Cicli `while`

- Esegue un blocco di codice ripetutamente
- Una condizioni controlla quante volte il ciclo è ripetuto

```
while (condition)
    statement
```

- Di norma l'istruzione è in verità un blocco di istruzioni (insieme di istruzioni delimitato da `{ }`)

Calcolare la crescita di un investimento

- Quando il saldo sul conto ha raggiunto una certa cifra?

```
while (balance < targetBalance)
{
   years++;
   double interest = balance * rate / 100;
   balance = balance + interest;
}
```

Diagramma di flusso del `while`

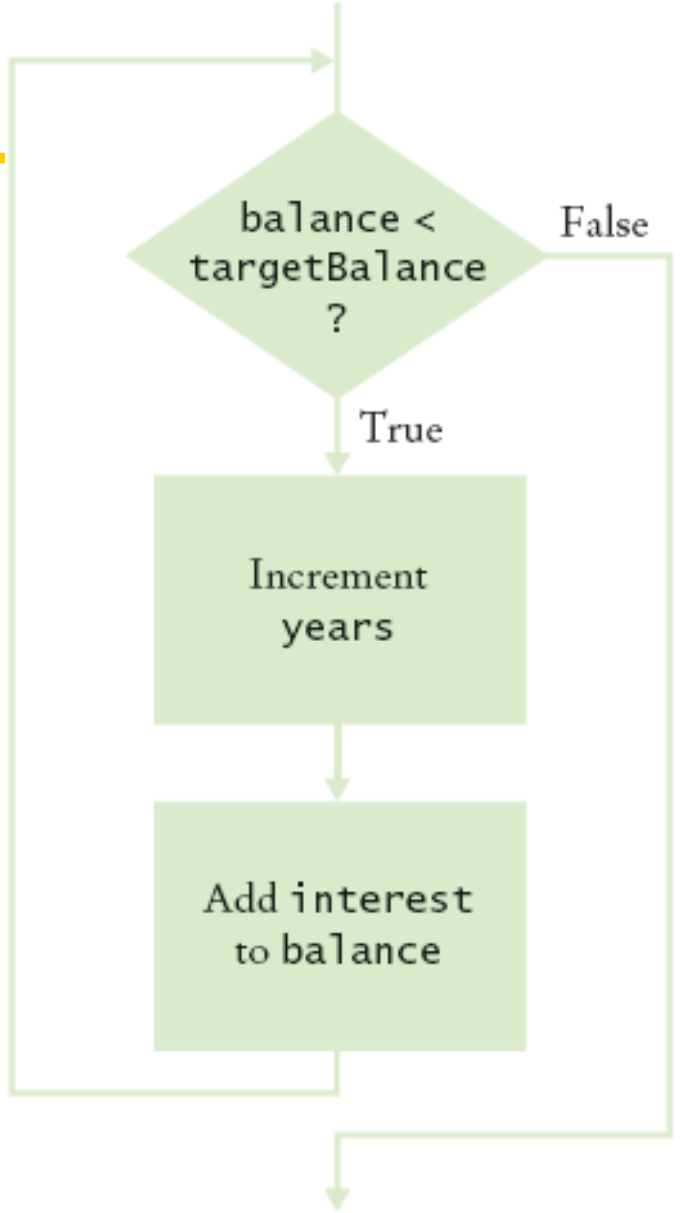


Figure 1 Flowchart of a `while` Loop

Sintassi 6.1 L'istruzione `while`

```
while (condition)
   statement
```

Esempio:

```
while (balance < targetBalance)
{
  years++;
  double interest = balance * rate / 100;
  balance = balance + interest;
}
```

Scopo:

Eseguire ripetutamente un'istruzione fintantoché la condizione è vera

Cicli do

- Esegue il corpo del ciclo almeno una volta:

```
do
   statement
while (condition);
```

- Esempio: controllare se un input è valido

```
double value;
do
{
   System.out.print("Please enter a positive number: ");
   value = in.nextDouble();
}
 while (value <= 0);
```

