

Iterazioni

L'istruzione while

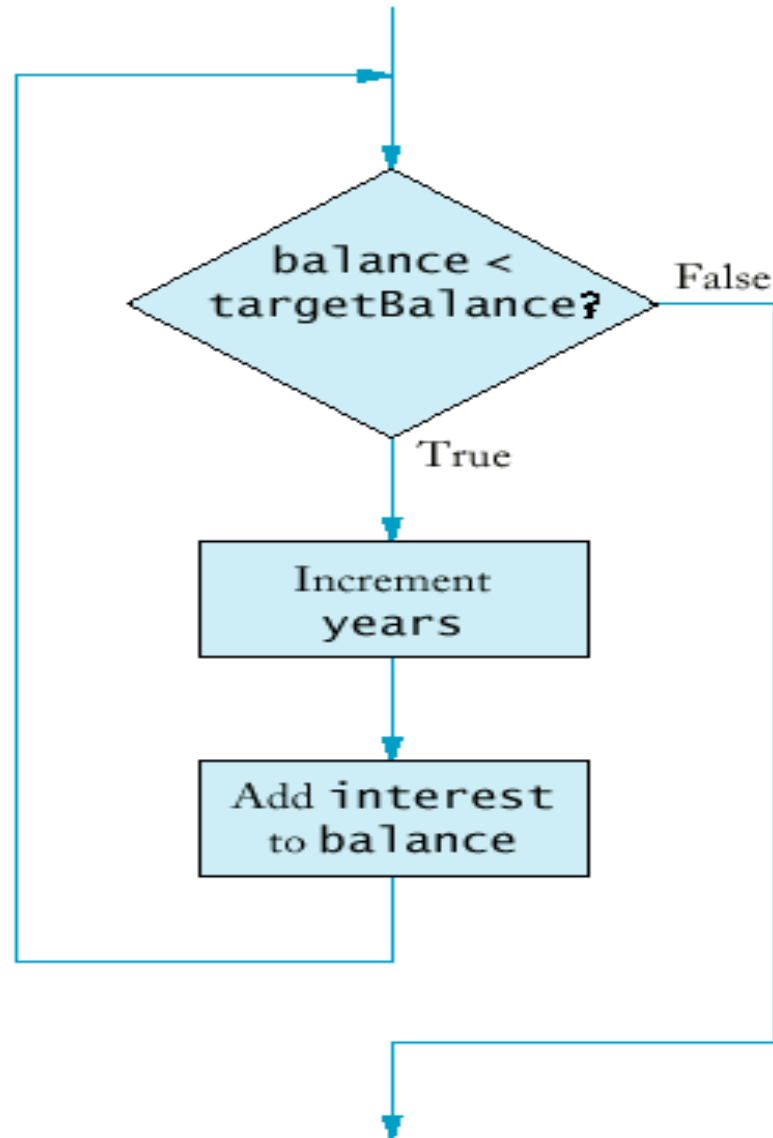
while (condition)
istruzione

- Ripete l'esecuzione di istruzione fino a che la condizione resta vera

while (balance < targetBalance)

```
{  
    year++;  
    double interest = balance * rate / 100;  
    balance = balance + interest;  
}
```

Diagramma di flusso per il ciclo while



File Investment.java

```
public class Investment {  
    public Investment(double  
        aBalance, double aRate) {  
        balance = aBalance;  
        rate = aRate;  
        years = 0;  
    }  
    public double getBalance() {  
        return balance;  
    }  
  
    public int getYears() {  
        return years;  
    }  
}
```

```
//accumula interessi fino a che il target è  
raggiunto  
public void waitForBalance(double  
    targetBalance) {  
    while (balance < targetBalance) {  
        years++;  
        double interest =  
            balance * rate / 100;  
        balance = balance + interest;  
    }  
}  
  
private double balance;  
private double rate;  
private int years;  
}
```

Errori comuni: I loop infiniti

1. **while** (year < 20) {
 balance = balance + balance * rate / 100;
}
2. **while** (year > 0) {
 year++;
}

L'istruzione do/while

- Esegue il corpo del ciclo almeno una volta:

do

istruzione

while (*condition*);

- Esempio:

double value;

