Corso di Laurea Ingegneria Informatica

Fondamenti di Informatica 1

Dispensa E03

Esempi di algoritmi e programmi

A. Miola

Settembre 2006

Contenuti

- Progettazione di algoritmi
- Problemi di ingresso - uscita
- Lettura e somma di due numeri interi
- Verifica anno bisestile
- Il giorno dopo…

Progettazione di algoritmi . . .

- **Scrivere un algoritmo è una attività complessa**
- **Una metodologia efficace nella progettazione degli algoritmi è basata sulla strategia top-down (dall'alto verso il basso)**
  - **la strategia top-down consiste nel decomporre iterativamente un problema da risolvere in sotto-problemi, fino a quando ciascun sotto-problema non possa essere risolto in modo elementare**
  - **la soluzione congiunta di tutti i sotto-problemi costituisce una soluzione per il problema iniziale**

. . . Progettazione di algoritmi

- **Progettazione di algoritmi per raffinamenti successivi**
  - scrivi una versione iniziale dell'algoritmo
    - una sequenza di uno o più passi
  - raffina ciascun passo dell'algoritmo, fintato che l'algoritmo non sia completo di tutti i dettagli
    - il raffinamento di un passo consiste nella sostituzione del passo con una istruzione semplice o con una istruzione di controllo
  - la progettazione di un algoritmo per raffinamenti successivi è una attività iterativa
    - il raffinamento di un passo può richiedere ulteriori raffinamenti

Problemi di ingresso-uscita . . .

- I problemi di ingresso-uscita sono una classe ristretta di problemi
  - i dati di ingresso del problema vengono letti dalla tastiera
  - i dati di uscita calcolati risolvendo il problema vengono visualizzati sullo schermo
- Un esempio di problema di ingresso-uscita
  - si vuole leggere dalla tastiera una coppia di numeri A e B, calcolarne la somma e visualizzarla sullo schermo
- Problemi di questo tipo portano alla scrittura di applicazioni (piuttosto che di metodi)

. . . Problemi di ingresso-uscita

- **Specifica di un problema di ingresso-uscita**
  - **dati di ingresso**
    - i parametri del problema, che devono essere letti dalla tastiera
  - **pre-condizione**
    - condizione che descrive le proprietà dei dati di ingresso
    - si assume che sia verificata, nel senso che si ipotizza che i dati di ingresso che vengono inseriti dalla tastiera soddisfano la pre-condizione
  - **dati di uscita**
    - i risultati del problema, che devono essere visualizzati sullo schermo
  - **post-condizione**
    - condizione che descrive le proprietà dei dati di uscita rispetto ai dati di ingresso

Lettura e somma di due numeri interi

- **Problema**
  - lettura e somma di due numeri interi
- **Dati di ingresso**
  - una coppia di numeri interi, A e B
- **Pre-condizione**
  - nessuna
- **Dati di uscita**
  - un numero S
- **Post-condizione**
  - S è la somma di A e B

Lettura e somma di due numeri interi

- **Si osservi la natura parametrica del problema della lettura e somma di due numeri interi**
  - **risolvere questo problema vuol dire identificare e implementare un algoritmo in grado di leggere e calcolare la somma di due numeri interi A e B**
    - **indipendentemente dagli specifici valori di A e B**
  - **in generale, i problemi di interesse per l'informatica sono parametrici, nel senso che dipendono da dati i cui valori non sono noti al momento in cui si vuole affrontare e risolvere il problema**

Algoritmo per la lettura e somma di due numeri interi

Algoritmo per il problema della lettura e somma di due numeri interi

1. leggi (dalla tastiera) i due numeri interi a e b
2. calcola la somma somma di a e b
3. visualizza somma (sullo schermo)

Una sequenza di tre passi

- il procedimento non è ancora sufficientemente dettagliato – per il nostro esecutore - e quindi i tre passi devono essere ulteriormente raffinati

Raffinamento del passo 1

- **Il passo 1 può essere raffinato usando una istruzione di controllo di sequenza**

1. leggi (dalla tastiera) i due numeri interi a e b

1.1 leggi il numero intero a dalla tastiera

1.2 leggi il numero intero b dalla tastiera

Raffinamento dei passi 1.1 e 1.2

- **I passi ottenuti dal raffinamento del passo 1 possono essere ulteriormente raffinati sulla base delle seguenti considerazioni**
  - **per la lettura dalla tastiera è possibile usare l'oggetto Lettore.in**
  - **per la lettura di un numero intero con l'oggetto Lettore.in va usato il metodo int leggiInt()**

1. ***leggi (dalla tastiera) i due numeri interi a e b***

1.1 ***leggi il numero intero a dalla tastiera***

```
a = Lettore.in.leggiInt();
```

