

**ESERCIZI
DI PROGRAMMAZIONE
DA TEMI D'ESAME**

- vettori -

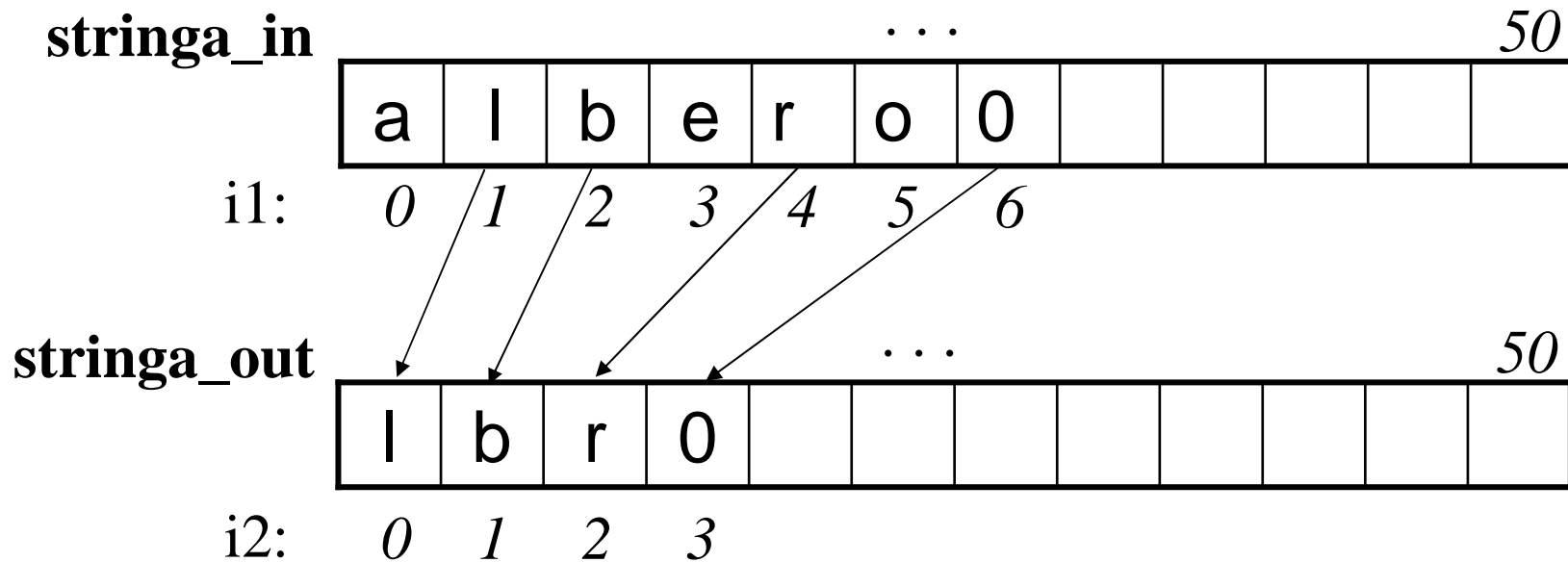
Esercizio 1

Sviluppare un programma che acquisisca dall'utente una stringa e la salvi in un vettore di caratteri. Successivamente, generi e stampi un altro vettore che contiene la stringa acquisita dalla quale siano eliminate tutte le vocali.

Esercizio 1

Sviluppare un programma che acquisisca dall'utente una stringa e la salvi in un vettore di caratteri. Successivamente, generi e stampi un altro vettore che contiene la stringa acquisita dalla quale siano eliminate tutte le vocali.

Algoritmo



➡ Ho bisogno di due indici: $i1$ e $i2$

ACQUISISCI stringain

```
i1=0;                                //contatore su stringa di ingresso
i2=0;                                //contatore su stringa di uscita
while(stringain[i1]!='\0'){
    Se è una consonante, inserisci il carattere stringain[i1] in stringaout
    i1++;
}
```