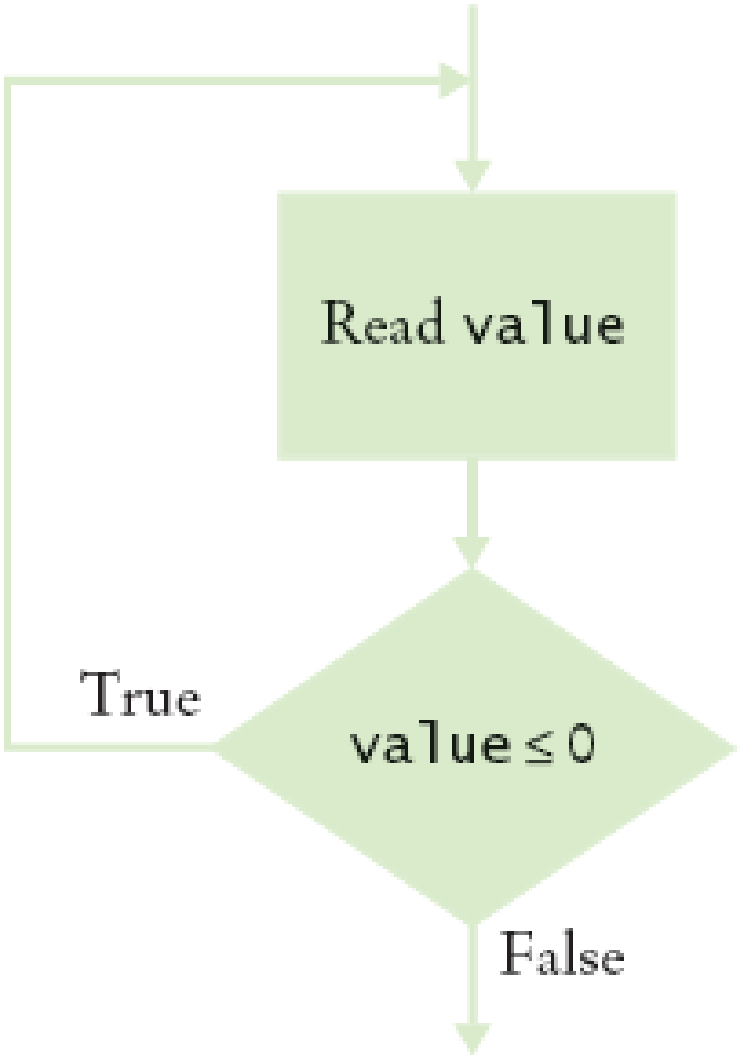
Continua

Cicli `do` (continua)

- Alternativa:

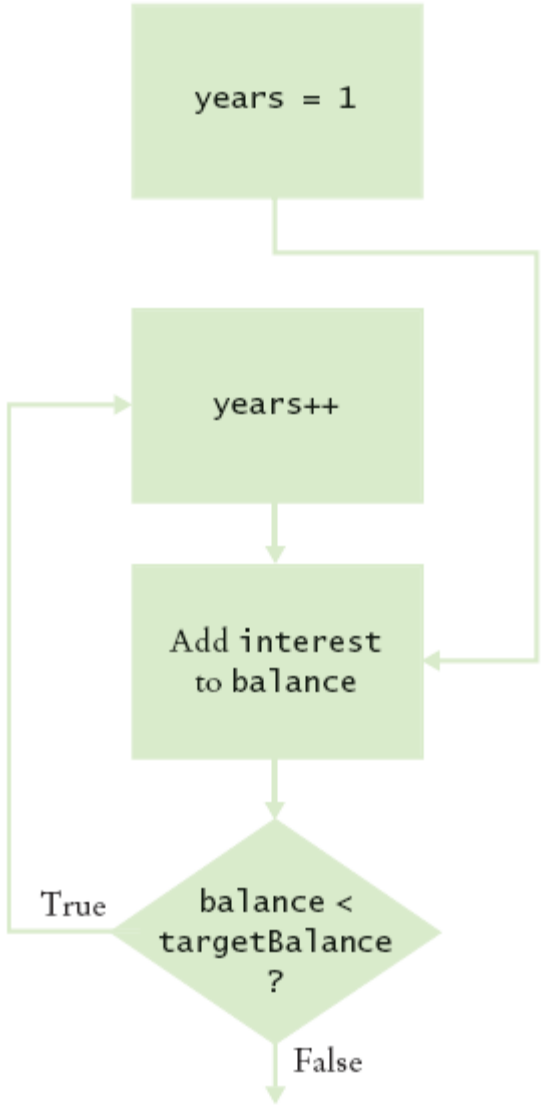
```
boolean done = false;
while (!done)
{
   System.out.print("Please enter a positive number: ");
   value = in.nextDouble();
   if (value > 0) done = true;
}
```

Diagramma di flusso del ciclo `do`



Flowchart of a do Loop

Codice a spaghetti



Spaghetti Code

Cicli `for`

- `for (`*`initialization`*`; `*`condition`*`; `*`update`*`)`

  `statement`

- Esempio:

```
for (int i = 1; i <= n; i++)
{
   double interest = balance * rate / 100;
   balance = balance + interest;
}
```

- Equivalente a

```
initialization;
while (condition)
{ statement;
  update; }
```

Continua

Cicli `for` (continua)

- Altri esempi:

```
for (years = n; years > 0; years--) . . .
for (x = -10; x <= 10; x = x + 0.5) . . .
```

