

# Lezione 2: esempi

# NUOVO PROBLEMA

vogliamo leggere interi da cin fino a  
quando non viene letto 0  
e vogliamo contare quanti positivi e  
quanti negativi vengono letti

attenzione: lo 0 non viene contato

## FORMALIZZIAMO

PRE= (cin contiene l'intero 0 preceduto da n interi diversi da 0, con  $n \geq 0$ )

POST= (  $neg + pos = n$  ) && ( neg è il numero degli interi negativi che precedono 0 su cin, mentre pos è il numero degli interi positivi che precedono 0 su cin)

il test del ciclo deve essere:

```
while (se non ho letto 0)
{
    metto a posto pos e neg
    leggo prossimo valore
}
//ho letto 0
```

ma la prima volta che arrivo al ciclo?

PRE= (cin contiene l'intero 0 preceduto da n interi diversi da 0, con  $n \geq 0$ )

```
main()
{
    int pos=0, neg=0, x;
    cin>>x;
    while (x !=0)
        {deve mantenere vero R}
```

R=(letti 1+neg+pos valori) &&  
(nei primi neg+pos c'erano neg negativi e pos positivi) &&  
(se  $x \neq 0 \Rightarrow \text{neg} + \text{pos} < n$ ) &&  
(se  $x = 0 \Rightarrow \text{neg} + \text{pos} = n$ )

```
while (x != 0)
{
    if(x < 0)
        neg=neg+1;
    else
        pos=pos+1;
    cin>>x;
}
```

R=(letti 1+neg+pos valori)  
&&  
(nei primi neg+pos c'erano  
neg negativi e pos positivi)  
&&  
(se  $x \neq 0 \Rightarrow \text{neg} + \text{pos} < n$ ) &&  
(se  $x = 0 \Rightarrow \text{neg} + \text{pos} = n$ )

# usiamo do-while, ma attenzione !!!

```
int pos=0, neg=0, x;  
do  
{  
    cin>>x;  
    if(x < 0)  
        neg=neg+1;  
    else  
        pos=pos+1;  
}  
while(x!=0);
```

```
int pos=0, neg=0, x;
```

```
do
```

```
{
```

```
    cin>>x;
```

```
    if(x < 0)
```

```
        neg=neg+1;
```

```
    else
```

```
        if(x>0)
```

```
            pos=pos+1;
```

```
}
```

```
while(x!=0);
```

# NON VA

R=(letti  $1+neg+pos$  valori) &&  
(nei primi  $neg+pos$  c'erano  
 $neg$  negativi e  $pos$  positivi)  
&&  
(se  $x \neq 0$  allora  $neg+pos < n$ )  
&&  
(se  $x=0$  allora  $neg+pos=n$ )



```
int pos=0, neg=0, x;  
do  
{  
    cin>>x;  
    if(x < 0)  
        neg=neg+1;  
    else  
        if(x>0)  
            pos=pos+1;  
}  
while(x!=0);
```

R= (se  $x \neq 0$  allora letti  
 $neg+pos \leq n$  valori con neg  
negativi e pos positivi)  
&&  
(se  $x=0$  allora letti n valori  
seguiti da 0, di cui neg sono i  
negativi e pos i positivi)

## NUOVO PROBLEMA

trovare il secondo tra  $n$  valori interi letti

**definizione di secondo:** se mettessimo gli  $n$  valori in ordine non decrescente, il secondo in quest'ordine è quello che cerchiamo

PRE=(cin contiene  $n \geq 0$ , seguito da  $n$  interi)

se  $n < 2$  il problema non ha senso. E' facile liberarsi da questo caso limite:

```
int n;  
cin >>n;  
if(n<2)  
    cout<<"problema mal posto";  
else  
{  
    qui ci sarà la parte interessante  
}
```

vogliamo entrare nel ciclo con primo e secondo giusti

```
int primo, secondo, temp;
```

```
cin >> primo >> secondo;
```

```
if(primo > secondo)
```

```
{
```

```
    temp=primo;
```

```
    primo=secondo;
```

```
    secondo=temp;
```

```
}
```

R=(primo=min dei valori letti, secondo= il secondo min. dei valori letti)

