ΤΕΧΝΙΚΕΣ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΟΥΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Εισαγωγή στη Java

Είσοδος

- Χρησιμοποιούμε την κλάση Scanner της Java
  - import java.util.Scanner;
- Αρχικοποιείται με το ρεύμα εισόδου:
  - Scanner in = new Scanner(System.in);
- Μπορούμε να καλέσουμε μεθόδους της Scanner για να διαβάσουμε κάτι από την είσοδο
  - nextLine(): διαβάζει μέχρι να βρει τον χαρακτήρα '\n'
  - next(): διαβάζει το επόμενο String μέχρι να βρει λευκό χαρακτήρα
  - nextInt(): διαβάζει τον επόμενο int
  - nextDouble(): διαβάζει τον επόμενο double.
  - nextBoolean(): διαβάζει τον επόμενο boolean.

Παράδειγμα

Με την εντολή αυτή φέρνουμε την κλάση Scanner μέσα στο πρόγραμμα μας ώστε να μπορούμε να φτιάξουμε αντικείμενα Scanner

```
import java.util.Scanner;

class TestIO
{
	public static void main(String args[])
	{
		System.out.println("Say Something:");
		Scanner input = new Scanner(System.in);
		String line = input.nextLine();
		System.out.println(line);
	}
}
```

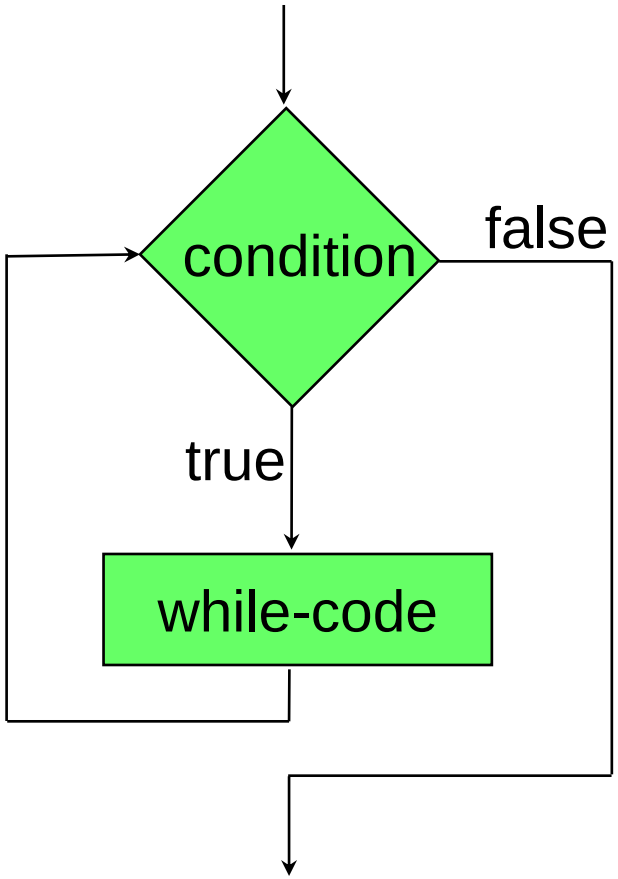
new: δημιουργεί ένα αντικείμενο τύπου **Scanner** (μία μεταβλητή) με το οποίο μπορούμε πλέον να διαβάζουμε από την είσοδο

Επαναλήψεις - While statement

- Στην Java το while statement έχει το εξής συντακτικό

```
while(condition)
{
      …while-code block…
}
```

  - Αν η συνθήκη είναι αληθής τότε εκτελείται το block κώδικα while-code
  - Ο while-code block κώδικας υλοποιεί τις επαναλήψεις και αλλάζει την συνθήκη.
  - Στο τέλος του while-code block η συνθήκη αξιολογείται εκ νέου
  - Ο κώδικας επαναλαμβάνεται μέχρι η συνθήκη να γίνει ψευδής.



