione iniziale el. minore*/
      /*determina posizione elemento minore
        fra i+1 e DIM*/
      for(j=i+1; j<DIM; j++) {
          if (v[j] < v[posMin])
              posMin = j;
      }
      /*in posMin esiste posizione el. minimo*/
      /*scambio valori fra posMin ed i*/
    }
```

Esempio



Ordinare vettore (sol.: 4)

```
/*ordinare: aggiungere le variabili mancanti*/
for(i=0; i<DIM-1; i++)
    {posMin = i; /*posizione iniziale el. minore*/
    for(j=i+1; j<DIM; j++) /*determ. minore*/
        if(Vt[j] < Vt[posMin])
            posMin = j;
    temp = Vt[i]; /*scambio valori*/
    Vt[i] = Vt[posMin];
    Vt[posMin] = temp;
    }
```

Esercizio

- leggere due stringhe (uno, due) e poi:
 - dire quale e' la stringa piu' lunga
 - se uno contiene due
 - se uno precede, segue o eguale a due

Esercizio

- scrivere un programma che
 - legge la stringa **ini** e la stringa **parte** e una terza stringa **base**. Quindi:
 - indica la lunghezza di **base**;
 - stampa se **base** e' uguale o no alla stringa **ini**;
 - accoda a **base** il suffisso "CIAO" e stampa il risultato
 - verifica se **base** contiene o no la stringa **parte**.

Soluzione (1/2)

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

int main()
{ char ini[10], parte[5], base[14];

  printf("dammi la stringa ini: ");
  gets(ini);
  printf("dammi la parte: ");
  gets(parte);
  printf("dammi la parte: ");
  gets(base);

  /*----> segue*/
```

Soluzione (2/2)

```
printf("lung. stringa: %d\n",strlen(base));

if(strcmp(base,ini) == 0)
    printf("|%s| = |%s|\n",base, ini);
else
    printf("|%s| != |%s|\n",base, ini);

strcat(base,"CIAO");
printf("%s\n",base);
if(strstr(base,parte) != 0)
    printf("%s contiene %s\n",base, parte);
else
    printf("%s non contiene %s\n",base, parte);
}
```

Struct

```
#include <stdio.h>
typedef struct{int a,b;double c;} AS;
main()
{ AS A;
  printf("dammi a: ");
  scanf("%d",&A.a);
  printf("dammi b: ");
  scanf("%d",&(A.b));
  A.c = (double) A.a/A.b;
  printf("risultato: %lf\n",A.c);
  return(0); }
```


Esercizio

- Scrivere un programma che carica i campi a e b di un vettore di N strutture S denominato $\forall t S$ e quindi pone nel campo c il valore della somma del quadrato dei campi a e b.

```
typedef struct{double a,b,c;} S;
```

```
typedef S VS[N];
```

usare la funzione di `<math.h>`

```
pow(base,esp);
```

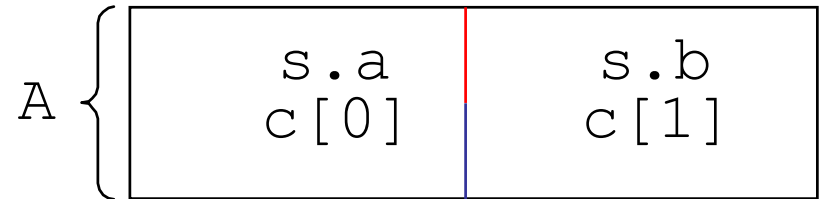
Soluzione

```
#include <stdio.h>
#include <math.h>
#define N 5
typedef struct{double a, b, c;} S;
typedef S VS[N];
main()
{VS VtS;    int i;
  /*carica i campi a e b*/
  for(i=0;i<N;i++)
    {printf("dammi due numeri: ");
     scanf("%lf%lf",&(VtS[i].a),&(VtS[i].b));}
  /*carica il campo c*/
  for(i=0;i<N;i++)
    VtS[i].c = pow(VtS[i].a,2) + pow(VtS[i].b,2);
  /*stampa*/
  for(i=0;i<N;i++)
    printf("%f\t%f\t%f\n",VtS[i].a,VtS[i].b,VtS[i].c);
}
```