PRE=(cin contiene  $n \geq 0$ , seguito da  $n$  interi tutti distinti)

```
n=n-2;
while(n>0)
{
    cin >>temp;
    n--;
    if(temp<secondo)
    {
        if(temp<primo).....
        else .....
    }
}
```

R=(primo=min dei valori letti,  
secondo= il secondo min. dei  
valori letti)

```
while(n>0)
```

```
{
```

```
    cin >>temp;
```

```
    n--;
```

```
    if(temp<secondo)
```

```
    {
```

```
        if(temp<primo) //temp nuovo primo
```

```
            {secondo=primo; primo=temp;}
```

```
        else //temp è nuovo secondo
```

```
            {secondo=temp;}
```

```
    }
```

```
}
```

R=(letti vn-n valori) &&  
(primo=min dei valori letti,  
secondo= il secondo min. dei  
valori letti) &&( 0<=n<=vn)

e se gli n valori non fossero necessariamente tutti distinti? Funzionerebbe lo stesso?

```
int primo, secondo, temp;  
cin >> primo >> secondo;  
if(primo > secondo)  
{  
    temp=primo;  
    primo=secondo;  
    secondo=temp;  
}
```

primo e secondo  
potrebbero  
coincidere !!!!!

R=(primo=min dei valori letti, secondo= il  
secondo min. dei valori letti)

Dobbiamo chiarire il problema:

se gli  $n$  valori fossero: 2 3 2 2 1 1 1  
mettendoli in ordine non decrescente  
diventano: 1 1 1 2 2 2 3

allora il secondo è 1 o 2? insomma vogliamo  
che primo e secondo siano diversi o no?

- a) primo e secondo possono coincidere
- b) non devono coincidere



```
while(n>0)
```

```
{
```

```
    cin >>temp;
```

```
    n--;
```

```
    if(temp<secondo)
```

```
    {
```

```
        if(temp<primo) //temp nuovo primo
```

```
            {secondo=primo; primo=temp;}
```

```
        else //temp è nuovo secondo
```

```
            {secondo=temp;}
```

```
    }
```

```
}
```

caso (a) primo e secondo  
possono coincidere  
il nostro programma va  
ancora?

se (temp>=secondo) niente da fare

siamo nel caso (temp<secondo)

se (temp<primo) //temp nuovo primo

{secondo=primo; primo=temp;}

else // primo <= temp < secondo

{secondo=temp;}

}

}

**insomma funziona !**

caso (b) primo e secondo devono essere diversi

```
int primo, secondo, temp; bool trov=false;
cin >> primo; temp=primo;
n=n-1;
while(n && !trov)
{
    cin>>temp;
    n--;
    if(primo!=temp)
        trov=true;
}
```

R=(letti  $x=vn-n$  valori)  
&&(primo è primo letto e  
temp  $x$ -esimo) &&(i primi  
 $x-1 = primo$ ) && (trov  $\Leftrightarrow$   
temp $\neq$ primo)  
&&(0 $\leq n \leq vn-2$ )

## altra soluzione con do-while

```
int primo, secondo, temp; bool trov=false;
cin >> primo;
n=n-1;
do{
    cin>>temp; n--;
    if(primo!=temp)
        trov=true;
}
while(n && !trov);
```

R=(letti  $x = v_{n-n}$  valori)  
&&(primo è primo letto e  
temp x-esimo) &&(i primi  
 $x-1 = primo$ ) && (trov  $\Leftrightarrow$   
temp!=primo)  
&&(0 $\leq n \leq v_{n-2}$ )

```
if(trov)
{
    if(primo>temp)
        {secondo=primo;primo=temp;}
    else
        secondo=temp;
```

```
while(n>0)
{
    che mantenga R
}
```

```
}
```

R=(letti vn-n valori)  
&&(considerando i valori  
distinti, primo è il minimo e  
secondo il secondo)

```
while(n>0)
```

```
{
```

```
    cin>>temp;
```

```
    n--;
```

```
    if(temp<secondo)
```

```
        if(temp<primo)
```

```
            {secondo=primo; primo=temp;}
```

```
        else
```

```
            if(temp>primo)
```

```
                secondo=temp;
```

```
}
```

R=(letti vn-n valori)

&&(considerando i valori  
distinti, primo è il minimo e  
secondo il secondo)