Παράδειγμα

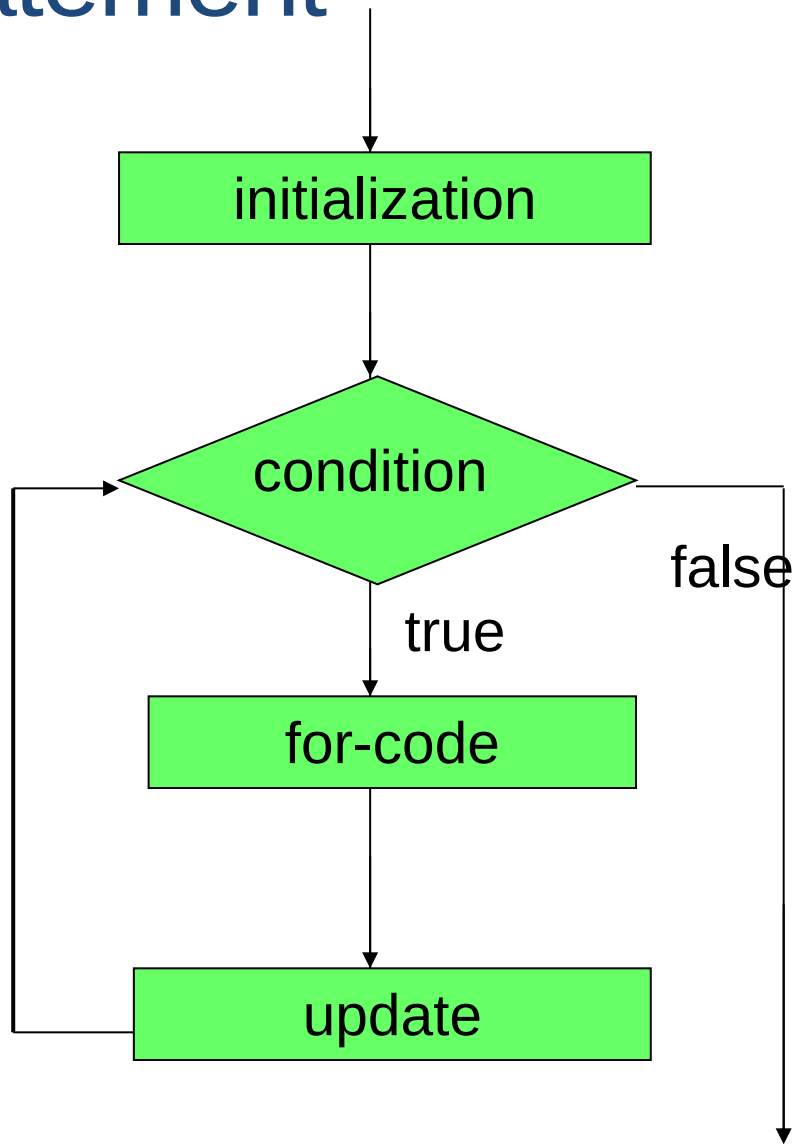
```
Scanner inputReader = new Scanner(System.in);
String input = inputReader.next();

while(input.equals("Yes"))
{
    System.out.println("Do you want to continue?");
    input = inputReader.next();
}
```

Επαναλήψεις – for statement

- Στην Java το for statement έχει το εξής συντακτικό

```
for(intialization;
    condition;
    update)
{
    …for-code block…
}
```

- Το όρισμα του for έχει 3 κομμάτια χωρισμένα με ;
  - Την αρχικοποίηση (initialization section): εκτελείται πάντα μία μόνο φορά
  - Τη λογική συνθήκη (condition): εκτιμάται πριν από κάθε επανάληψη.
  - Την ενημέρωση (update expression): υπολογίζεται μετά το κυρίως σώμα της επανάληψης.
  - Ο κώδικας επαναλαμβάνεται μέχρι η συνθήκη να γίνει ψευδής.



Παράδειγμα

```
for(int i = 0; i < 10; i = i+1)
{
    System.out.println("i = " + i);
}
```

Ορισμός της μεταβλητής `i`

Ανάθεση: υπολογίζεται η τιμή του `i+1` και ανατίθεται στη μεταβλητή `i`.