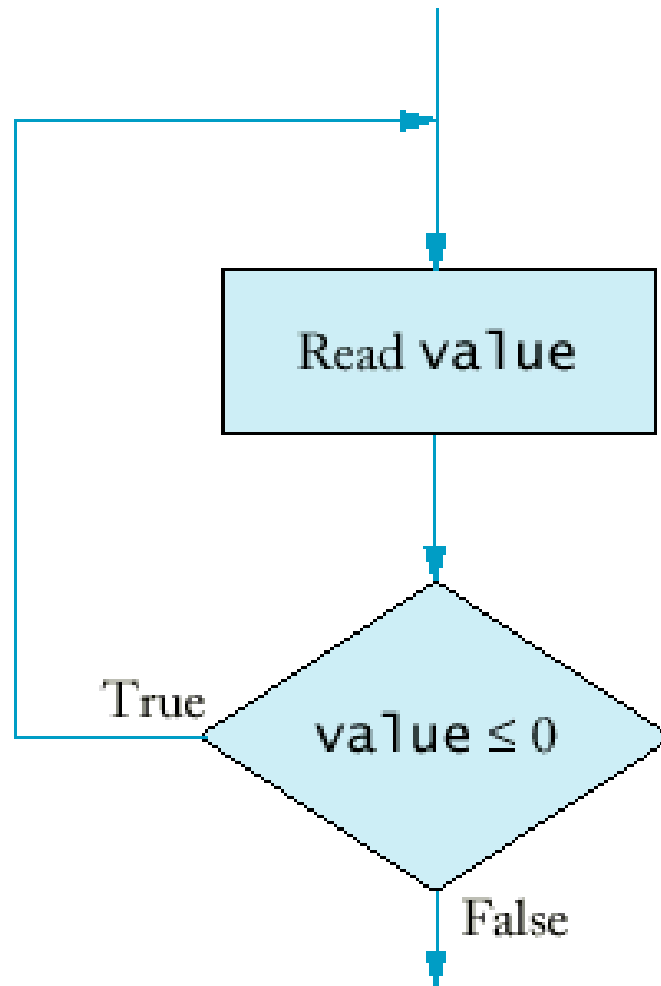
do

{

int input = console.nextInt();

} **while** (input <= 0);

Daigramma di flusso per do Loop



L'istruzione `for`

for (*initialization; condition; update*)
 istruzione

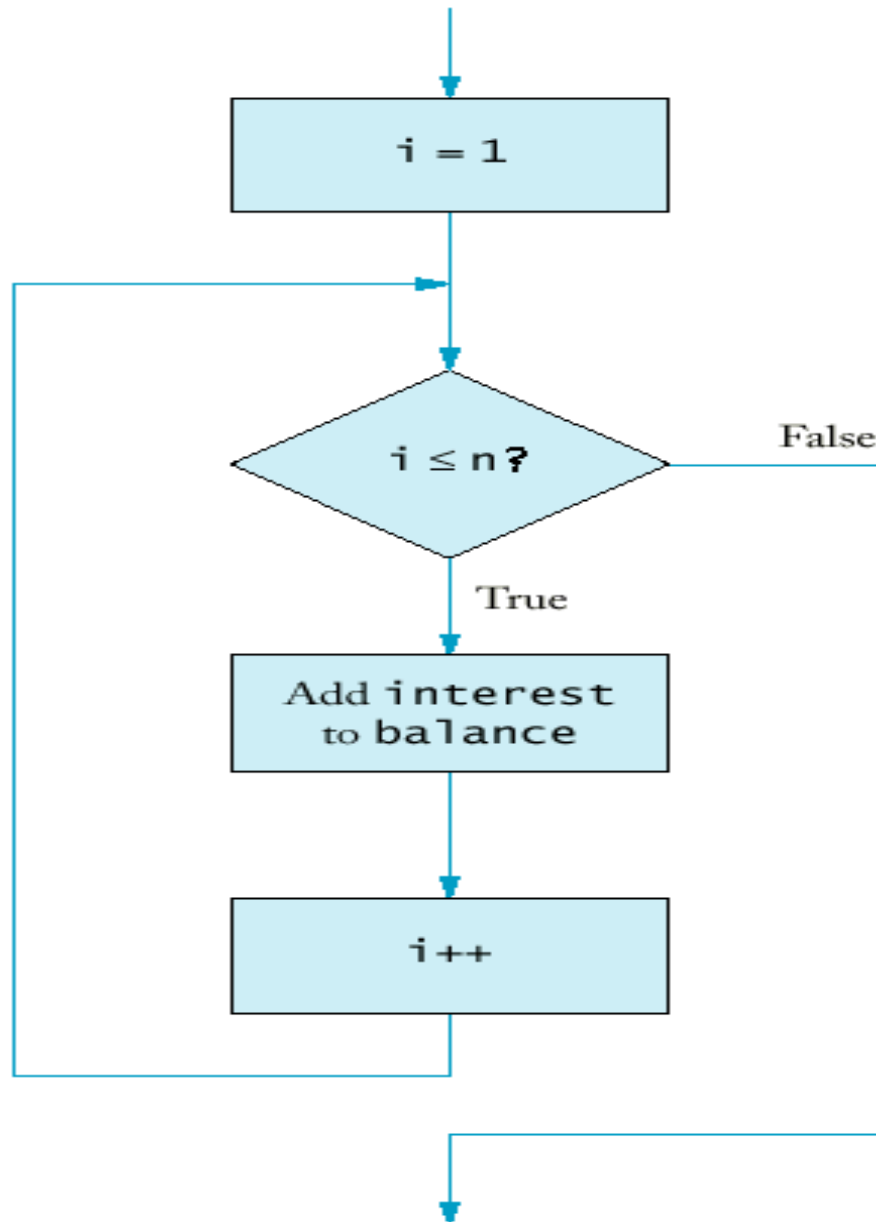
Esempio:

