## ESERCIZIO: Lettura/Scrittura Array

Non è possibile leggere/scrivere un intero vettore con un'unica operazione (a parte il caso particolare delle ***stringhe***); occorre leggere/scrivere ognuno dei suoi componenti

```
int main() {
   int i,frequenza[25];
   for (i=0; i<25; i++)
   {   scanf("%d",&frequenza[i]);
       frequenza[i]=frequenza[i]+1;
   }   /*    legge a terminale le componenti del
              vettore frequenza e le incrementa
       */
}
```



## ESERCIZIO: Assegnamento

Anche se due variabili vettore sono dello ***stesso tipo***, NON è possibile l'assegnamento diretto:

```
int F[25], frequenza[25];
F = frequenza;       /* NO */
```

ma ***occorre copiare componente per componente***:

```
for (i=0; i<25; i++)
     F[i] = frequenza[i];
```



## ESERCIZIO: MAX e min di un vettore

```
#define N 15   /* è noto a tutti che la dimensione
                  del vettore è N */

int minimo (int vet[]);
int massimo (int vet[]);

int main ()
{int i, a[N];
 printf ("Scrivi %d numeri interi\n", N);
 for (i=0; i<N; i++)
        scanf ("%d", &a[i]);
 printf ("L'insieme dei numeri è: ");
 for (i=0; i<N; i++)
      printf(" %d",a[i]);
 printf ("Il minimo vale %d e il
          massimo è %d\n", minimo(a), massimo(a));
}
```



## ESERCIZIO: MAX e min di un vettore

```
int minimo (int vet[])
{int i, min;
 min = vet[0];
 for  (i = 1; i<N; i++)
   if (vet[i]<min) min = vet[i];
 return min;
}


int massimo (int vet[])
{int i, max;
 max = vet[0];
 for (i = 1; i<N; i++)
    if (vet[i]>max) max = vet[i];
 return max;
}
```

## ESERCIZIO: Ricerca di un elemento

```
#include <stdio.h>
#define N 15

int ricerca (int vet[], int el);
int main ()
{int i;
 int a[N];
 printf ("Scrivi %d numeri interi\n", N);
 for (i = 0; i<N; i++)
     scanf ("%d", &a[i]);
 printf ("Valore da cercare: ");
 scanf ("%d",&i);
 if (ricerca(a,i)) printf("\nTrovato\n");
     else printf("\nNon trovato\n");
}
```



## ESERCIZIO: Ricerca di un elemento

```
int ricerca (int vet[], int el)
{int  i=0;
 int  T=0;
 while ((i<N) && (T==0))
  { if (el==vet[i]) T=1;
    i++;}
 return T;
}
```

Proposta di esercizio ulteriore: ricercare ***se e quali*** elementi di un vettore ***V1*** di float sono contenuti in un altro vettore ***V2*** di float. Le dimensioni dei due vettori possono essere diverse



## ESERCIZIO: Ricerca di un elemento

Sapendo che il vettore è ***ordinato*** (esiste una relazione d'ordine totale sul dominio degli elementi), la ricerca può essere ottimizzata

- ***Vettore ordinato in senso non decrescente:***

| 2 | 3 | 5 | 5 | 7 | 8 | 10 | 11 |
|---|---|---|---|---|---|---|---|

`se i<j` si ha `V[i]<=V[j]`

- ***Vettore ordinato in senso crescente:***

| 2 | 3 | 5 | 6 | 7 | 8 | 10 | 11 |
|---|---|---|---|---|---|---|---|

`se i<j` si ha `V[i]<V[j]`

In modo analogo si definiscono l'ordinamento in senso ***non crescente*** e ***decrescente***



## ESERCIZIO: RICERCA BINARIA

***Ricerca binaria di un elemento in un vettore ordinato in senso non decrescente*** in cui il primo elemento è `first` e l'ultimo `last`

La tecnica di ***ricerca binaria***, rispetto alla ricerca esaustiva, consente di ***eliminare ad ogni passo metà degli elementi del vettore***

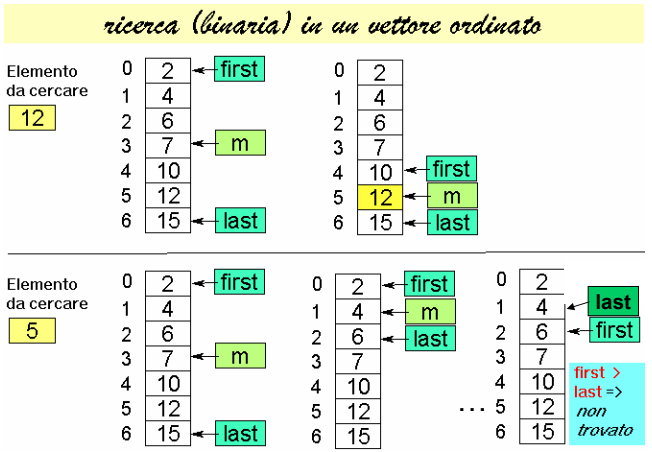
# ESERCIZIO: RICERCA BINARIA

- Si confronta l'elemento cercato **`el`** con quello mediano del vettore, **`V[med]`**
- Se **`el==V[med]`**, fine della ricerca (**`trovato=true`**)
- Altrimenti, se il vettore ha almeno due componenti (**`first<=last`**):
  - se **`el<V[med]`**, ripeti la ricerca nella prima metà del vettore (indici da **`first`** a **`med-1`**)
  - se **`el>V[med]`**, ripeti la ricerca nella seconda metà del vettore (indici da **`med+1`** a **`last`**)



# ESERCIZIO: RICERCA BINARIA

*Esempio*






# ESERCIZIO: RICERCA BINARIA

```
int ricerca_bin (int vet[], int el)
{int first=0, last=N-1, med=(first+last)/2;
 int T=0;
 while ((first<=last)&&(T==0))
  { if (el==vet[med])
          T=1;
    else
       if (el < vet[med])  last=med-1;
       else   first=med+1;
    med = (first + last) / 2;
  }
 return  T;
}
```



# ESERCIZIO: Ricerca binaria di un elemento

```
#include <stdio.h>
#define N 15

int ricerca_bin (int vet[], int el);
int main ()
{int i;
 int a[N];
 printf ("Scrivi %d numeri interi ordinati\n", N);
 for (i = 0; i<N; i++)
     scanf ("%d", &a[i]);
 printf ("Valore da cercare: ");
 scanf ("%d",&i);
 if (ricerca_bin(a,i)) printf("\nTrovato\n");
     else printf("\nNon trovato\n");
}
```

# OSSERVAZIONI

Si noti che la ricerca binaria può essere definita facilmente in ***modo ricorsivo***

Si noti infatti che si effettua un ***confronto dell'elemento cercato `el` con l'elemento di posizione media del vettore `V[med]`***

- Se l'elemento cercato è uguale si termina (caso base)
- Altrimenti se **`el < V[med]`** si effettua una ricerca binaria sulla prima metà del vettore
- Altrimenti (**`se el > V[med]`**) si effettua una ricerca binaria sulla seconda metà del vettore

Esercizio: si scriva procedura per ***ricerca binaria ricorsiva***