- Ισοδύναμο με while

```
int i = 0;
while(i < 10)
{
    System.out.println("i = " + i);
    i = i+1;
}
```

Παράδειγμα

```
for(int i = 0; i < 10; i ++)
{
     System.out.println("i = " + i);
}
```

i++ ισοδύναμο με το i = i+1

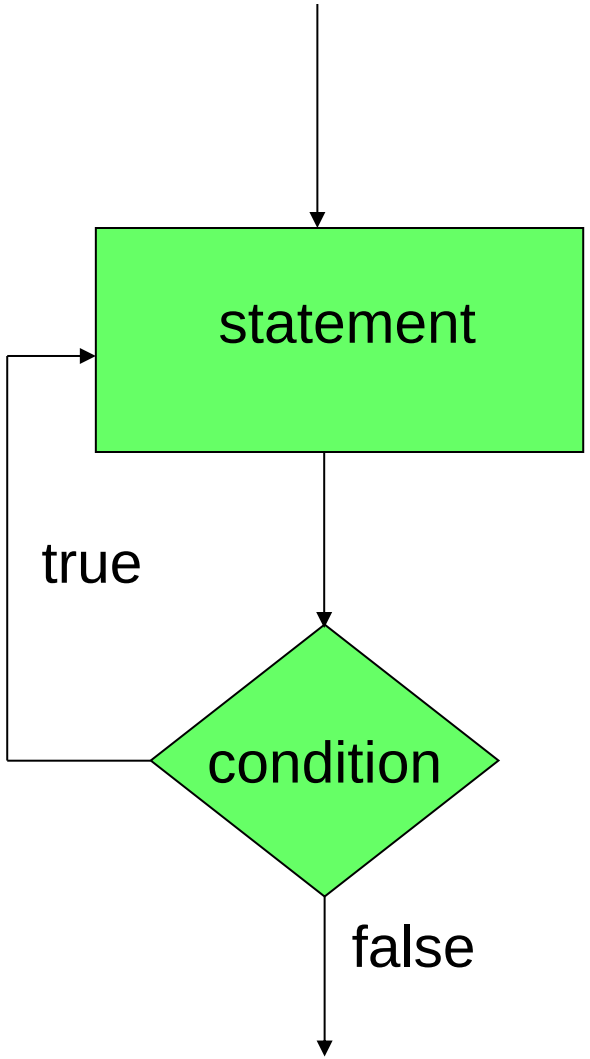
- Ισοδύναμο με while

```
int i = 0;
while(i < 10)
{
     System.out.println("i = " + i);
     i ++;
}
```

To Do-While statement

- Ένα **do while** statement έχει το εξής συντακτικό:

```
Initialize
do
{
      …while-code block…

}while(condition)
```



- Το while code εκτελείται τουλάχιστον μία φορά; Μετά αν η συνθήκη είναι αληθής ο κώδικας εκτελείται ξανά.
- Το while code εκτελούν το βρόγχο και αλλάζουν την συνθήκη.

Παράδειγμα

- Κάνετε πρόγραμμα που παίρνει σαν είσοδο ένα αριθμό και υλοποιεί μια αντίστροφη μέτρηση. Αν ο αριθμός είναι θετικός η αντίστροφη μέτρηση γίνεται προς τα κάτω μέχρι το μηδέν, αν είναι αρνητικός γίνεται προς τα πάνω μέχρι το μηδέν. Η διαδικασία επαναλαμβάνεται μέχρι ο χρήστης να δώσει την τιμή μηδέν.

