Programmazione

2 crediti

restanti 8 crediti al II semestre

il testo è:

Programmazione consapevole

di G.Filè,

nelle librerie Progetto

informazioni sul C++ sulla rete:

- www.cplusplus.com

sito del corso è:

http://elearning.math.unipd.it/moodle/

→ci troverete TUTTO

→FORUM in cui fare e rispondere a domande

sequenza delle lezioni
lunedì e martedì 11:30-13:30
venerdì 9:30-11:30

il compitino fatto il17/11/201 4
vale 0,1,2 o 3 punti nell'esame scritto finale

Cosa abbiamo fatto

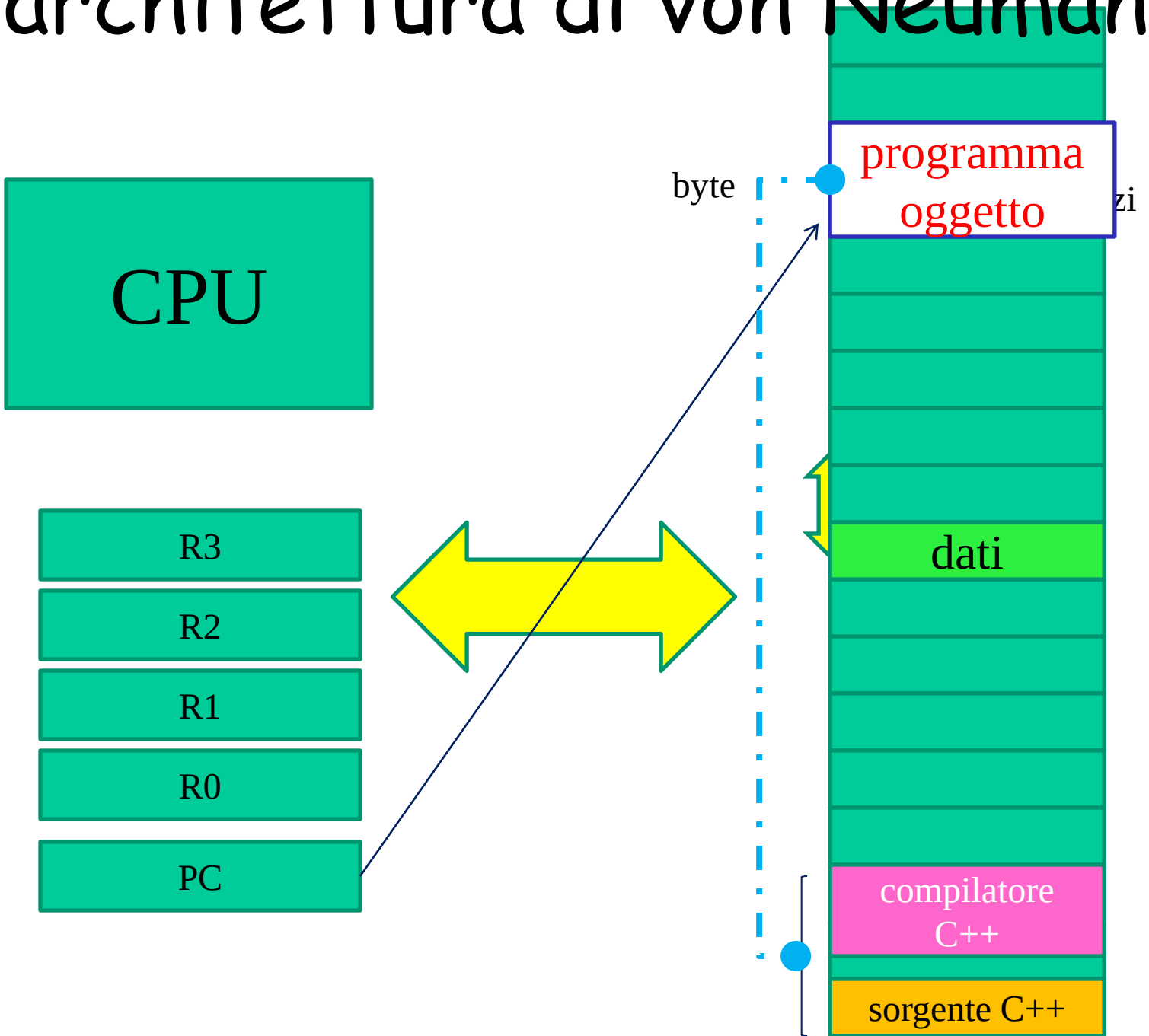
- architettura di von Neumann

- Linguaggio C++ minimale (cap. 2, 3 e 5.2 del testo)