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

Equivalente a

```
    Inizializzazione;  
    while (condizione) {  
        istruzione; update; }
```


Diagramma di flusso ciclo for



Esempio

- Aggiungiamo alla classe **Investment** il metodo **waitYears** che accumula gli interessi corrispondenti ad un certo numero di anni

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

Loop annidati

- **Esempio:** stampiamo il triangolo

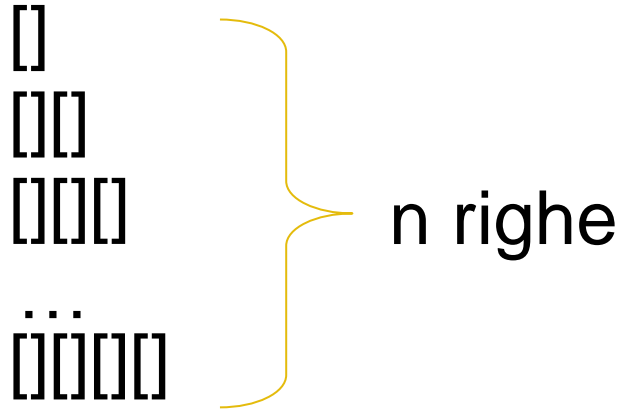


Diagram illustrating the output of the nested loop program, showing a triangle of brackets ([]) with a brace indicating n righe (rows).

```
[]  
[][]  
[][][]  
[] [] [] []
```

```
for (int i = 1; i <= n; i++)  
{  
    // forma una riga del triangolo  
    for (int j = 1; j <= i; j++)  
        r = r + "[]";  
    r = r + "\n";  
}
```

Es.: lettura ciclica input (test interno)

```
import java.util.Scanner;
public class SommaInput{
    public static void main(String[] args) {
        double somma=0;
        Scanner in = new Scanner(System.in);
        System.out.println("Immetti valore oppure Q per uscire");
        boolean done = false;
        while (!done) {
            String input = in.next();
            if (input.equalsIgnoreCase("Q"))
                done = true;
            else {
                double x = Double.parseDouble(input);
                somma+=x;
            }
        }
        System.out.println("la somma e`:" + somma);
    }
}
```

Es.: lettura ciclica input (test inizio)

```
import java.util.Scanner;
```

```
public class SommaInput{
```

```
    public static void main(String[] args) {
```

```
        double somma=0;
```

```
        String input;
```

```
        Scanner in = new Scanner(System.in);
```

```
        System.out.println("Immetti valore oppure Q per uscire");
```

```
        while (!(input = in.next()).equalsIgnoreCase("Q")) {
```

```
            double x = Double.parseDouble(input);
```

```
            somma+=x;
```

```
        }
```

```
        System.out.println("la somma e`:" + somma);
```

```
    }
```

```
}
```

Scandire i caratteri di una stringa

- `s.charAt(i)` è l' $(i+1)$ -esimo carattere della stringa `s`

```
for (int i = 0; i < s.length(); i++)  
{  
    char c = s.charAt(i);  
    ...  
}
```

Esempio: un programma che conta le vocali

- `s.indexOf(ch)` è l'indice della posizione in cui `c` appare per la prima volta in `s`, o -1 se `c` non appare in `s`

```
int numVocali = 0;
String vocali = "aeiou";
for (int i = 0; i < s.length(); i++)
{
    char c = s.charAt(i);
    if (vocali.indexOf(c) >= 0)
        numVocali++;
}
```

Problema

- Vogliamo costruire una classe Dado che modelli un dado
 - l'interfaccia pubblica deve contenere un metodo che simuli il lancio di un dado restituendo a caso il valore di una delle sue facce
 - serve un generatore di numeri casuali
-

Numeri casuali

- La classe Random modella un generatore di numeri casuali
- Random generatore = new Random();
 - crea un generatore di numeri casuali
- int n = generatore.nextInt(a);
 - restituisce un intero **n** con $0 \leq n < a$
- double x = generatore.nextDouble();
 - restituisce un double **x** con $0 \leq x < 1$

Esempio uso di Random

```
import java.util.Random;
public class Dado {
    //costruttore che costruisce un dado
    // con s facce

    public Dado(int s) {
        facce = s;
        generatore = new Random();
    }

    public int lancia() {
        return 1 +
            generatore.nextInt(facce);
    }

    private Random generatore;
    private int facce;
}
```

// Questo programma simula 10 lanci del
dado

```
public class DadoTester {
    public static void main(String[] args)
    {
        Dado d = new Dado(6);
        final int LANCI = 10;
        for (int i = 1; i <= LANCI; i++) {
            int n = d.lancia();
            System.out.print(n + " ");
        }
        System.out.println();
    }
}
```