Diagramma di flusso del `for`

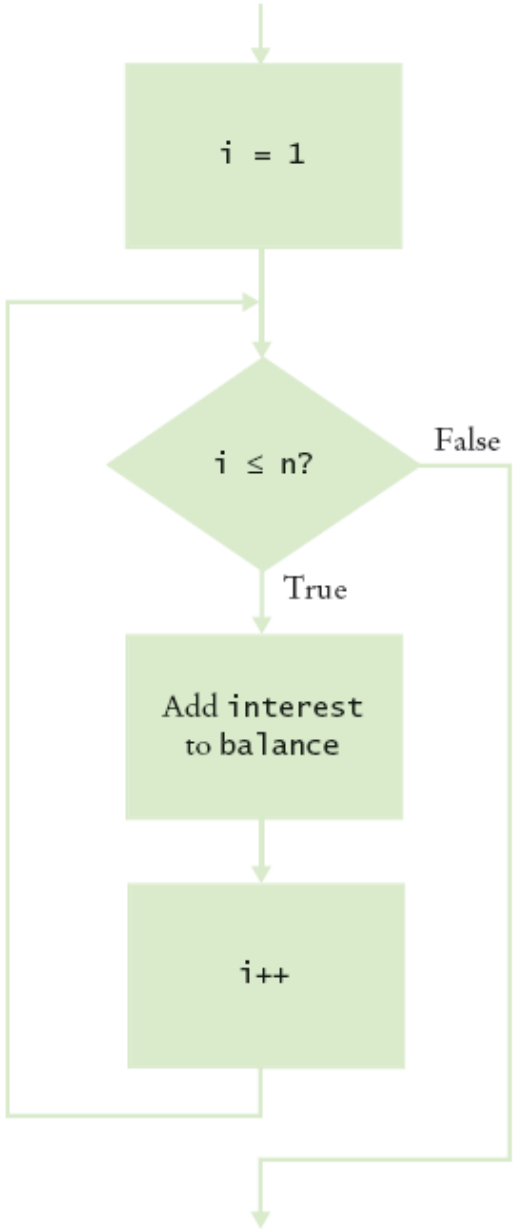


Figure 2 Flowchart of a for Loop

Sintassi 6.2 L'istruzione `for`

```
for (initialization; condition; update)
   statement
```

Esempio:

```
for (int i = 1; i <= n; i++)
{
   double interest = balance * rate / 100;
   balance = balance + interest;
}
```

Scopo:

Eseguire un'inizializzazione e mantenere un enunciato in esecuzione, aggiornando un'espressione mentre una determinata condizione è vera.

Il problema del “ciclo e mezzo”

- Talvolta la condizione che fa terminare un ciclo può essere valutata soltanto nel mezzo del ciclo stesso
- In tal caso si introduca una variabile booleana per controllare il ciclo:

```
boolean done = false;
while (!done)
{
    Print prompt
    String input = read input;
    if (end of input indicated)
       done = true;
    else
    {
       Process input
    }
}
```

Numeri casuali e simulazioni

- In una simulazione vengono generati ripetutamente numeri casuali e questi vengono usati per simulare una certa attività
- Generatore di numeri casuali

```
Random generator = new Random();

int n = generator.nextInt(a); // 0 < = n < a

double x = generator.nextDouble(); // 0 <= x < 1
```

- Simulazione del lancio di un dado (intero casuale tra 1 e 6)

```
int d = 1 + generator.nextInt(6);
```

ch06/random1/Die.java

```
01: import java.util.Random;
02:
03: /**
04:    This class models a die that, when cast, lands on a random
05:    face.
06: */
07: public class Die
08: {
09:    /**
10:       Constructs a die with a given number of sides.
11:       @param s the number of sides, e.g. 6 for a normal die
12:    */
13:    public Die(int s)
14:    {
15:       sides = s;
16:       generator = new Random();
17:    }
18:
19:    /**
20:       Simulates a throw of the die
21:       @return the face of the die
22:    */
```

Continua

ch06/random1/Die.java (cont.)

```
23:    public int cast()
24:    {
25:       return 1 + generator.nextInt(sides);
26:    }
27:
28:    private Random generator;
29:    private int sides;
30: }
```

ch06/random1/DieSimulator.java

```
01: /**
02:    This program simulates casting a die ten times.
03: */
04: public class DieSimulator
05: {
06:    public static void main(String[] args)
07:    {
08:       Die d = new Die(6);
09:       final int TRIES = 10;
10:       for (int i = 1; i <= TRIES; i++)
11:       {
12:          int n = d.cast();
13:          System.out.print(n + " ");
14:       }
15:       System.out.println();
16:    }
17: }
```

ch06/random1/DieSimulator.java (cont.)

Output:

```
6 5 6 3 2 6 3 4 4 1
```

Second Run:

```
3 2 2 1 6 5 3 4 1 2
```