- qualche programma con dimostrazioni di correttezza (cap. 4 del testo)

- uso del laboratorio e del software per gli esercizi e gli esami

architettura di von Neumann



come invocare il compilatore

g++ prova.cpp lo compila in ./a.exe (.out)

./a.exe (.out) lo esegue

Il C(++) in una nocciolina

- dichiarazioni
- input/output
- assegnazione
- condizionale
- while

Dichiarazioni (cap 2)

```
int  x=0, y=1, z=2, w;
     w=x*y+z;
```

L-valore e R-valori

TIPI

int e bool

il programma più semplice che c'è

```
#include<iostream>   // preprocessing
using namespace std;

main()
{
int x = 0, y=3;
cout << x << y;
cin  >> x >> y;
}
```

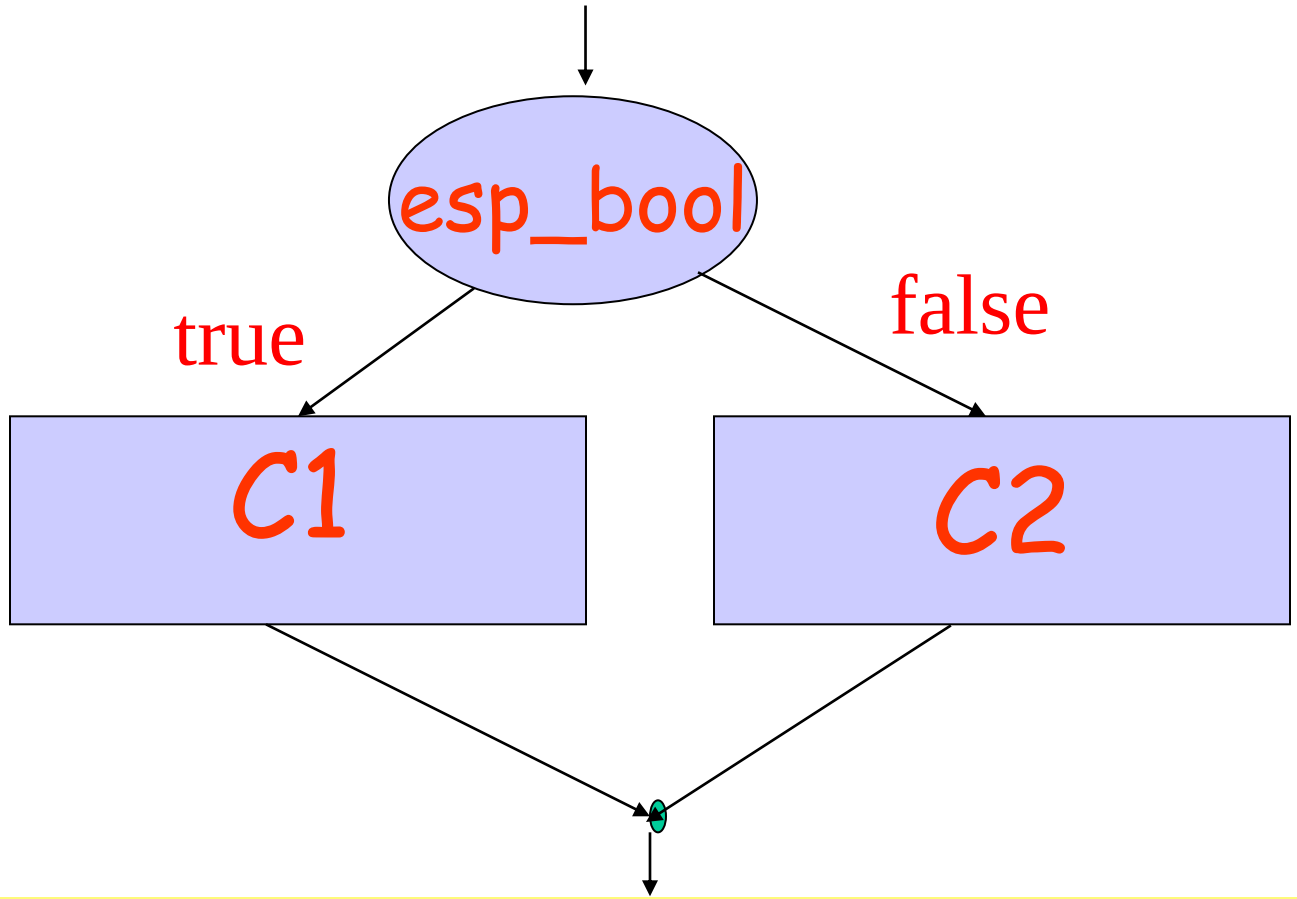
condizionale o if-then-else

```
if (esp_bool)
 {
    C1
}
else
{
   C2
}
```