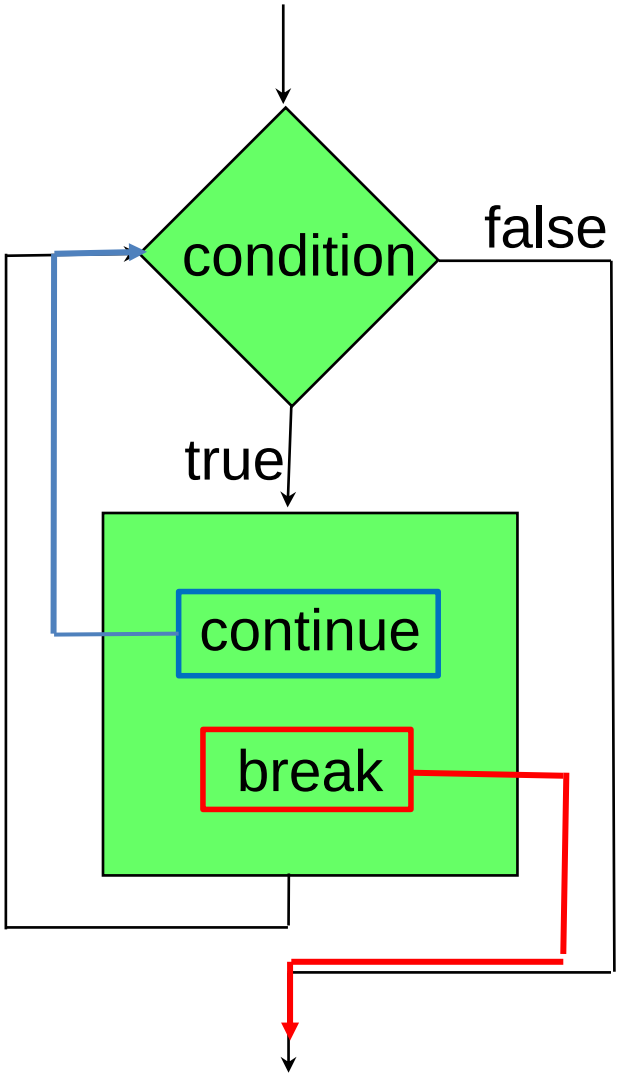
```java
import java.util.Scanner;

class CountDown
{
    public static void main(String[] args)
    {
        Scanner reader = new Scanner(System.in);
        int inputInt = reader.nextInt();
        while (inputInt != 0)
        {
            if (inputInt < 0 ){
                for (int i = inputInt; i < 0; i ++)
                {
                    System.out.println("Counter = " + i);
                }
            } else if (inputInt > 0){
                for (int i = inputInt; i > 0; i --)
                {
                    System.out.println("Counter = " + i);
                }
            }
            inputInt = reader.nextInt();
        }
    }
}
```

```java
import java.util.Scanner;

class CountDown2
{
    public static void main(String[] args)
    {
        Scanner reader = new Scanner(System.in);
        int inputInt;
        do
        {
            inputInt = reader.nextInt();
            if (inputInt < 0 ){
                for (int i = inputInt; i < 0; i ++)
                {
                    System.out.println("Counter = " + i);
                }
            } else if (inputInt > 0){
                for (int i = inputInt; i > 0; i --)
                {
                    System.out.println("Counter = " + i);
                }
            }
        }while (inputInt != 0)
    }
}
```

Οι εντολές break και continue

- continue: Επιστρέφει τη ροή του προγράμματος στον έλεγχο της συνθήκης σε ένα βρόγχο.
  - Βολικό για τον έλεγχο συνθηκών πριν ξεκινήσει η εκτέλεση του βρόγχου, ή για πρόορη επιστροφή στον έλεγχο της συνθήκης
- break: Μας βγάζει έξω από την εκτέλεση του βρόχου από οποιοδήποτε σημείο μέσα στον κώδικα.
  - Βολικό για να σταματάμε το βρόγχο όταν κάτι δεν πάει καλά.
- Κάποιοι θεωρούν οι εντολές αυτές χαλάνε το μοντέλο του δομημένου προγραμματισμού.