//quando esce devo ancora inserire 0 in stringaout

Inserisci 0 in stringaout

STAMPA stringaout

```
int i1, i2, i;
char stringain[50], stringaout[50];

printf("Inserisci stringa\n");
scanf("%s", stringain);

i1=0;                                //contatore su stringa di ingresso
i2=0;                                //contatore su stringa di uscita
while(stringain[i1]!='\0'){
    if(stringain[i1]!='a' && stringain[i1]!='e' && stringain[i1]!='i' && ...)
        stringaout[i2++]=stringain[i1];
    i1++;
}                                     //quando esce i2 contiene il posto in cui andrebbe inserito 0

stringaout[i2]='\0';
printf("Stringa di uscita:  %s\n", stringaout);
```

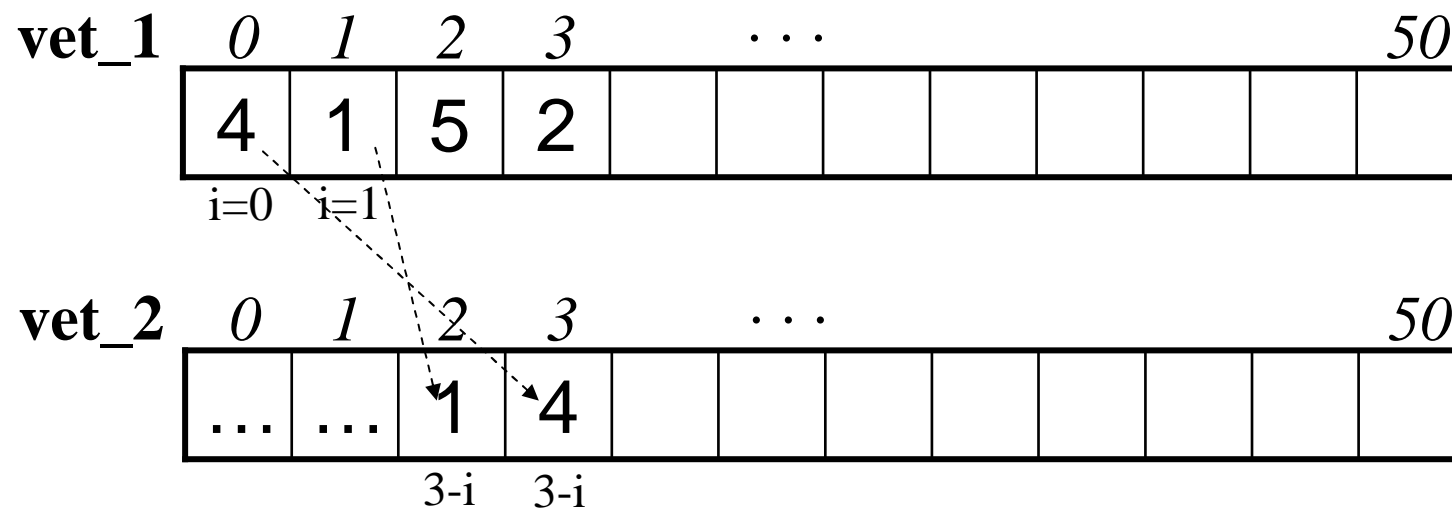
Esercizio 2

Sviluppare un programma che acquisisce dall'utente al massimo 50 numeri interi positivi (interrompendo l'acquisizione se viene inserito il numero 0), li inserisca in un vettore *vet_1* e produca un vettore *vet_2* che li contiene in ordine inverso.

Esercizio 2

Sviluppare un programma che acquisisce dall'utente al massimo 50 numeri interi positivi (interrompendo l'acquisizione se viene inserito il numero 0), li inserisca in un vettore *vet_1* e produca un vettore *vet_2* che li contiene in ordine inverso.