blocco then

blocco else

diagramma di flusso



ramo then

ramo else

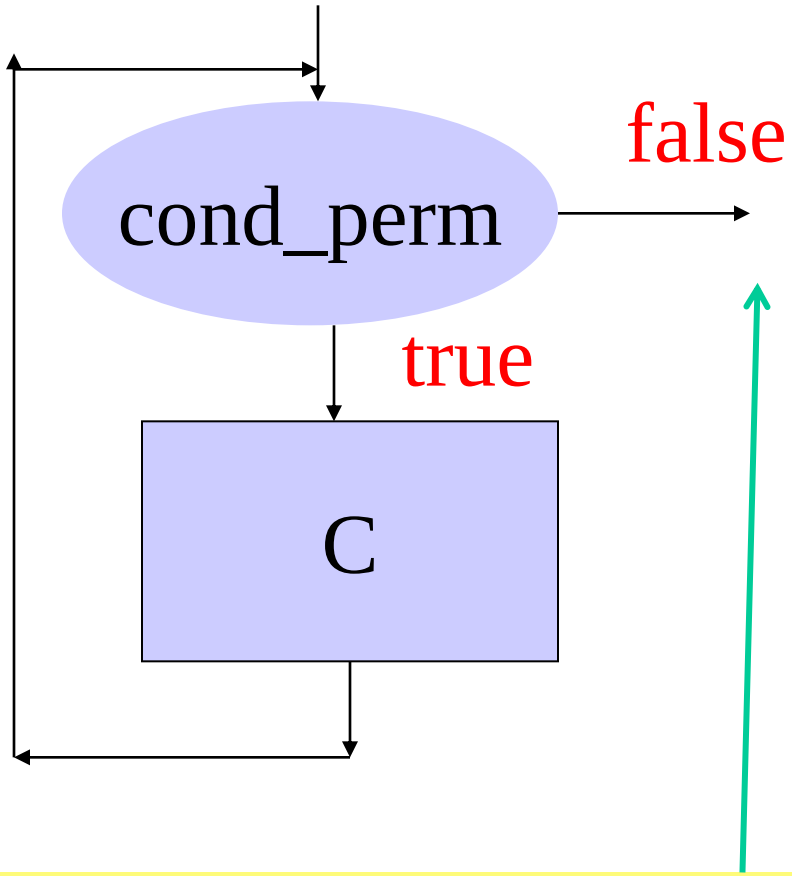
1 punto d'entrata ed 1 d'uscita

comando iterativo o while

```
while(cond_perm)
{
   C
}
```

corpo del while

while



1 punto d’entrata ed 1 d’uscita

esempio di while:

```
int x=0;
while(x < 10)
  {
     x=x+1;
  }
//POST= (x=10)
```