Οι εντολές break και continue



Παράδειγμα

```
while (…)
{
  if (everything is ok){

    < rest of code>

  }// end of if
} // end of while loop
```

```
while (… && !StopFlag)
{
  < some code >

  if (I should stop){
    StopFlag = true;
  }else{
    < some more code>
  }
} // end of while loop
```

```
while (…)
{
  if (I don't like something){
      continue;
  }

  < rest of code>

} // end of while loop
```

```
while (…)
{
  < some code>

  if (I should stop){
      break;
  }

  < some code>
} // end of while loop
```

```
import java.util.Scanner;

class CountDownWithContinue
{
    public static void main(String[] args)
    {
        Scanner reader = new Scanner(System.in);
        int inputInt = reader.nextInt();
        while (inputInt != 0)
        {
            if (inputInt%2 == 0){
                inputInt = reader.nextInt();
                continue;
            }
            if (inputInt < 0 ){
                for (int i = inputInt; i < 0; i ++)
                {
                    System.out.println("Counter = " + i);
                }
            } else if (inputInt > 0){
                for (int i = inputInt; i > 0; i --)
                {
                    System.out.println("Counter = " + i);
                }
            }
            inputInt = reader.nextInt();
        }
    }
}
```

Η αντίστροφη μέτρηση εκτελείται μόνο για περιττούς αριθμούς

```java
import java.util.Scanner;

class CountDown2
{
    public static void main(String[] args)
    {
        Scanner reader = new Scanner(System.in);
        int inputInt;
        do
        {
            inputInt = reader.nextInt();
            if (inputInt < 0 ){
                for (int i = inputInt; i < 0; i ++)
                {
                    System.out.println("Counter = " + i);
                }
            } else if (inputInt > 0){
                for (int i = inputInt; i > 0; i --)
                {
                    System.out.println("Counter = " + i);
                }
            }
        }while (inputInt != 0)
    }
}
```

```
import java.util.Scanner;

class CountDownWithBreak
{
    public static void main(String[] args)
    {
        Scanner reader = new Scanner(System.in);
         do
         {
             int inputInt = reader.nextInt();
             if (inputInt == 0){
                  break;
             }
             if (inputInt < 0 ){
                 for (int i = inputInt; i < 0; i ++)
                 {
                     System.out.println("Counter = " + i);
                  }
             } else if (inputInt > 0){
                 for (int i = inputInt; i > 0; i --)
                 {
                      System.out.println("Counter = " + i);
                  }
             }
         }while (true)
    }
}
```

Η συνθήκη αυτή ορίζει ένα ατέρμονο βρόγχο (infinite loop). Πρέπει μέσα στο πρόγραμμα να έχουμε μια εντολή break (ή return που θα δούμε αργότερα) για να μην κολλήσει το πρόγραμμα μας. Αυτή η κατασκευή είναι βολική όταν έχουμε πολλαπλές συνθήκες εξόδου.