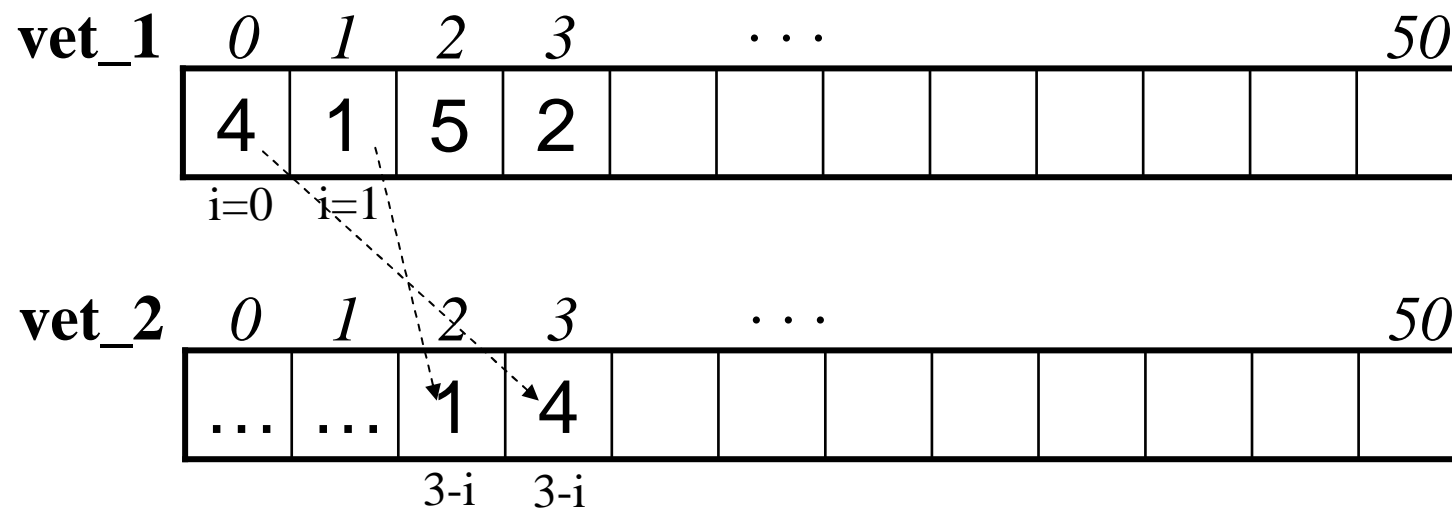
Algoritmo (per la parte di inversione)



Esercizio 2

Sviluppare un programma che acquisisce dall'utente al massimo 50 numeri interi positivi (interrompendo l'acquisizione se viene inserito il numero 0), li inserisca in un vettore *vet_1* e produca un vettore *vet_2* che li contiene in ordine inverso.

Algoritmo (per la parte di inversione)



Supponiamo n sia l'ultima posizione del vettore (3 in questo caso):

```
for(i=0; i<=n; i++)  
    vet_2[n-i]=vet_1[i];
```



```
printf("Inserisci al massimo 50 numeri positivi (0 per terminare)\n");
```

```
i=0;    // prossima posizione libera da occupare
```

```
do{
```

```
    scanf("%d", &num);
```

```
    if(num!=0){
```

```
        vet_1[i]=num;
```

```
        i++;
```

```
    }
```

```
} while(num!=0 && i<50);
```

```
n=i-1; //ora n indica l'ultima posizione occupata del vettore
```

```
for(i=0; i<=n; i++)
```

```
    vet_2[n-i]=vet_1[i];
```

```
printf("Vettore invertito\n");
```

```
for(i=0; i<=n; i++)
```

```
    printf("%d\n", vet_2[i]);
```

NB: per forzare l'input di numeri positivi:

```
printf("Inserisci al massimo 50 numeri positivi (0 per terminare)\n");  
i=0;    // prossima posizione libera da occupare  
do{  
    do  
        scanf("%d", &num);  
    while(num<0)  
        if(num!=0){  
            vet_1[i]=num;  
            i++;  
        }  
} while(num!=0 && i<50);  
n=i-1; //ora n indica l'ultima posizione occupata del vettore  
...
```

Esercizio 3

Sviluppare un programma che acquisisca da tastiera due array contenenti 10 numeri interi (`int num1[10]`, `int num2[10]`), assicurandosi (per ogni array) che l'utente non inserisca un numero già inserito (in questo caso, ripetere l'acquisizione di ogni elemento che sia già stato inserito).

Si trovino quindi tutti gli elementi comuni ad entrambi gli array e si stampi un messaggio indicante, per ogni elemento comune, l'indice occupato nel primo array e nel secondo.