INVARIANTE

```
int x=0;

while(x < 10)
  {
     x=x+1;
  }
//POST=(x=10)
```

R={0<= x <=10}

{0<= x <10}

uscita R && { x >=10}

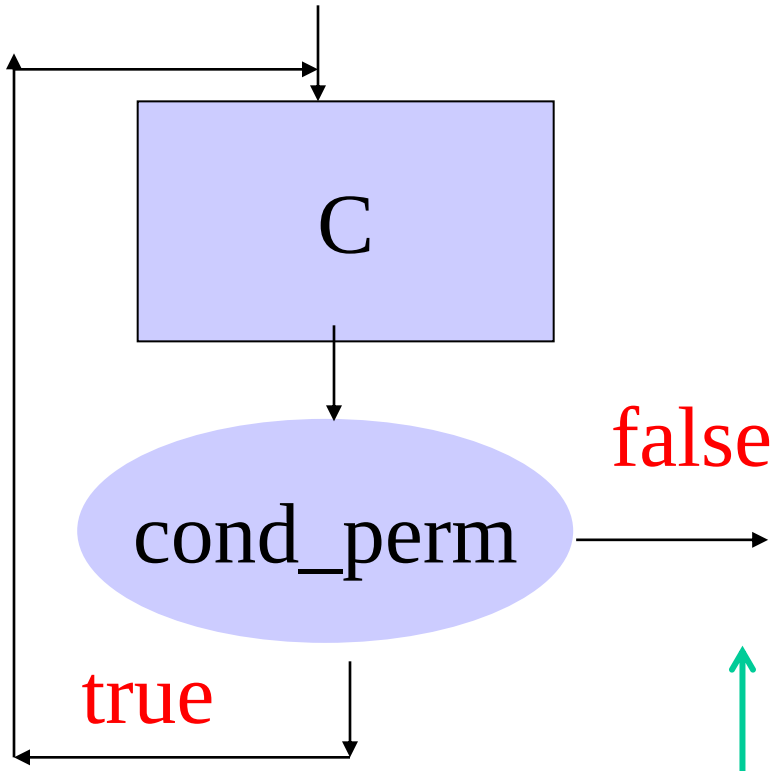
valgono 3 fatti:

1) R è vera la prima volta che l'esecuzione arriva al ciclo (condiz. iniziale)

2) R vale ogni volta che l'esecuzione ritorna all'inizio del ciclo (invarianza)

3) R && !(cond_perm) => POST (condiz. d'uscita)

do-while



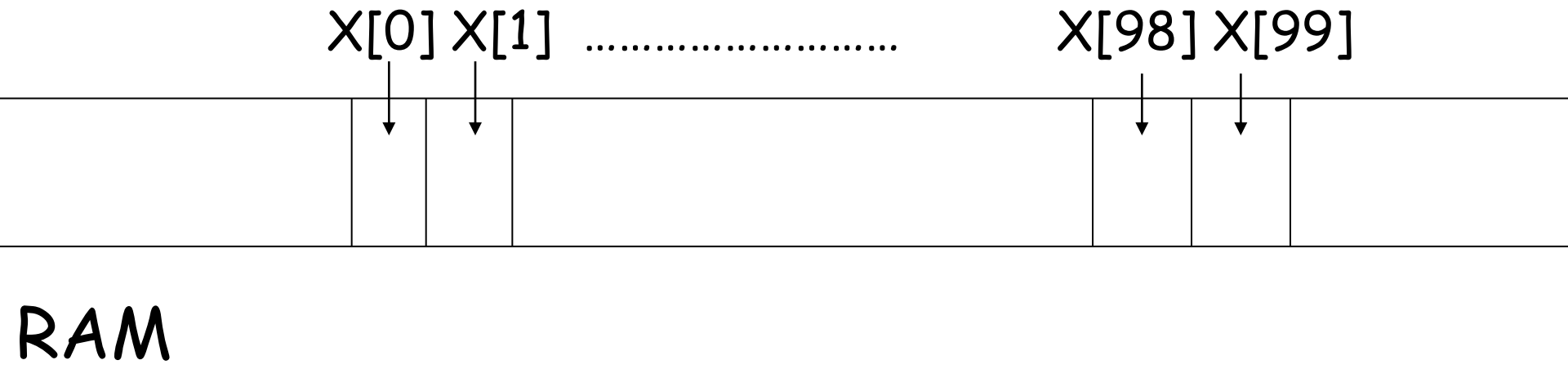
1 punto d’entrata ed 1 d’uscita

```
int x=0;
do        {0<= x <10}
    x=x+1;
while(x < 10);   R={1<= x <=10}
                 uscita  R && { x >=10}
//POST=(x=10)
```

array

5.2 del testo

```
int X[100];
```



array a 2, 3, 4, 5, …. dimensioni:

```
int X[5][10];
int Y[3][4][10];
int Z[10][10][20][30];
```

