Università di Torino – Facoltà di Scienze MFN
Corso di Studi in Informatica

Programmazione I - corso B
a.a. 2009-10

prof. Viviana Bono

Blocco 6 – Invariante di ciclo

Definire correttamente un ciclo: invarianti di ciclo

Richiamo della terminologia

Istruzione while: tutto il costrutto

`while` **(***BoolExpr***)**

Corpo

è UNA istruzione (composta):
più precisamente, è una istruzione while, la cui esecuzione consiste nella ripetizione (controllata da *BoolExpr* nel modo ben noto) delle istruzioni costituenti il *Corpo*.

Ciò che viene ripetuto è pertanto il *Corpo* del while, non l'istruzione while!

Analoga terminologia si usa per le altre *istruzioni iterative*: *for* e *do-while*.

Ideare un ciclo: si parte dallo "stato al passo generico"

Problema. Definire un metodo statico che legga da consolle una sequenza di interi avente almeno un elemento e terminata a un "non-intero", e ne scriva sullo schermo il massimo.

Risoluzione: Devo costruire un ciclo all'uscita del quale nella cella **max** sia contenuto il massimo.

Per far ciò, al generico passo la cella **max** deve contenere il massimo dei valori immessi fino a quel momento.

Assumendo che ciò sia vero per effetto dei passi precedenti e assumendo che la condizione hasNextInt() sia vera e che quindi io debba eseguire il corpo del ciclo,

che cosa devo fare nel corpo del ciclo?

1. leggere un nuovo intero: int x = tastiera.nextInt();
2. se è maggiore di **max** sostituirlo a max,
   altrimenti non far nulla: if(x > max) max = x;

Ideare un ciclo a partire dall'invariante (continua)

Inizializzazione.

Inizialmente, cioè prima di entrare nell'istruzione while, bisogna aver letto un intero, e tale intero è ovviamente il massimo:

int max = tastiera.nextInt();

In conclusione, il ciclo che risolve il problema è:

```
int max = tastiera.nextInt();
while(tastiera.hasNextInput()) {
  int x = tastiera.nextInt();
  if(x > max) max = x;
}
```

(naturalmente, bisognerà anche aver inizializzato *tastiera*, ecc.)

Che cosa è un invariante di un ciclo?

Invariante di ciclo = proposizione (riguardante i contenuti delle variabili di una procedura o programma) la quale:

- è vera immediatamente prima di eseguire l'istruzione iterativa;
- è vera dopo ogni ripetizione del corpo del ciclo;
- quindi, in particolare, è vera all'uscita dall'istruzione iterativa, cioè all'uscita "definitiva" dal ciclo.

È cioè una proposizione che descrive la situazione (della memoria) all'inizio di ogni ripetizione del corpo del ciclo.

Nell'esempio precedente l'invariante è:

La variabile **`max`** contiene il massimo dei valori immessi finora.

Come non ideare un ciclo

Se, nell'ideare un ciclo, si pensa subito al "primo passo" invece che al passo generico, ciò può indurre a scrivere programmi più complicati e spesso errati.

Ad esempio, nel problema del massimo, partendo dal primo passo qualcuno potrebbe ragionare così:

- poiché voglio ci sia almeno un elemento, faccio una lettura:

```
int x = tastiera.nextInt();
int max = Integer.MIN_VALUE;
```

- ora entro nel ciclo e faccio il primo passo:

  poiché ho già fatto la lettura fuori dal ciclo, faccio prima il confronto e poi una nuova lettura:

```
while(...) {
 if(x > max) max = x;
 x = tastiera.nextInt();
 ...
```

Come non ideare un ciclo (continua)

- In questo modo, ad ogni passo si confronta con max il valore x letto nel passo precedente: questo è piuttosto innaturale, e condurrà nella migliore delle ipotesi ad un programma inutilmente complicato e difficile da capire (perché si deve ragionare su due passi alla volta), o più probabilmente ad un programma errato come il seguente:

ATTENZIONE: PROGRAMMA ERRATO !

```
int x = tastiera.nextInt();
int max = Integer.MIN_VALUE;
while(tastiera.hasNextInput()) {
 if(x > max) max = x;
 x = tastiera.nextInt();
}
```


Nel seguito del corso vedremo esempi di problemi per i quali è quasi impossibile scrivere correttamente la soluzione iterativa senza partire col pensare alla "situazione al passo generico", cioè all'invariante di ciclo (ad es. ricerca binaria)