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

-3, 6 e 7 ottobre teoria in aula

-13 lab (uso PC)

-14 teoria in aula

-21 lab (prog con correttezza)

-28 teoria in aula

-4 novembre lab (prog con correttezza)

-compitino nella settimana del 17 novembre

-vale per al massimo 3 punti nell'esame scritto finale

Cosa faremo

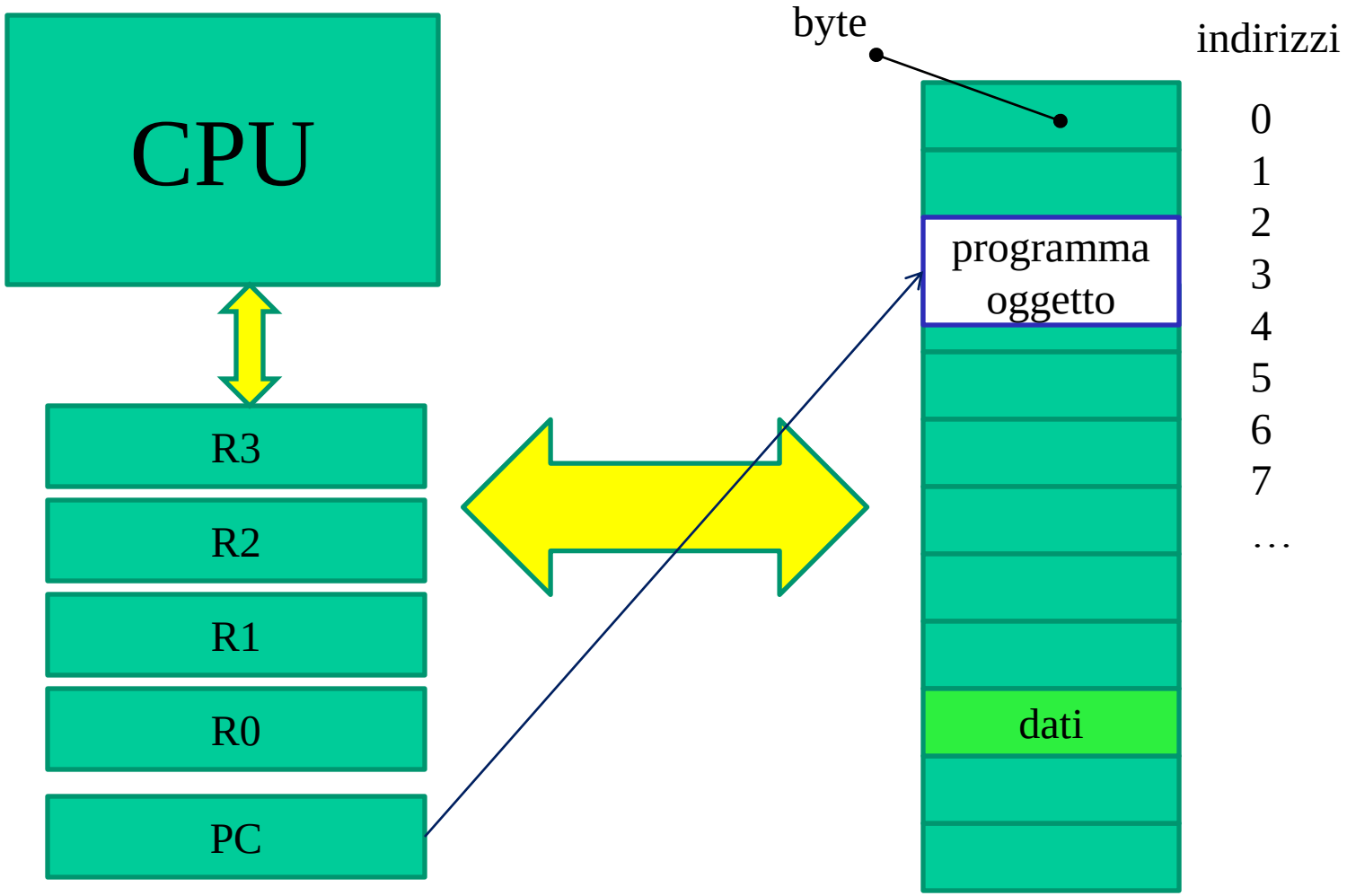
- architettura di von Neumann

- 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



il compilatore C++
(che è un programma)
traduce il programma
C++ (sorgente)
nel programma
eseguibile (oggetto)

come compilare ed eseguire un programma:

scrivere un programma su un file, per esempio prova.cpp

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)

Nei programmi usiamo variabili.