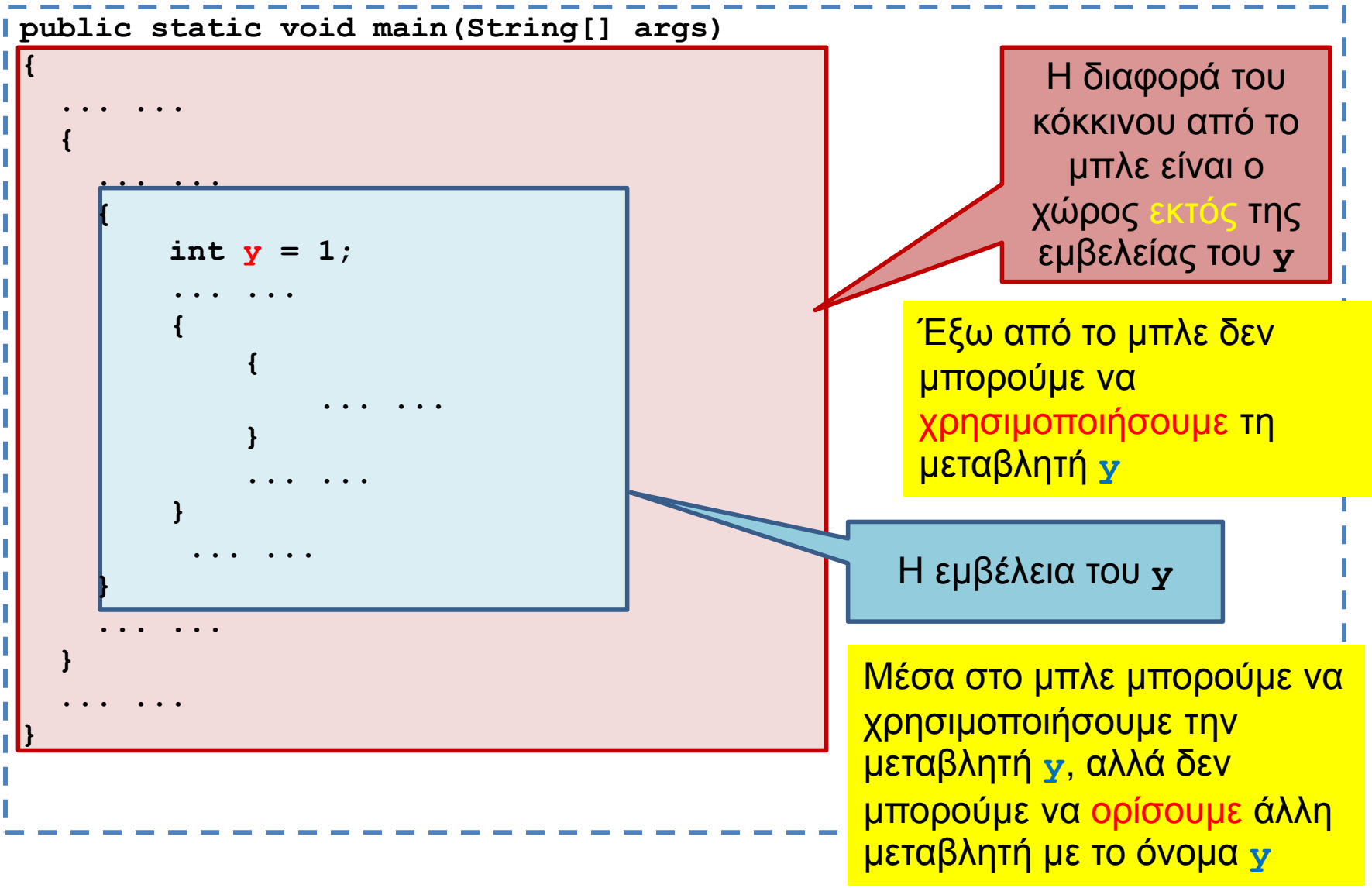
Εμβέλεια (scope) μεταβλητών

- Προσέξτε ότι η μεταβλητή `int i` πρέπει να οριστεί σε κάθε for, ενώ η `inputInt` πρεπει να οριστεί έξω από το while-loop αλλιώς ο compiler διαμαρτύρεται.
  - Προσπαθούμε να χρησιμοποιήσουμε μια μεταβλητή εκτός της εμβέλειας της
- Η κάθε μεταβλητή που ορίζουμε έχει εμβέλεια (scope) μέσα στο block το οποίο ορίζεται.
  - Τοπική μεταβλητή μέσα στο block.
- Μόλις βγούμε από το block η μεταβλητή χάνεται
  - Ο compiler δημιουργεί ένα χώρο στη μνήμη για το block το οποίο εκτελούμε, ο οποίος εξαφανίζεται όταν το block τελειώσει.
- Ένα block μπορεί να περιλαμβάνει κι άλλα φωλιασμένα blocks
  - Η μεταβλητή έχει εμβέλεια και μέσα στα φωλιασμένα blocks
  - Δεν μπορούμε να ορίσουμε μια άλλη μεταβλητή με το ίδιο όνομα σε ένα φωλιασμένο block

Παράδειγμα με το scope μεταβλητών

```
public static void main(String[] args)
{
        int y = 1;
        int x = 2;
        for (int i = 0; i < 3; i ++)
        {
            y = i;
            double x = i+1;
            int z = x+y;
            System.out.println("i = " + i);
            System.out.println("y = " + y);
            System.out.println("z = " + z);
        }
        System.out.println("i = " + i);
        System.out.println("z = " + z);
        System.out.println("y = " + y);
        System.out.println("x = " + x);
}
```