1.2 ***leggi il numero intero b dalla tastiera***

```
b = Lettore.in.leggiInt();
```

Raffinamento dei passi 2 e 3

- **Il passo 2 può essere raffinato usando una istruzione semplice di assegnazione**

  2. *calcola la somma somma di a e b*

```
somma = a + b;
```

- **Il passo 3 può essere raffinato usando una istruzione semplice di invio di un messaggio all'oggetto System.out**

  3. *visualizza somma (sullo schermo)*

```
System.out.println(somma);
```

Algoritmo per la lettura e somma di due numeri interi

- **In sintesi, il problema della lettura e somma di due numeri interi può essere risolto dal seguente algoritmo**

1. *leggi (dalla tastiera) i due numeri interi a e b*
   1.1 *leggi il numero intero a dalla tastiera*

```
a = Lettore.in.leggiInt();
```

   1.2 *leggi il numero intero b dalla tastiera*

```
b = Lettore.in.leggiInt();
```

2. *calcola la somma somma di a e b*

```
somma = a + b;
```

3. *visualizza somma (sullo schermo)*

```
System.out.println(somma);
```

Variabili e oggetti per l'algoritmo

- **L'algoritmo per la lettura e somma di due numeri interi fa uso delle seguenti variabili e oggetti**
  - la variabile intera **a**
    - rappresenta il primo numero intero letto dalla tastiera
  - la variabile intera **b**
    - rappresenta il secondo numero intero letto dalla tastiera
  - la variabile intera **somma**
    - rappresenta la somma di **a** e **b**
  - l'oggetto **Lettore.in** che rappresenta la tastiera, da cui vanno letti i dati
    - per utilizzare questo oggetto non è necessaria nessuna variabile
  - l'oggetto **System.out** che rappresenta lo schermo, su cui vanno visualizzati i risultati
    - per utilizzare questo oggetto non è necessaria nessuna variabile

Programma per la lettura e somma di due numeri interi

- **L'algoritmo per la lettura e somma di due numeri interi può essere codificato da una applicazione Java**
  - **l'algoritmo viene implementato dal metodo main**

```
import fiji.io.*;
/* Applicazione che legge dalla tastiera due numeri
  interi
 * e ne calcola e visualizza la somma. */
class SommaDueNumeri {
    public static void main(String[] args) {
        ...
    }
}
```

Programma per la lettura e somma di due numeri interi

```
public static void main(String[] args) {
     int a;              // il primo numero intero
     int b;              // il secondo numero intero
     int somma;          // la somma di a e b
  /* leggi dalla tastiera i due numeri interi a e b */
     System.out.println("Scrivi due numeri interi");
  /* leggi il numero intero a dalla tastiera */
     a = Lettore.in.leggiInt();
  /* leggi il numero intero b dalla tastiera */
     b = Lettore.in.leggiInt();
  /* calcola la somma somma di a e b */
     somma = a+b;
  /* visualizza somma (sullo schermo) */
     System.out.print("La somma dei due numeri è ");
     System.out.println(somma);
 }
```

Verifica se un anno è bisestile

Problema

- Scrivere un'applicazione che verifichi se un certo anno sia bisestile

Dati di ingresso

- Anno

Pre-condizione

- L'anno è un intero positivo maggiore di 1600

Dati di uscita

- Valore booleano: **true** se l'anno è bisestile, altrimenti **false**

Post-condizione

Condizioni

❑ Un anno (>1600) è bisestile se:

- **è divisibile per 4; (condizione non sufficiente)**
  - **L'anno 2004 è bisestile**
  - **L'anno 1900 non è bisestile**

- **può essere divisibile per 100 (condizione non ancora sufficiente);**
  - **L'anno 2000 è bisestile**
  - **L'anno 1900 non è bisestile**

- **oltre ad essere divisibile per 4, se è divisibile per 100, allora deve essere necessariamente divisibile anche per 400;**

Algoritmo per verificare se un anno è bisestile

- Dato un numero intero data, verifico che tutte le condizioni siano verificate:

- verifica che data sia divisibile per 4
- se true allora
  - verifica che data sia divisibile per 100
  - se true allora
    - verifica che data è anche divisibile per 400
    - se true allora restituisci “bisestile”
    - altrimenti restituisci “non bisestile”
  - altrimenti restituisci “bisestile”
- altrimenti restituisci “non bisestile”

Codifica Java

- Realizziamo un metodo bisestile che ha come unico parametro di input un intero data, che rappresenta un anno, e restituisce true se l'anno è bisestile, altrimenti false

```
public static boolean bisestile(int data){
     boolean bis;   //vero se l'anno e' bisestile

     if(data%4==0){ // se è divisibile per 4
             if(data%100==0){
                     /* se è divisibile per 100        */
                     /* allora per essere bisestile    */
             /* deve essere divisibile per 400 */
                     if(data%400==0)
                             bis = true;
                     else    bis = false;
             }
             /* è divisibile per 4 ma non per 100 */
             else bis = true;
     }
     /* non e' divisibile per 4 */
     else bis = false;

     return bis;
}
```

