A.A. 2003-2004

Esercitazioni Programmazione 1

Settimana 2

bottiglie.c

Cicli for

```
main(){

for (int num=100; num>=0 ; num-- )
            cout << num
                 << " bottiglie di vino alla festa. \n ";

cout << endl ;
}
```

1-2

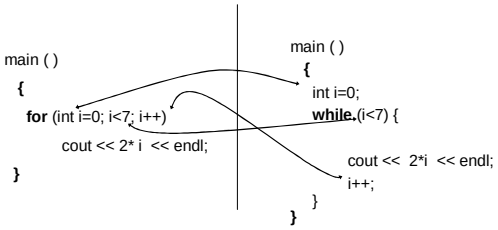
Stampare i primi 7 numeri pari.

```
main ( )
 {
   for (int i=0; i<7; i++)
      cout << 2*i<< endl;
   // Alla fine dell'i-esima iterazione ho
   // stampato i primi i numeri pari
   }
```

1-3

for e while

Stampare i primi 7 numeri pari.



Ad ogni passo: ha stampato i+1 numeri pari.
Come termina?

1-4

Array

Il giorno piu' freddo

Scrivere un programma che registra le temperature della settimana, per poi eseguire alcune operazioni sui dati.
Per es:
determinare il giorno più freddo,
stampare le temperature registrate,
stampare la differenza di temperatura rispetto allo stesso giorno della settimana precedente….

int A[7];

equivale a dichiarare 7 variabili di tipo int:

A[0], A[1], A[2], A[3], A[4], A[5], A[6]

1-6

Warm-up: scorrere un array

```
int A[7];

                    Inizializzo tutti gli elementi a 0
for (int i=0; i<7; i=i+1)
      A[i] =0;

                    Inizializzo tutti gli elementi da cin
for (int i=0; i<7; i=i+1)
      cin >> A[i];

                    Stampo tutti gli elementi

   for (int i=0; i<7; i=i+1)
        cout << A[i] ;
```

1-7

Il giorno piu' freddo:

Scrivere un programma che memorizza in un array di elementi di tipo int le temperature della settimana.

Il programma determina il giorno più freddo, e stampare a video la temperature di ogni giorno.

Inizializza array ***ok***

Calcola giorno piu' freddo → Calcolo minimo, Calcolo suo indice

Stampa giorno piu' freddo → indice del min + 1

Stampa array ***ok***

1-8

```
{ int temperature[7];

/* inizializzazione  array */
  cout << "inserisci le temperature di questa settimana"
       << endl;
   for (int i=0; i<7; i++)
        cin >> temperature[i] ;
```

1-9

```
/* calcolo del giorno più freddo
(= calcolo elemento minimo e suo indice)*/

int min = temperature[0];
int indice_min=0;

for (int i=1; i<7; i++)
            if (temperature[i] < min) {
                 min = temperature[i];
                 indice_min=i;
                } //attenzione alle parentesi
```

1-10

```
/*stampa giorno piu' freddo */
cout << "Temperatura minima: " << min;
cout << " il " << indice_min +1 << ". giorno"<< endl;


/* stampa array */
//come stampa?  stessa riga o in colonna?
for(int i=0;i<7;i++)
cout   << temperature[i] << " ";
cout << endl;
```

1-11

biblioteca, discussione soluzioni

Cattiva domanda: e' giusto?

Buona domanda: posso fare meglio?

Leggi costo prossimo libro e sottrai al saldo
finche' restano soldi, oppure finche' l'utente inserisce il valore sentinella
(che non va considerato come costo di libro)

ATTENZIONE :

Faccio anche sulla sentinella le operazioni che si fanno sui dati? *almeno pericoloso*

Faccio una volta di troppo o di meno lettura, somma ? *errore*

Buona scelta variabili, test significativi.