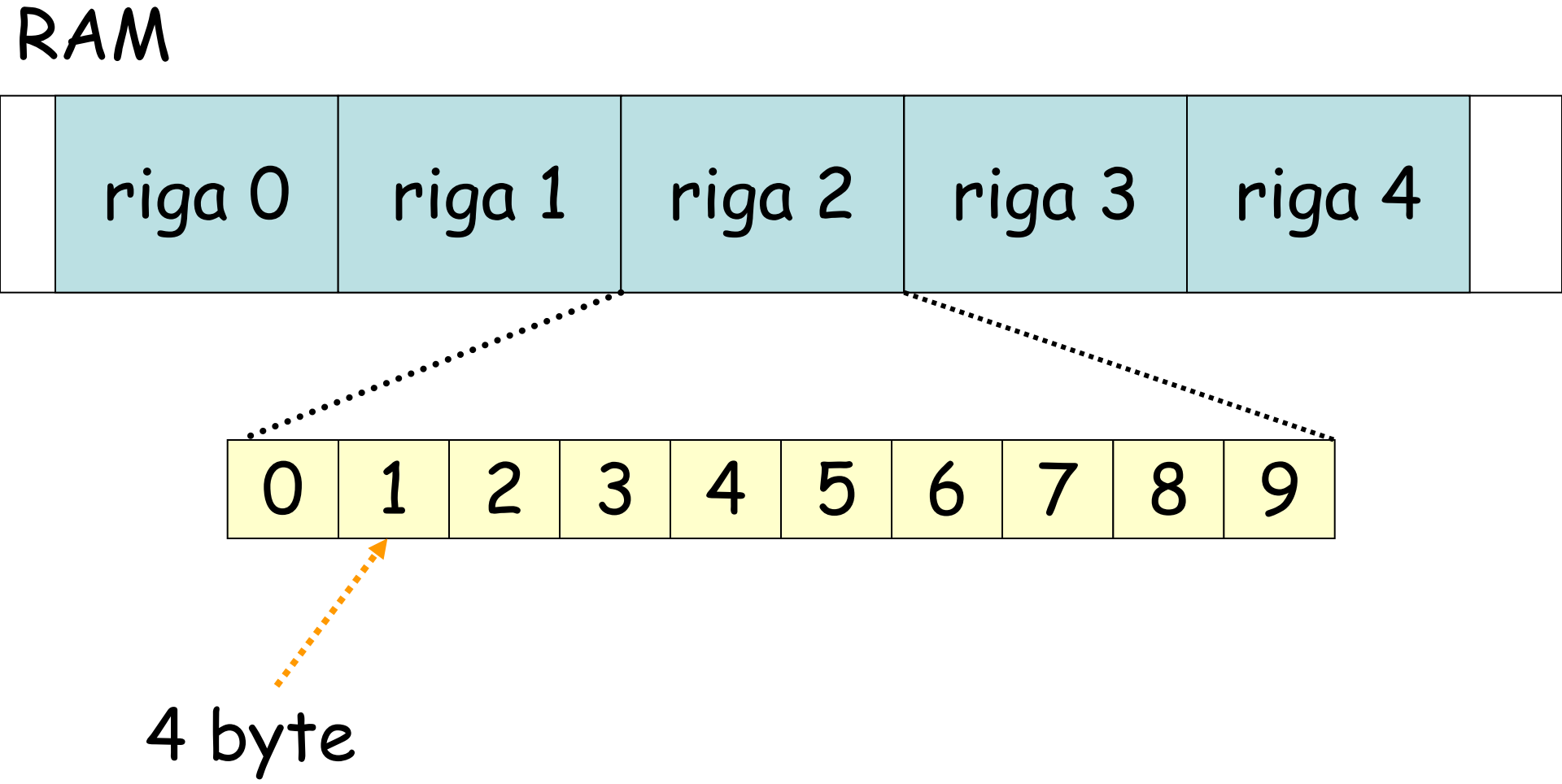
e così via

limite della prima dimensione è 5, della seconda è 10

elementi: X[0][0] X[4][9]
Z[0][0][0][1]

Y[3][0][1] non esiste

di nuovo gli elementi sono accostati nella RAM per righe: int X[5][10]



e int Y[3][4][10]; ?