Union

```
#include <stdio.h>
typedef union {struct {int a,b;} s;int
  c[2];} Aut;
```

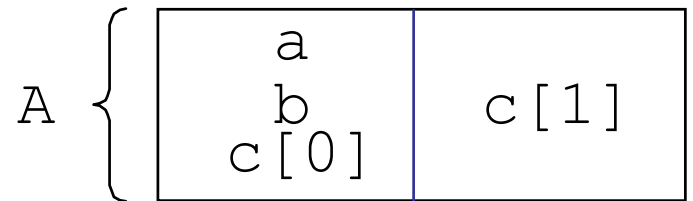
```
main()
{ Aut A; int i;
  printf("dammi a: ");
  scanf("%d",&(A.s.a));
  printf("dammi b: ");
  scanf("%d",&(A.s.b));
  for(i=0;i<2;i=i+1)
    printf("A.c[%d]=%d\n",i,A.c[i]);
  return(0);
}
```



Union

```
#include <stdio.h>
typedef union {int a,b; int c[2];} Aut;
```

```
main()
{ Aut A; int i;
  printf("dammi a: ");
  scanf("%d",&(A.a));
  printf("dammi b: ");
  scanf("%d",&(A.b));
  for(i=0;i<2;i=i+1)
    printf("A.c[%d]=%d\n",i,A.c[i]);
  return(0);
}
```



Esercizio

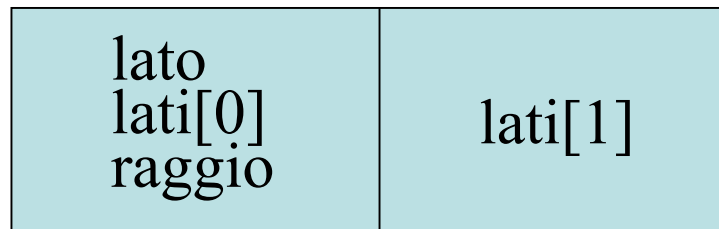
- Si deve caricare un vettore di N elementi contenente in alternativa:
 - il lato di un quadrato
 - i due lati di un rettangolo
 - il raggio di una circonferenza

Successivamente, per ogni elemento caricato, si devono riempire due appositi campi dell'elemento con il perimetro e l'area della figura.

Soluzione

Definizione struttura dati:

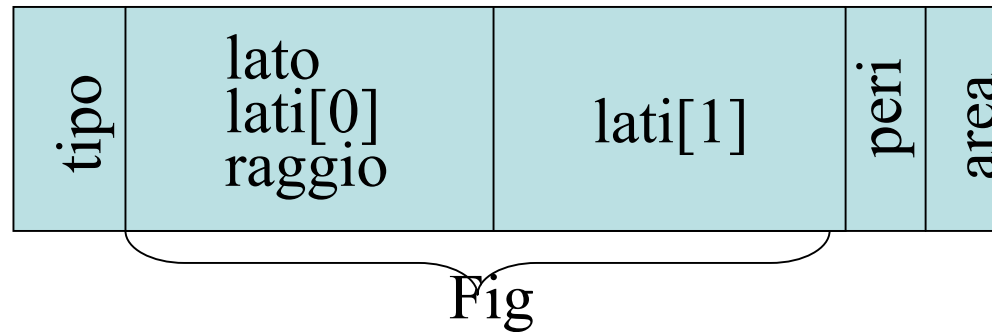
```
#define DIM 3
typedef union {double lato, raggio,
               lati[2];} Dimens;
```



--> CONTINUA

Soluzione

```
typedef struct {int tipo; Dimens Fig;  
                double peri, area;} FIGURA;  
/*tipo =0: quadrato; =1 rettangolo; =2  
  circonferenza*/
```



```
typedef FIGURA VtFg [DIM] ;
```

Soluzione

```
int main()  
{ VtFg Vett;  int i;  
  
    /*caricamento struttura*/  
  
    /*calcolo perimetri e aree*/  
  
    /*stampa finale*/  
    return(0);  
}
```