Solamente di tipo intero e booleano

una dichiarazione è:

int x=0, y=1, z=2, w;

tipo intero x, y, z sono dichiarate con inizializzazione, mentre w non è inizializzata quindi è indefinita: vietato usarne il valore

int x=1;

x ha valore 1

e questo 1 è in memoria a qualche indirizzo

1 è l'R-valore di x e l'indirizzo è l'L-valore di x

i valori interi rappresentabili nel computer sono un intervallo:

lim_inf.....0.....lim_sup

INT_MIN INT_MAX

sono costanti che il C++ mette a disposizione

Variabili booleane

bool x, ultimo= false, primo=true;

solo 2 valori di tipo bool: true e false

input/output (cap 3.1)

standard input = tastiera cin
standard output = video cout

```
int x=0, y=3;

cout<< x << y; // stampiamo 0 3
cin >> x >> y;  //leggiamo nuovi
                //valori
```

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;
}
```

assegnazione (cap 3.2)

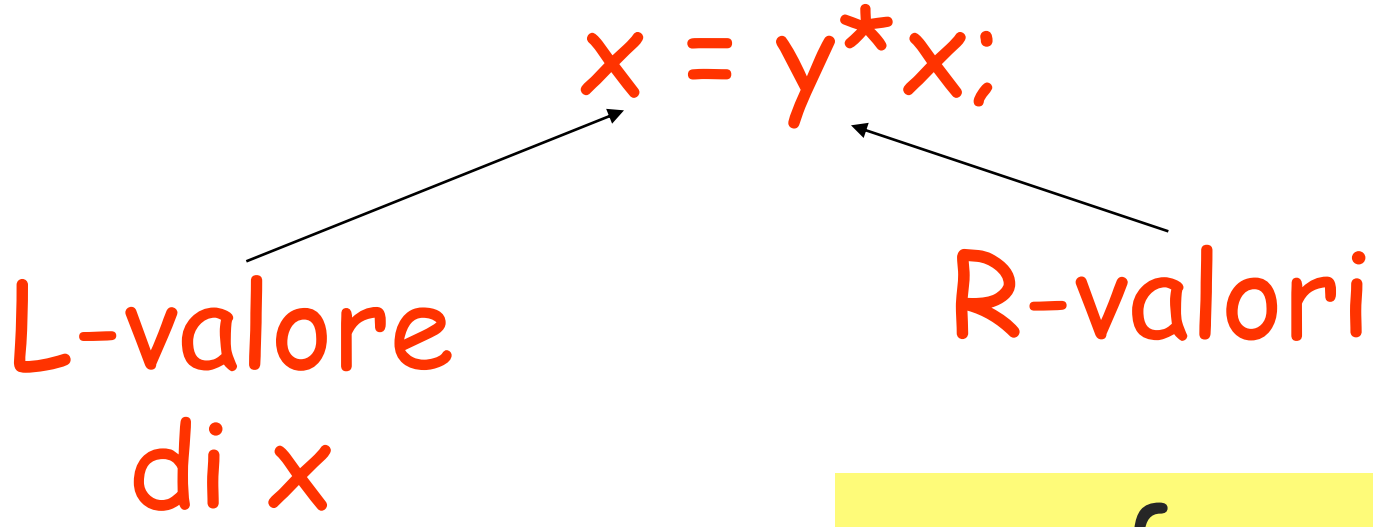
```
int x=0, y=2;

x = y * 12 ;
```

2*12 viene assegnato a x

```
y=x-1;