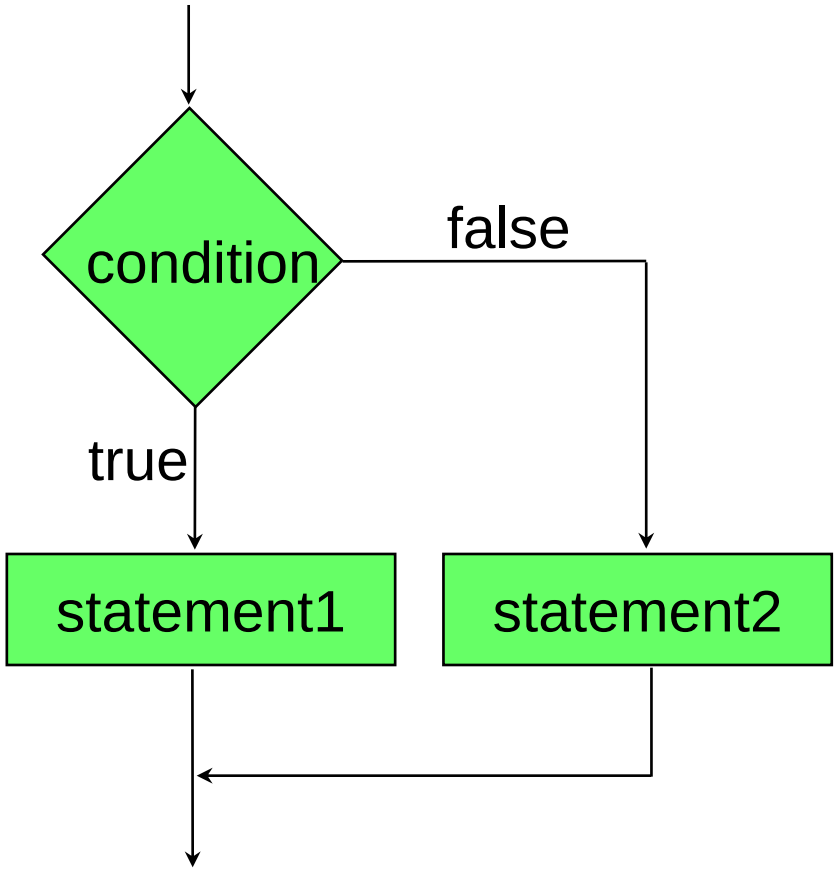
Ο κώδικας έχει λάθη σε τρία σημεία



Κάθε block έχει το δικό του χώρο μνήμης. Σε ένα χώρο μνήμης μια μεταβλητή μπορεί να οριστεί μόνο μία φορά. Ο χώρος μνήμης ενός block περιλαμβάνει και τα φωλιασμένα blocks.