1-13

- Prendiamo sentinella = -1,
  per rendere piu' chiaro il problema legato alla sentinella
  (la biblioteca vuole poter considerare anche libri offerti gratuitamente, per cui 0 diventa un valore possibile).
- Ci concentriamo sull'essenziale.
  Semplifichiamo la lettura: il *budget* e' fissato a 4 (non e' realistico, ma semplifica i test)

1-14

problema: legge una volta di troppo

```
int saldo = 4;  // inizializzo saldo = budget.
int libro;

cin >> libro;

while ( saldo > 0  &&  libro !=-1)
 {saldo = saldo-libro; // sottraggo libro al saldo solo se
                       //sicuro che libro non e' sentinella
  cin >> libro;
                    //NB: nuovo saldo potrebbe essere <0
                    // leggo comunque
  }

cout << "saldo finale: " << saldo<< endl;
```

1-15

problema: sottrae la sentinella

```
int saldo = 4;  //inizializzo saldo = budget
int libro = 1;  //valore fittizio

while ( saldo > 0  &&  libro !=-1 )
  {cin >> libro;
                    //sottraggo libro a saldo PRIMA
                   //di valutare se libro e' sentinella
   saldo = saldo-libro;
   }

cout << "saldo finale: " << saldo<< endl;
```

1-16

variante con **brutto errore**
(difficilmente visibile ai test)

```
int saldo = 4;  //inizializzo saldo = budget
int libro;  // errore capitale
            //risposta secondo fortuna

while ( saldo > 0  &&  libro !=-1 )
  {cin >> libro;
   saldo = saldo-libro;
   }

cout << "saldo finale: " << saldo<< endl;
```

1-17

problema: sottrae la sentinella

versione do while

```
int saldo = 4;  //inizializzo saldo = budget
int libro;

do
{cin >> libro;

 saldo = saldo-libro;
                    //sottrae senza sapere se e' sentinella
}
while ( (saldo > 0)  &&  (libro !=-1) );

cout << "saldo finale: " << saldo<< endl;
```

1-18

problema: legge una volta di troppo

versione do while

```
int saldo = 4;  //inizializzo saldo = budget
int libro =0;    // valore fittizio: sporco

do
{saldo = saldo-libro;  //saldo ora puo' essere negativo

 cin >> libro;            //legge anche se saldo <0
}
while ( (saldo > 0)  &&  (libro !=-1) );

cout << "saldo finale: " << saldo<< endl;
```

1-19

TOP: uso sentinella booleana

```
int saldo = 4;  //inizializzo saldo =budget
int libro;
bool continua=true;  //vero se non ho letto
                     //il valore sentinella

while ( saldo > 0  &&  continua==true )
  {cin >> libro;

   if (libro ==-1)
       continua=false; //ho letto il valore sentinella!
                       // esco dal ciclo
    else
       saldo = (saldo -libro);
  }
```

1-20 cout << "saldo finale: " << saldo<< endl;

Padroneggiare i cicli

CONDIZIONI di CONTROLLO

saper usare i booleani

Un test per questo ciclo

```
int x;
cout << "valore? " << endl;
cin >> x;

while (  ...  )
{cout << "valore?" << endl;
 cin >> x;
}
```

1-22

Scrivere il test che corrisponde a ciascuna delle seguenti condizioni:

- Continua fino a leggere i valori 18 o 30.
- Continua a leggere fino ad incontrare un numero che sia pari e positivo
- Fermati se leggi un intero non compreso tra 0 e 9
- Fermati se leggi un intero compreso tra 0 e 9

1-23

Chiedersi sempre:

il finche' nella specifica, fornisce una condizione di terminazione oppure per continuare ?

SPECIFICA: ripeti al più 10 volte *oppure* finchè l'utente lo desidera

cosa deve essere vero perché si continui?

ho ripetuto meno di 10 volte e l'utente vuole continuare

come termina il ciclo ?

ho ripetuto 10 o più volte oppure l'utente ha voluto smettere prima

while(volte**<**10 **&&** continua==true)...

SPECIFICA: ripeti finchè l'utente 1 *e* l'utente 2 lo desiderano

cosa deve essere vero perché si continui?

l'utente 1 vuole continuare e l'utente 2 vuole continuare

come termina il ciclo ?

l'utente 1 ha voluto smettere oppure l'utente 2 ha voluto smettere

while(continua1==true **&&** continua2==true) …

1-24

Test e terminazione

ok =0;

COSA FANNO QUESTI FRAMMENTI DI PROG?

test

```
while (ok==1) {
        cout << “continui? si[1] no[0] \n” ;
        cin >> ok;
        }
```

assegnazione

```
while (ok=1) {
        cout << “continui? si[1] no[0] \n” ;
        cin >> ok;
        }
```

1-26

Qual è il comportamento di questi 3 programmi?

```
int N=10;
int x=0;
while (x != N) {
    cout << x;
    x=x+2;
    }
```

stampa i pari fino a 10 in modo pericoloso !! Se cambio N=11?!