x=y*x;
```



perché è lì che si deve mettere il valore di y*x

per fare la somma

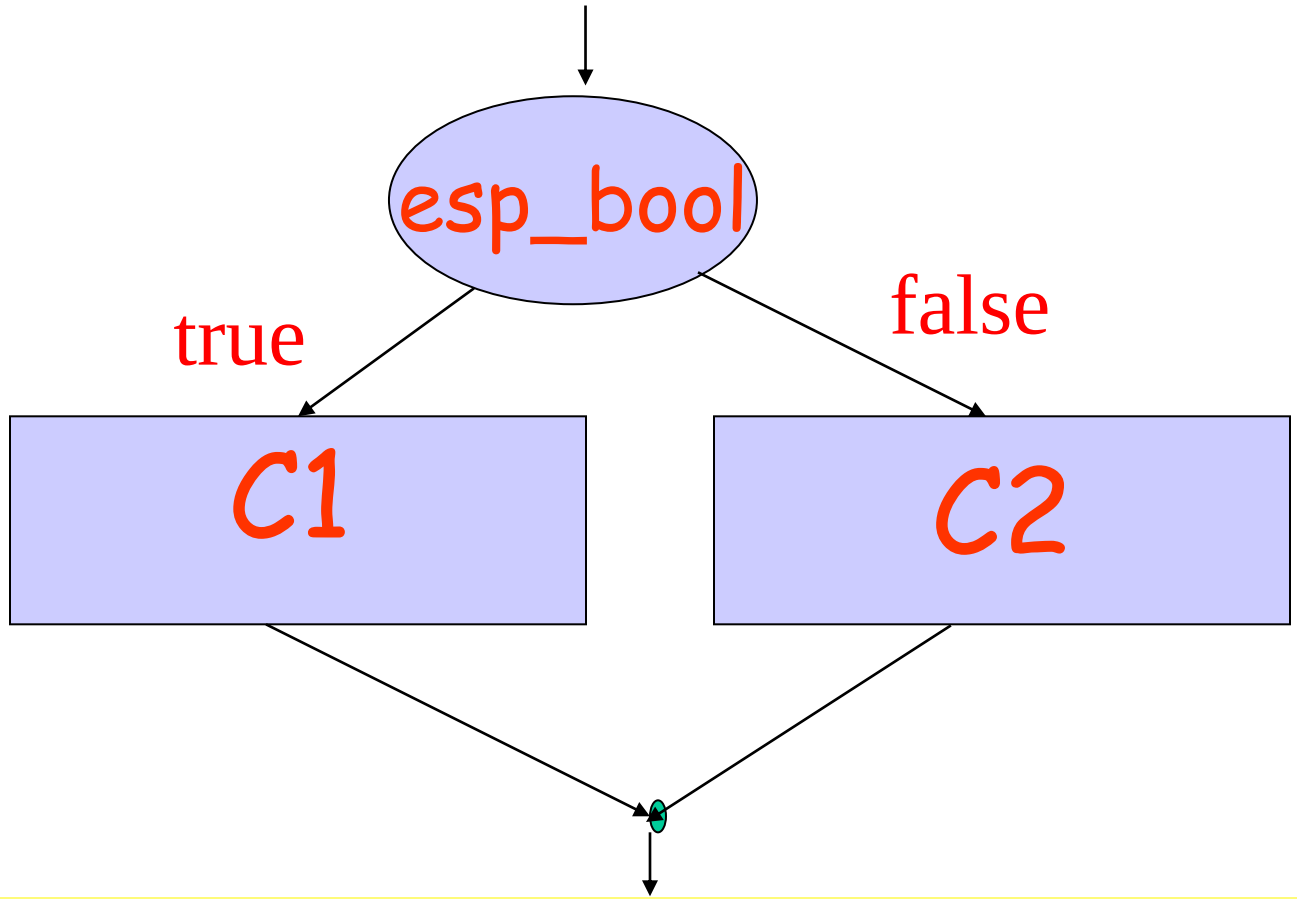
condizionale o if-then-else

```
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

```
//PRE=(cin contiene v1 e v2)

main()
{int x, y;
cout<<"inserire 2 interi";
cin >> x >> y;
cout<<" valore di x="<<x<<"valore
di y="<<y;
int SOM=x+y;

//POST=(x ha R-valore v1, y ha R-valore v2
e SOM ha R-valore v1+v2)
```

//POST=(x ha R-valore v1, y ha R-valore v2 e SOM ha R-valore v1+v2)

```
if(SOM>0)
  y=SOM;
else
 x=SOM;
}
```

//POST1=(SOM>0 => y=SOM && x=v1) && (SOM <=0) => x=SOM && y=v2)