Soluzione

```
/*caricamento struttura*/
for(i=0; i<DIM; i++)
    {printf("che figura è: ");
      scanf("%d", &Vett[i].tipo);
      if (Vett[i].tipo == 0)
          {printf("dammi il lato: ");
            scanf("%lf", &Vett[i].Fig.lato);}
      else if (Vett[i].tipo == 1)
          {printf("dammi i lati: ");
            scanf("%lf%lf", &Vett[i].Fig.lati[0], \
                  &Vett[i].Fig.lati[1]);}
      else
          {printf("dammi il raggio: ");
            scanf("%lf", &Vett[i].Fig.raggio);}
    }
```

Soluzione

```
/*calcolo*/
  for(i=0; i<DIM; i++)
    if(Vett[i].tipo == 0)
      {Vett[i].peri = 4*Vett[i].Fig.lato;
       Vett[i].area = Vett[i].Fig.lato*\
          Vett[i].Fig.lato;}
    else if (Vett[i].tipo == 1)
      {Vett[i].peri=
        2*Vett[i].Fig.lati[0]*Vett[i].Fig.lati[1];
       Vett[i].area = Vett[i].Fig.lati[0]*\
Vett[i].Fig.lati[1];}
    else
      {Vett[i].peri=2*PI*Vett[i].Fig.raggio;
       Vett[i].area=PI*Vett[i].Fig.raggio *\
Vett[i].Fig.raggio;}
```

Esercizi svolti non presentati a lezione

Esercizio

- Scrivere un programma che
 - carica la matrice `Matrix 3 X 4` di `double`
 - determina il massimo elemento in valore assoluto di ogni riga e lo salva nel vettore `VS` di 3 elementi
 - divide tutti gli elementi di ogni riga per il corrispondente elemento della matrice `VT`
 - stampa la matrice ottenuta
- `double fabs(double val):` ritorna il valore assoluto di `val` (uso ***vass=fabs(val)***);

Soluzione (1/2)

```
#include <stdio.h>
#include <math.h>
#define NRighe 3
#define NColon 4
typedef double MATRIX[NRighe][NColon];
typedef double VET[NRighe];
main()
{MATRIX matri;
  VET vs;
  int i,j;
  /*carica la matrice*/
  for(i=0;i<NRighe; i=i+1)
    for(j=0; j<NColon; j=j+1)
      {printf("dammi elemento %d,%d: ",i,j);
        scanf("%lf",&matri[i][j]);}
  /*--->segue pagina successiva*/
```

Soluzione (2/2)

```
/*determina massimo per riga*/
for(i=0;i<NRighe; i++)
    { vs[i] = fabs(matri[i][0]);
      for(j=0;j<NColon; j++)
          if(vs[i]<fabs(matri[i][j]))
              vs[i] = fabs(matri[i][j]);}

/*normalizza*/
for(i=0;i<NRighe; i++)
    for(j=0;j<NColon; j++)
        matri[i][j]=matri[i][j]/vs[i];

/*stampa*/
for(i=0;i<NRighe; i++)
    { for(j=0;j<NColon; j++)
        printf("%6.3f\t",matri[i][j]);
      printf("\n");}
}/*fine main*/
```

Esempio

- Scrivere un programma che gestisce un vettore di elementi formati da un campo `nome` di 8 caratteri ed un campo `cognome` di 10.
 - Caricare il vettore di N elementi
 - elencare tutti gli elementi del vettore che hanno il campo `nome` eguale al valore di una stringa immessa da tastiera

Soluzione (1/2)

```
#define N 5
#define DIMN 9
#define DIMC 11
typedef struct {
    char Nome [DIMN] ,
        Cognome [DIMC] ;
    } REC ;
typedef REC VRC [N] ;
```


Soluzione (2/2)

```
#include <stdio.h>
#include <string.h>
#define N 5
#define DIMN 9
#define DIMC 11
typedef struct {char Nome[DIMN], Cognome[DIMC];}REC;
typedef REC VRC[N];
main()
{int i, t=0; char nm[DIMN]; VRC v;
  for(i=0; i<N; i++)
    { printf("nome: "); gets(v[i].Nome);
      printf("cognome: "); gets(v[i].Cognome);}
  printf("nome da cercare: "); gets(nm);
  for(i=0; i<N; i++)
    if(strcmp(nm,v[i].Nome) == 0)
      {printf("%s\t%s\n", v[i].Nome,v[i].Cognome);
        t=1;}
  if(t==0) printf("non ci sono corrispondenze\n");
}
```