Acquisizione di un vettore con esclusione di numeri già inseriti

```
for(i=0; i<10; i++){  
    do  
        scanf("%d", &num);  
    while(<numero ripetuto>);  
    v1[i]=num;  
}
```

} Acquisisci in num un numero
che non sia già stato inserito

Acquisizione di un vettore con esclusione di numeri già inseriti

```
for(i=0; i<10; i++){
```

```
do
```

```
scanf("%d", &num);
```

```
while(<numero ripetuto>);
```

```
v1[i]=num;
```

```
}
```

Acquisisci in num un numero
che non sia già stato inserito

Raffiniamo il
codice in modo da
calcolare nella variabile
ripetuto la condizione da
verificare

```
for(i=0; i<10; i++){
```

```
do{
```

```
scanf("%d", &num);
```

```
ripetuto=0;
```

```
for(j=0; j<i; j++){
```

```
if(v1[j]==num) ripetuto=1;
```

```
}while(ripetuto);
```

```
v1[i]=num;
```

```
}
```

Identificazione degli elementi in comune tra v1 e v2

Per ogni elemento $v1[i]$ con $i=0, \dots, 9$

- scorri tutti gli elementi $v2[j]$ con $j=0..9$

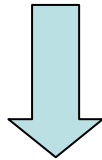
- se $v1[i] = v2[j]$ allora stampa messaggio

Identificazione degli elementi in comune tra v1 e v2

Per ogni elemento $v1[i]$ con $i=0, \dots, 9$

- scorri tutti gli elementi $v2[j]$ con $j=0..9$

- se $v1[i] = v2[j]$ allora stampa messaggio



```
for(i=0; i<10; i++)  
    for(j=0; j<10; j++)  
        if(v1[i]==v2[j])  
            printf("Elemento in comune %d, in posizione %d e %d\n", v1[i], i, j);
```

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

IL CODICE COMPLETO...

```
main(){
```

```
    int v1[10], v2[10];
```

```
    int i, j, comune;
```

```
    int num;
```

```
    printf("Inserisci il primo vettore (10 numeri interi)\n");
```

```
    for(i=0;i<10;i++){
```

```
        do{
```

```
            scanf("%d",&num);
```

```
            comune=0;
```

```
            for(j=0; j<i; j++)
```

```
                if(v1[j]==num)
```

```
                    comune=1;
```

```
        }while(comune);
```

```
        printf("Inserito numero %d\n", num);
```

```
        v1[i]=num;
```

```
    }
```

```
    printf("Inserisci il secondo vettore (10 numeri interi)\n");
```

```
    for(i=0;i<10;i++){
```

```
        do{
```

```
            scanf("%d",&num);
```

```
            comune=0;
```

```
            for(j=0; j<i; j++)
```

```
                if(v2[j]==num)
```

```
                    comune=1;
```

```
        }while(comune);
```

```
        printf("Inserito numero %d\n", num);
```

```
        v2[i]=num;
```

```
    }
```

```
    for(i=0;i<10;i++)
```

```
        for(j=0; j<10; j++)
```

```
            if(v1[i]==v2[j])
```

```
                printf("Elemento in comune %d, in posizione %d e %d\n", v1[i], i, j);
```

```
    system("PAUSE");
```

```
    return 0;
```

```
}
```


Esercizi proposti (vedere anche i temi d'esame dell'anno scorso!)

- 1) Scrivere un programma che riceva in ingresso due vettori *vet_1* e *vet_2*, ciascuno di 10 elementi interi e produca in uscita un vettore *vet_3* in cui ogni elemento sia il minimo degli elementi di ugual posizione in *vet_1* e *vet_2*.
- 2) Scrivere un programma che riceva in ingresso al massimo 10 interi e li stampi in ordine inverso (nota: simile all'esercizio 2 dei lucidi, ma in questo caso si può fare a meno di definire un vettore *vet_2*)
- 3) Scrivere un programma che acquisisca da tastiera al massimo 50 numeri interi, interrompendo l'acquisizione quando venga immesso il numero 0. Salvare i numeri acquisiti in un vettore e, successivamente, stampare il minimo, il massimo e la media.
- 4) Scrivere un programma che acquisisca da tastiera un vettore di 20 interi compresi tra 5 e 10, quindi stampi per ogni elemento quante volte compare nel vettore.
- 5) Sviluppare un programma che acquisisca dall'utente due vettori:
 - un insieme di caratteri (al più 24) *vet_1*;
 - una stringa di caratteri (di al massimo 50 caratteri)Successivamente, il programma deve generare e stampare un altro vettore che contiene la stringa acquisita dalla quale siano eliminati tutti i caratteri specificati in *vet_1*.