Non è sempre vero che la post dopo un condizionale sia un condizionale

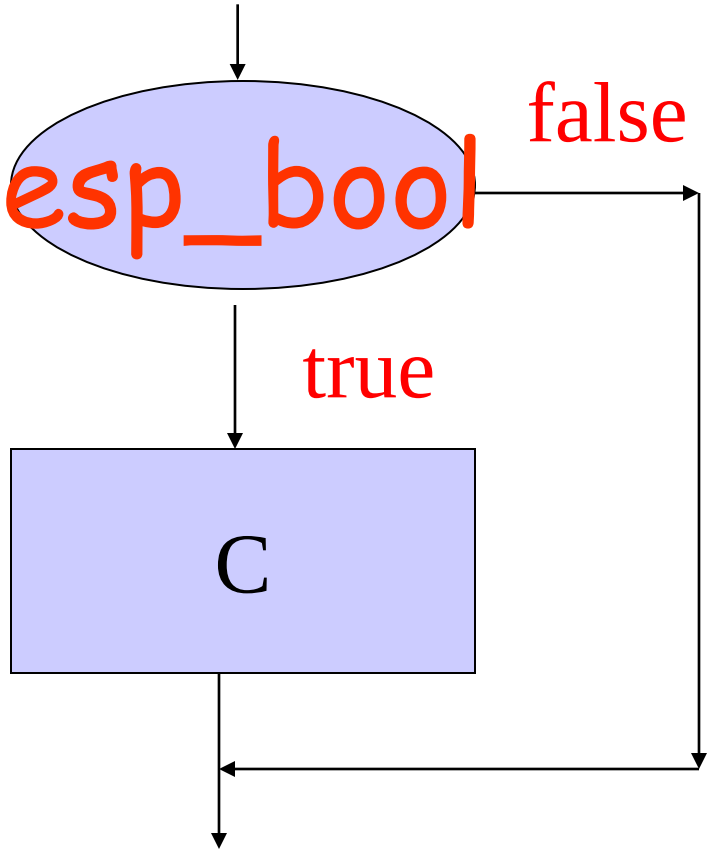
```
 if(x < y)
  SOM=x;
else
  SOM=y;
```

//POST=(SOM è il valore minore tra x e y)

ma anche questa sarebbe possibile:
(x < y => SOM=x) && (x>=y => SOM=y)

if(esp_bool) {C}

solo il ramo then

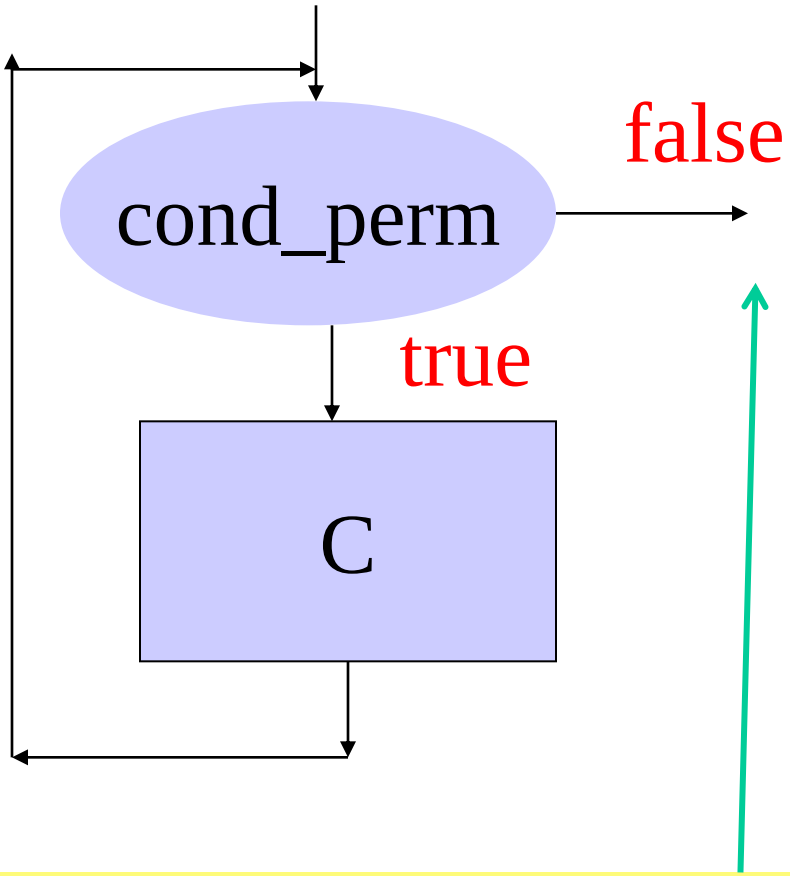


comando iterativo o while

```
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)
```

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

R={0<= x <=10}

{0<= 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)

che succede in questo caso?

```
int x=3;
while(x < 10)
  {
     x=x+1;
  }
//POST= (x=10)
```