ESERCIZIO: Input Output

- Non è possibile leggere/scrivere un intero vettore (a parte come vedremo le ***stringhe***); occorre leggere/scrivere le sue componenti:

```
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
  /* dichiarazione di due funzioni */
int minimo (int vet[]);
int massimo (int vet[]);

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);
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;
}
```

ESERCIZIO: Ricerca di un elemento

- Sapendo che il vettore è ***ordinato***, la ricerca può essere ottimizzata.
  - ***Vettore ordinato in senso non decrescente:***
    - Esiste una relazione d'ordine totale sul dominio degli elementi del vettore e:
    - **`i<j`** si ha **`V[i]<=V[j]`**

2	3	5	5	7	8	10	11

  - ***Vettore ordinato in senso crescente:***
    - **`i<j`** si ha **`V[i]<V[j]`**

2	3	5	6	7	8	10	11

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

ESERCIZIO: RICERCA BINARIA

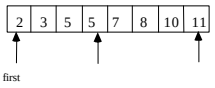
- Ricerca binaria di un elemento in un vettore ordinato in senso non decrescente in cui il primo elemento è **`first`** e l'ultimo **`last`**.
- La tecnica di ***ricerca binaria*** rispetto alla ricerca esaustiva, consente di eliminare ad ogni passo metà degli elementi del vettore.

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`**).

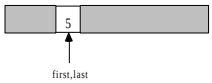
ESERCIZIO: RICERCA BINARIA

- Esempio: si cerca il valore **`el=4`**
- **`med = (first+last)/2`**
- **`el < V[med]`**



- **`el > V[med]`**

- Vettore a una componente: fine della ricerca con insuccesso



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 di un elemento

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

int ricerca (int vet[], int el);
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 medio 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
- Si scriva una procedura di ricerca binaria ricorsiva

ESERCIZIO: Ordinamento di un vettore

- Dati n valori interi forniti in ordine qualunque, stampare in uscita l'elenco dei valori dati in ordine crescente.
- Vettore di elementi di un certo tipo, sul quale è definita una ***relazione d'ordine totale*** (ad esempio, tipo float)
  - **Naive sort (o selection sort, o ordinamento per minimi successivi):**
  - Ad ogni passo seleziona il minimo nel vettore e lo pone nella prima posizione, richiamandosi ed escludendo dal vettore il primo elemento.

ESERCIZIO: Ordinamento di un vettore

- Primo livello di specifica:
  - leggi gli elementi
  - ordina il vettore
  - stampa il vettore ordinato

  } Realizzo 3 procedure
- Secondo livello di specifica per l'ordinamento

  while (<il vettore ha più di una componente>)
  { <individua il minimo nel vettore corrente in posizione **posmin**>
  <scambia se necessario il primo elemento del vettore corrente con quello in posizione **posmin** >
  <considera come vettore corrente quello precedente tolto il primo elemento>
  }

ESERCIZIO: Ordinamento di un vettore

```
#include <stdio.h>
#define N 5
void leggi(int a[]); /*input del vettore */
void scrivi(int a[]); /*stampa*/
void naive_sort (int vet[]); /* ordina /*

main ()
{int i, a[N];

 leggi(a);
 naive_sort(a);
 scrivi(a);
}
```

ESERCIZIO: Ordinamento di un vettore

```
void  naive_sort (int vet[])
{int  j, i, posmin, min;
 for  (j=0; j < N; j++)
   {posmin=j;
    min = vet[j];
    for (i=j+1;i<N; i++)
        if (vet[i]<min)
         {min=vet[i];posmin=i;}

    if (posmin != j)       /*scambio */
       { min=vet[posmin];
         vet[posmin]=vet[j];
         vet[j]= min;
       }
    }
}
```

ESERCIZIO: Ordinamento di un vettore

```
void leggi(int a[])
{int i;

 printf ("Scrivi %d interi\n", N);
 for (i = 0; i < N; i++)
      scanf ("%d", &a[i]);
 }

void scrivi(int a[])
{int i;

 printf ("Vettore ordinato:\n");
 for (i = 0; i < N; i++)
      printf ("%d\t", a[i]);
 }
```

OSSERVAZIONI

- Si eseguono tutti i confronti anche se il vettore è già ordinato (esistono altri metodi più efficienti, li vedrete in altri corsi).
- Esercizi
  - Strutturarlo su più file
  - Scrivere una versione ricorsiva dell'algoritmo naive sort.