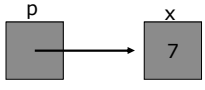
```
for ( int i=0; i <10; i= i *2 )
    cout << i;
```

```
for ( int i=1; i <10; i=i*2 )
    cout << i;
```

1-27

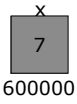
Puntatori

Puntatori



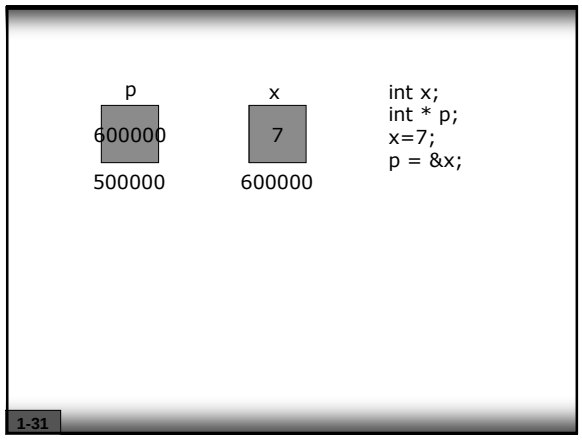

```
int x =7;
int *p ;
p = &x;

cout << *p  ??
```

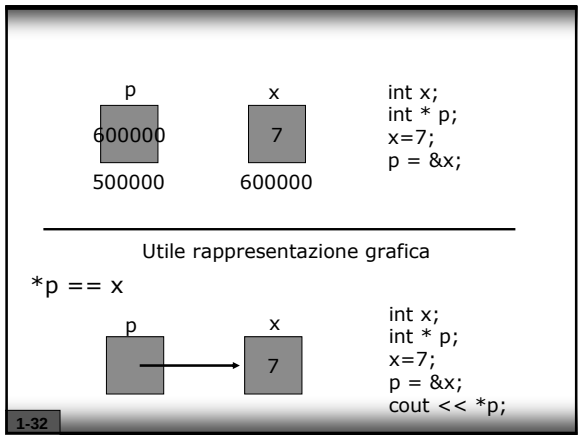
1-29

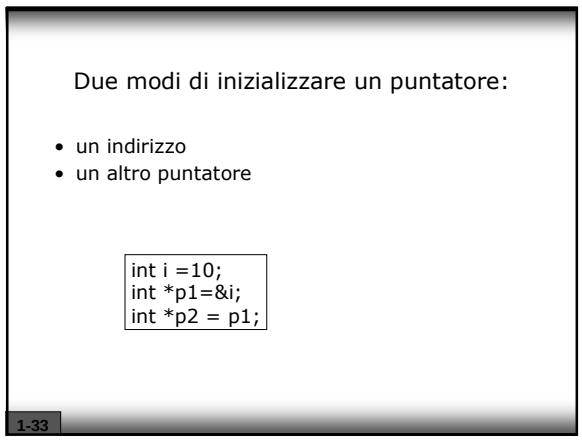


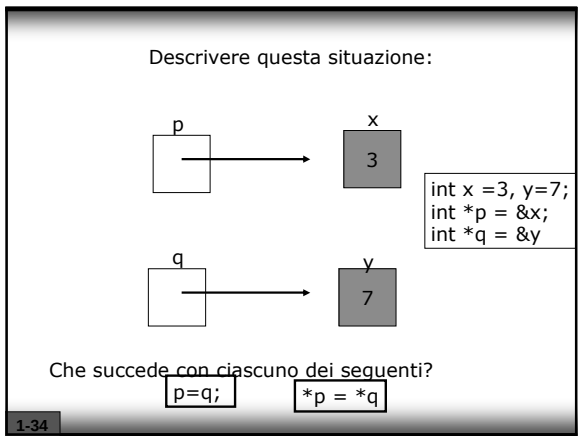

```
int x;
x = 7;
```

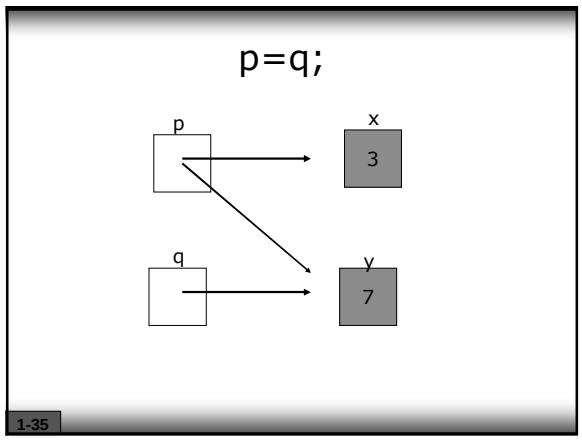
1-30

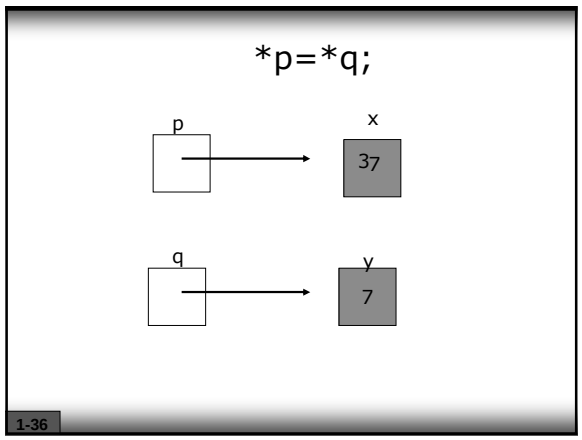

p
x
600000
7
500000
600000
int x;
int * p;
x=7;
p = &x;
1-31


p
x
600000
7
500000
600000
int x;
int * p;
x=7;
p = &x;
Utile rappresentazione grafica
*p == x
p
x
7
int x;
int * p;
x=7;
p = &x;
cout << *p;
1-32


Due modi di inizializzare un puntatore:
un indirizzo
un altro puntatore
int i =10;
int *p1=&i;
int *p2 = p1;
1-33


Descrivere questa situazione:
p
x
3
int x =3, y=7;
int *p = &x;
int *q = &y
q
y
7
Che succede con ciascuno dei seguenti?
p=q;
*p = *q
1-34


p=q;
p
x
3
q
y
7
1-35


*p=*q;
p
x
37
q
y
7
