Iterazioni

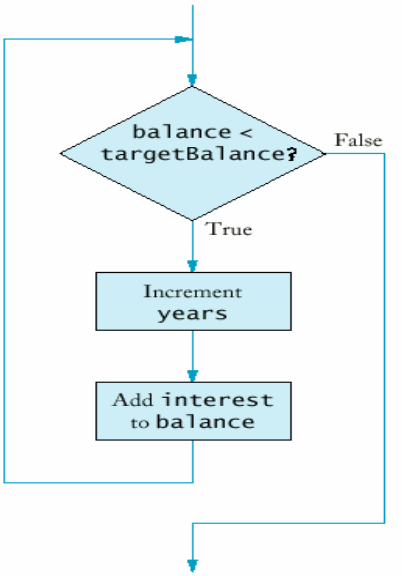
L'istruzione while

```
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

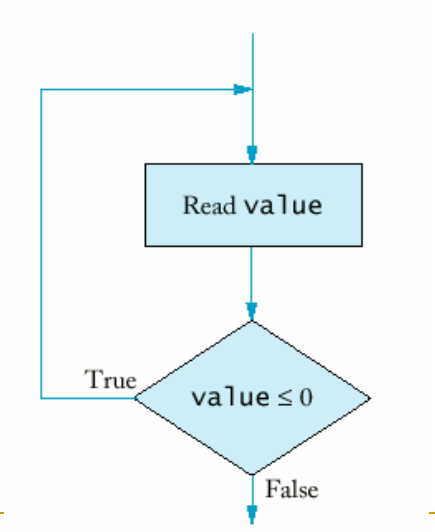
- Esegue il corpo del ciclo almeno una volta:

```
do
   istruzione
while (condition);
```

- Esempio:

```
double value;
do
{
   String input = console.readLine();
   value = Integer.parseInt(input);
} while (input <= 0);
```

Daigramma di flusso per do Loop



L'istruzione for

for (*initialization*; *condition*; *update*)
istruzione

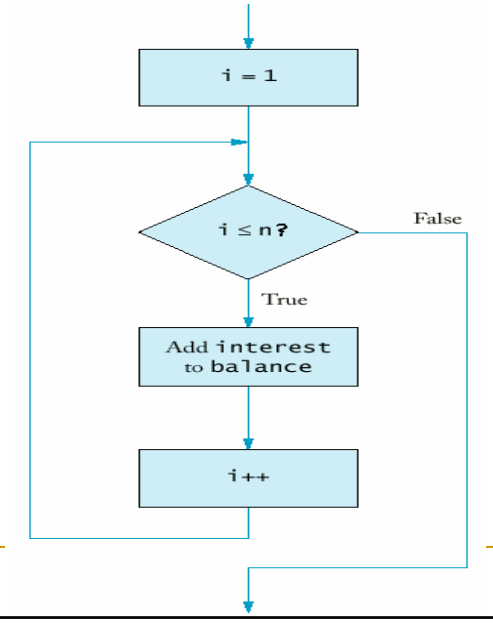
Esempio:

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

Equivalente a

Initializzazione;
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 innestati

- Esempio: stampiamo il triangolo

[]
[][]
[][][]
…
[][][][]

n righe

```
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.io.*;
public class SommaInput{
    public static void main(String[] args) throws IOException {
        double somma=0;
        InputStreamReader reader = new InputStreamReader(System.in);
        BufferedReader console = new BufferedReader(reader);
        boolean done = false;
        while (!done) {
            String input =   console.readLine();
            if (input.length() == 0)
                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.io.*;

public class SommaInput{
    public static void main(String[] args) throws IOException {
        double somma=0;
        String input;
        InputStreamReader reader = new  InputStreamReader(System.in);
        BufferedReader console = new BufferedReader(reader);
        while ((input =  console.readLine()).length() != 0) {
            double x = Double.parseDouble(input);
            somma+=x;
        }
     System.out.println("la somma e`:"+ somma);
    }
}
```

I token

- La classe StringTokenizer può scomporre una stringa nei suoi elementi (*token*)
- I token sono separati da spaziature (spazi, tab, e caratteri per andare a capo)
  - Esempio: "4.3 7 -2" e` scomposta in tre token: "4.3", "7", "-2"
- StringTokenizer tokenizer = new StringTokenizer(input);
  - Crea uno StringTokenizer per la stringa input
  - E` possibile usare un separatore diverso indicandolo come secondo argomento del costruttore
- tokenizer.hasMoreTokens()
  - Restituisce **true** se ci sono altri token da estrarre e false altrimenti
- tokenizer.nextToken()
  - restituisce il prossimo token

Esempio uso StringTokenizer

```
import java.util.StringTokenizer;
import java.io.*;

public class TokenTest {
    public static void main(String[] args) throws IOException {
      double somma=0;
      String input;
      InputStreamReader reader = new InputStreamReader(System.in);
      BufferedReader console = new BufferedReader(reader);
      input= console.readLine();
      StringTokenizer tokenizer = new StringTokenizer(input);
      while (tokenizer.hasMoreTokens()){
          String token = tokenizer.nextToken();
          double x = Double.parseDouble(token);
          somma+=x;
      }
      System.out.println("la somma e`:"+ somma);
    }
}
```

Scandire i caratteri di una stringa

- s.charAt(i) è l' i-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 modella un dado che quando viene lanciato cade a **caso** su uno dei suoi lati

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);
  - retituisce un intero n con 0 <= n < a
- double x = generatore.nextDouble();
  - restitisce un double x con 0 <= 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 +
            generator.nextInt(facce);
    }

    private Random generator;
    private int facce;
}
```

```
// Questo programma simula 10 lanci
     del dado

public class TestaDado {
   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();
   }
}
```