Το if-else statement

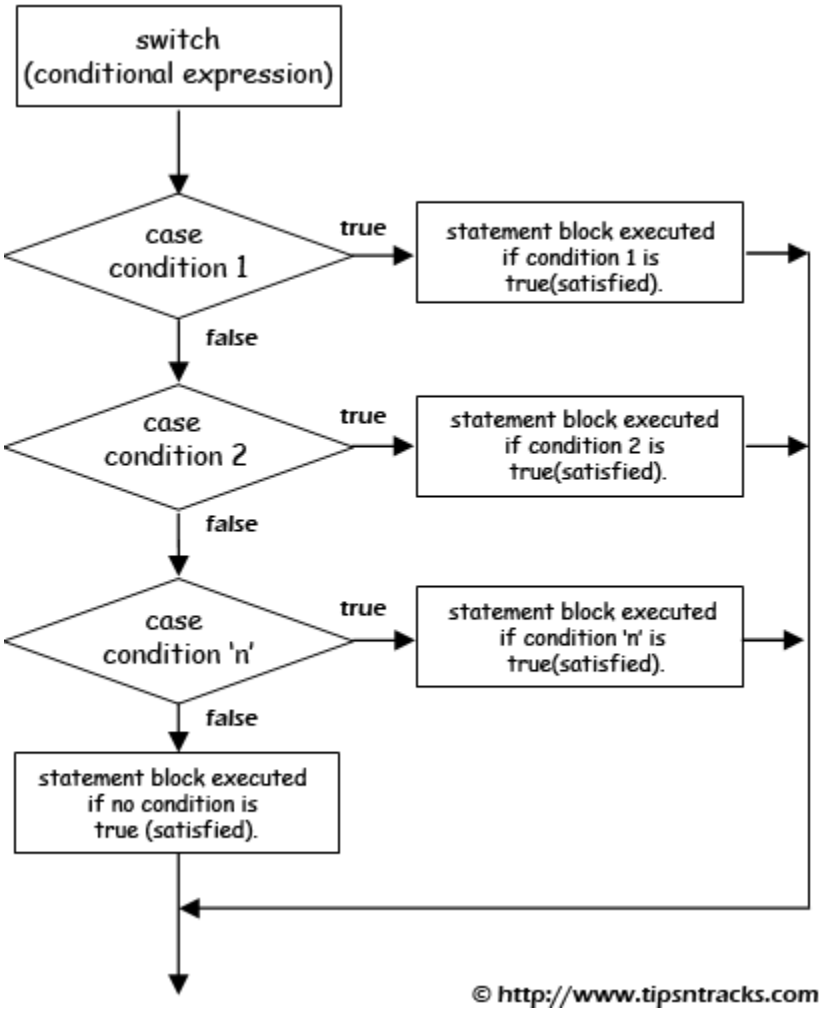
- Το if-else statement δουλεύει καλά όταν στο condition θέλουμε να περιγράψουμε μια επιλογή με δύο πιθανά ενδεχόμενα.
- Τι γίνεται αν η συνθήκη μας έχει πολλά ενδεχόμενα?



Switch statement

Συντακτικό:

```
switch (<condition expression>){
  case <condition 1>:
    code statements 1
    break;
  case <condition 2>:
    code statements 2
    break;
 case <condition 3>:
    code statements 3
    break;
  default:
    default statements
    break;
}
```



Ο κώδικας μέσα το switch είναι ένα μόνο block, και γι αυτό χρειαζόμαστε τα break

Παράδειγμα

- Ένα πρόγραμμα που να εύχεται καλημέρα σε τρεις διαφορετικές γλώσσες ανάλογα με την επιλογή του χρήστη.

```
import java.util.Scanner;

class SwitchTest{
  public static void main(String args[])
  {
    Scanner input = new Scanner(System.in);
    String option = input.next();

    switch(option){
      case "GR": // if (option.equals("GR")
        System.out.println("kalimera");
        break;
      case "EN": // if (option.equals("EN")
        System.out.println("good morning");
        break;
      case "FR": // if (option.equals("FR")
        System.out.println("bonjour");
        break;
      default:
        System.out.println("I do not speak this language.\n" +
                           "Greek, English, French only");
    }
  }
}
```

Αν θέλουμε να μπορούμε να απαντάμε και με μικρά?

```
import java.util.Scanner;

class SwitchTest2{
  public static void main(String args[])
  {
    Scanner input = new Scanner(System.in);
    String option = input.next();

    switch(option){
      case "GR":
      case "gr":
        System.out.println("kalimera");
        break;
      case "EN":
      case "en":
        System.out.println("good morning");
        break;
      case "FR":
      case "fr":
        System.out.println("bonjour");
        break;
      default:
        System.out.println("I do not speak this language.\n" +
                           "Greek, English, French only");
    }
  }
}
```

```
import java.util.Scanner;

class OtherSwitchTest
{
    public static void main(String args[])
    {
        Scanner input = new Scanner(System.in);
        System.out.println("Pick a curtain");
        int option = input.nextInt();
        switch (option-1)
        {
            case 0:
                System.out.println("You selected curtain 1. Zong!");
                break;
            case 1:
                System.out.println(
                        "You selected curtain 2. Congratulations!");
                break;
            case 2:
                System.out.println("You selected curtain 3. Zong!");
                break;
            default:
                System.out.println("Zong!");
        }
    }
}
```