ogni strato è un array int [4][10] immagazzinato in memoria come visto prima

inizializzare un array a 3 dimensioni da cin

```
main()
{
 int A[3][10][10], i=0;
 while(i<3)
 { int j=0;
   while(j<10)
    { int z=0;
      while(z<10)
        {cin >> A[i][j][z]; z++}
       j++;
     }
   i++;
  }
}
```

iterazione for

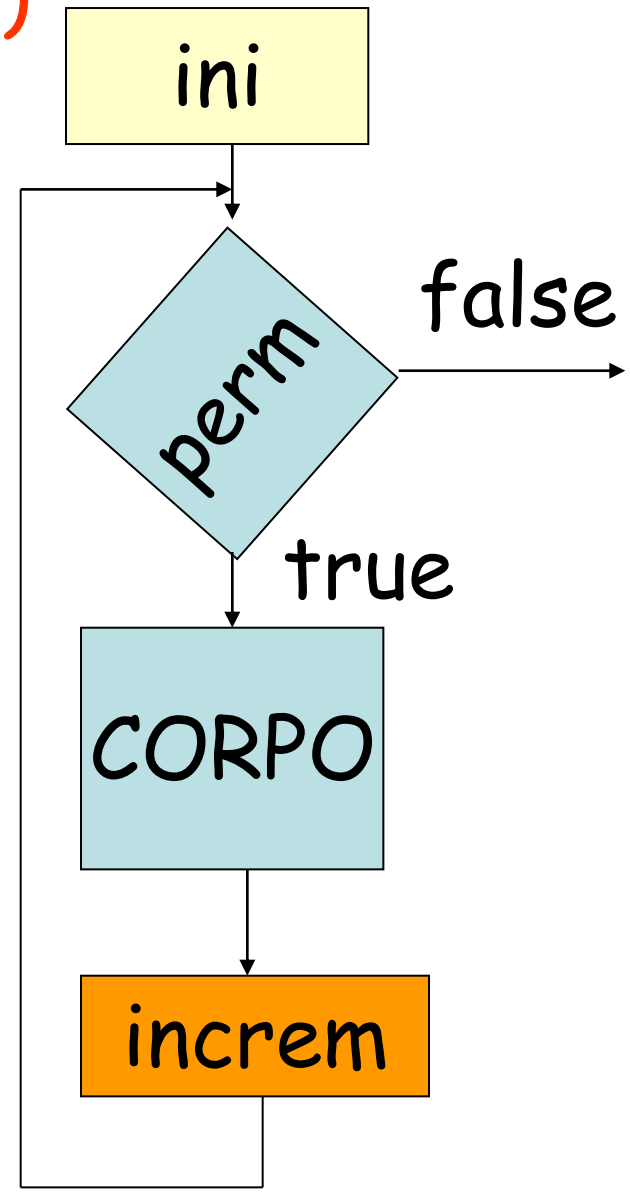
```
int A[3][10][10];
for(int i=0; i<3; i=i+1)
   for(int j=0; j<10; j=j+1)
      for(int z=0; z<10; z=z+1)
         cin>>A[i][j][z];
```

for(ini ; condiz-perm ; increm)
  { CORPO }

{ ini;condiz-perm;
{CORPO}
increm}

for (ini ; condiz-perm ; increm)

{ CORPO}



visibilità delle variabili
pag. 63 del testo:

```
int i=4; cout<<i<<endl;
for(int i=5; i<10; i++)
{
  cout<<i<<endl;
  int i=0;
  cout<<i<<endl;
  i++;
}
cout<<i<<endl;
```

for e while

```
for ( ini ; cond-perm ; increm)
{ CORPO}
```

è equivalente a

```
{ini;
while (perm)
{{CORPO}
increm;}
}
```

in certi casi è più comodo il for in altri il while

Regola di prova del for è la stessa del while

PRE <for(ini; perm; increm) CORPO> POST

è come **R**

PRE <ini; while(perm)CORPO;incr> POST

anche qui prova in 3 parti

esercizio: trovare le posizioni del minimo e del massimo valore in un array ad una dimensione di 100 elementi

PRE =(A[100] definito, sia $\underline{A}$ il valore)

POST=(posmin e posmax $\in$ [0,99],
A[posmin] <=A[0..99] e
A[posmax]>=A[0..99]), A=$\underline{A}$

```
int posmin=0, posmax=0;
for(int i=1; i<100; i++)  //R
  {
    if(A[posmin]>A[i])
       posmin=i;
     else
    if(A[posmax]<A[i])
       posmax=i;
  }
```

R=(0<=i<=100, 0<=posmin <i, 0<=posmax< i,) &&
(A[posmin]<=A[0..i-1]) &&
(A[posmax]>=A[0..i-1]) &&(A[0..i-1]=A[0..i-1])