La classe Bisestile

```
import fiji.io.*;
/* Applicazione che verifica se un anno e' bisestile
 */
class Bisestile{

  public static void main(String[] args){
    //precondizione: anno >1600

    int anno;  //INPUT
    /*lettura dei parametri di input */
    System.out.println("Scrivi una data corrisp. ad un anno (>1600)
                                                              ");
    anno = Lettore.in.leggiInt();

    if (bisestile(anno))
        System.out.println("e' bisestile");
    else
        System.out.println("NON e' bisestile");

  }//end main
```

Attenzione

- **In questa applicazione non vi è nessuna verifica delle precondizioni !!!**

Il giorno dopo

Problema

- Applicazione che calcola la data del giorno successivo ad una data ricevuta in input

Esempio: 13/1/2005 ⇒ 14/1/2005

Dati di ingresso

- La data di cui si vuole calcolare il giorno successivo espressa tramite tre numeri interi che rappresentano il giorno, il mese e l'anno.

Pre-condizione

- L'anno è un intero positivo maggiore di 1600
- Il mese è un intero positivo minore o uguale di 12
- Il giorno è un intero positivo generalmente minore o uguale di 31, ma questo valore è funzione del mese.

Dati di uscita

- Il giorno successivo

Condizioni

- **Una *data* è rappresentata da tre diversi input interi: giorno, mese, anno.**
- **Generalmente la data del giorno dopo viene calcolata incrementando di uno il valore del giorno. 13/1/2005 ⇒ 14/1/2005**
- **Dobbiamo fare attenzione alle condizioni limite:**
  - Se il giorno corrisponde all'ultimo del mese.

    29/2/2004 ⇒ 1/3/2004

  - Se il giorno è l'ultimo del mese ed il mese è dicembre.

    31/12/2004 ⇒ 1/1/2005

L’algoritmo

- Letti giorno, mese e anno verifico che non mi trovo in una condizione limite e incremento il giorno.
- Impiego la variabile intera giorni_del_mese per rappresentare il massimo numero dei giorni per il mese.

…L'algoritmo

- Verifico che giorno sia uguale a giorni_del_mese
- Se *true*, allora
  - giorno viene posto a 1
  - verifico che mese sia uguale a 12 (dicembre)
  - se *true*, allora
    - mese viene posto a 1
    - incremento anno di 1
  - Altrimenti, incremento mese di 1
- Altrimenti incremento giorno di 1

I giorni di un mese

- **Per calcolare il massimo numero di giorni per un dato mese, impiego il metodo GiorniMese, che tra le altre cose terrà conto se un anno è bisestile o meno:**

```
public static int GiorniMese(int mese,int anno){
   int giorni;
   if(mese==2)
     if(Bisestile.bisestile(anno))
         giorni = 29;
     else giorni = 28;
   else
     if(mese==4 || mese==6 || mese==9 || mese==11)
         giorni = 30;
     else giorni = 31;
  return giorni;
}
```

La classe: IlGiornoDopo

```
import fiji.io.*;
/* Applicazione che calcola la data del giorno dopo */
class IlGiornoDopo{

  public static void main(String[] args){

    int giorno, mese, anno;  //INPUT
    int g,m,a;//OUTPUT
    int giorni_del_mese; //quanti giorni ha il mese m

    /*lettura della data di partenza senza controllo
     * sulla correttezza dei dati */
    giorno = Lettore.in.leggiInt();
    mese = Lettore.in.leggiInt();
    anno = Lettore.in.leggiInt();
```

… La classe: IlGiornoDopo

```
        a=anno; //nel caso in cui non vengano modificate
        m=mese;
        /* calcolo dei giorni del mese */
        giorni_del_mese=GiorniMese(mese, anno);

        /* calcolo della data successiva */
        /* se il giorno e' l'ultimo del mese */
        if (giorno==giorni_del_mese){
                g=1;
                if (mese==12){   //ultimo giorno dell'anno
                        m=1;
                        a=anno+1;}
                else
                        m=mese+1;
               }
        else   //nel caso generale basta incrementare il giorno
                g=giorno+1;

        System.out.println(g + " " +m+ " " + a);
}
```

Riferimenti al libro di testo

- Per lo studio di questi argomenti si fa riferimento al libro di testo, e in particolare al capitolo
  - 5 su Problemi, algoritmi e oggetti
- Con particolare riferimento ai seguenti paragrafi
  - 5.2.3 e 5.2.4