1-36

Attenzione ai tipi (rigidissimi)

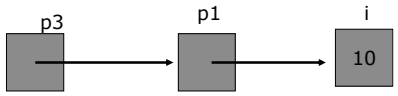
```
int i =10;
int *p1=&i;
int *p2 = p1;
```

```
int *p3 = i;
```
NO

```
int *p3   = &p1;

int **p3 = &p1;
```
NO, perche'?
Cosa sarebbe corretto?

1-37

```
int i =10,  *p1,  **p3;
p1=&i;
p3 = &p1;
```



1-38

Qualche esame

Esame marzo 2003

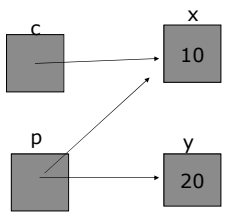
```
int x=10, *c =&x;
int y=20, *p=&y;

p=c;
cout <<(*p) <<endl;
```

Cosa succede?

1. Non compila
2. Stampa 20
3. Stampa 10
4. "segmentation fault"

1-40

```
int x=10, *c =&x;
int y=20, *p=&y;

p=c;
cout <<(*p) <<endl;
```



1-41

Esame 28 marzo 2003

```
char c = 'a', k = 'b', *p, **q=&p, *r = &k;
q=&r;
p=*q;
cout << *p << **q <<*r <<endl;
```

1-42

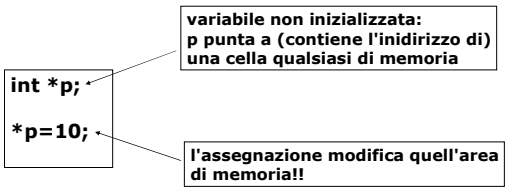
Attenzione!

- Ad una variabile puntatore e' potenzialmente assegnabile qualuque indirizzo della memoria.
- E' **vitale che punti solo agli oggetti cui si vuole accedere**

1-43

Evitate un grave errore

Prima di modificare un oggetto puntato,
verificare che il puntatore contenga un indirizzo valido!



1-44