ΤΕΧΝΙΚΕΣ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΟΥΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Εισαγωγή στη Java II

HelloWorld.java

```
class HelloWorld
{
    public static void main(String args[])
    {
        // print message
        System.out.println("Hello world!");
    }
}
```

- `javac HelloWorld.java`
- `java HelloWorld`

Χωρίς κανένα επίθεμα!

Division.java

```
class Division
{
  public static void main(String args[])
  {
    int enumerator = 32;
    int denominator = 10;
    double division;
    division = enumerator/(double)denominator;
    System.out.println("Result = " + division);
  }
}
```

- Ορισμός μεταβλητών
- Η Java είναι strongly typed γλώσσα: κάθε μεταβλητή θα πρέπει να έχει ένα τύπο.
- Οι τύποι int και double είναι πρωταρχικοί (βασικοί) τύποι (primitive types)
- Εκτός από τους βασικούς τύπους, όλοι οι άλλοι τύποι είναι κλάσεις

Έξοδος

- Για έξοδο μπορούμε να καλέσουμε τις συναρτήσεις του System.out αντικειμένου:
  - println(String s): για να τυπώσουμε ένα αλφαριθμητικό s και τον χαρακτήρα '\n'
  - print(String s): τυπώνει το s αλλά δεν αλλάζει γραμμή
  - printf: Formatted output
    - printf("%d",myInt); // τυπώνει ένα ακέραιο
    - printf("%f",myDouble); // τυπώνει ένα πραγματικό
    - printf("%.2f",myDouble); // τυπώνει ένα πραγματικό με δύο δεκαδικά

Είσοδος

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

Παράδειγμα

```
import java.util.Scanner;

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

Παράδειγμα

```
import java.util.Scanner;

class TestIO2
{
    public static void main(String args[])
    {
        Scanner input = new Scanner(System.in);
        double d= input.nextDouble();
        System.out.println("division by 4 = " + d/4);
        System.out.println("1+ (division by 4) = " +1+d/4);
        System.out.printf("1+ (division of %.2f by 4) = %.2f",d, 1+d/4);
    }
}
```

Το + λειτουργεί ως concatenation τελεστής μεταξύ Strings, άρα μετατρέπει τους αριθμούς σε Strings

Τι θα τυπώσει αυτό το πρόγραμμα?

Βρόγχοι – Το if-then-else Statement

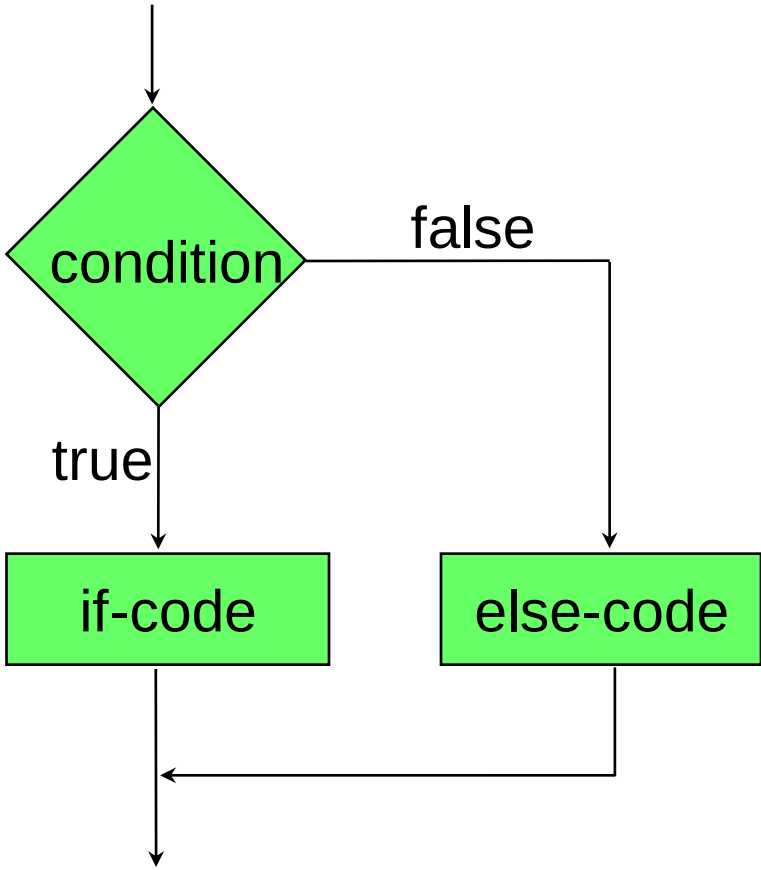
- Στην Java το if-then-else statement έχει το εξής συντακτικό

```
if (condition){
     …if-code block…
}else{
     …else-code block…
}
```

  - Αν η συνθήκη είναι αληθής τότε εκτελείται το block κώδικα if-code
  - Αν η συνθήκη είναι ψευδής τότε εκτελείται το block κώδικα else-code.

- Ο κώδικας του if-code block ή του else-code block μπορεί να περιέχουν ένα άλλο (φωλιασμένο (nested)) if statement

- Προσοχή: ένα `else` clause ταιριάζεται με το τελευταίο ελεύθερο `if` ακόμη κι αν η στοίχιση του κώδικα υπονοεί διαφορετικά.

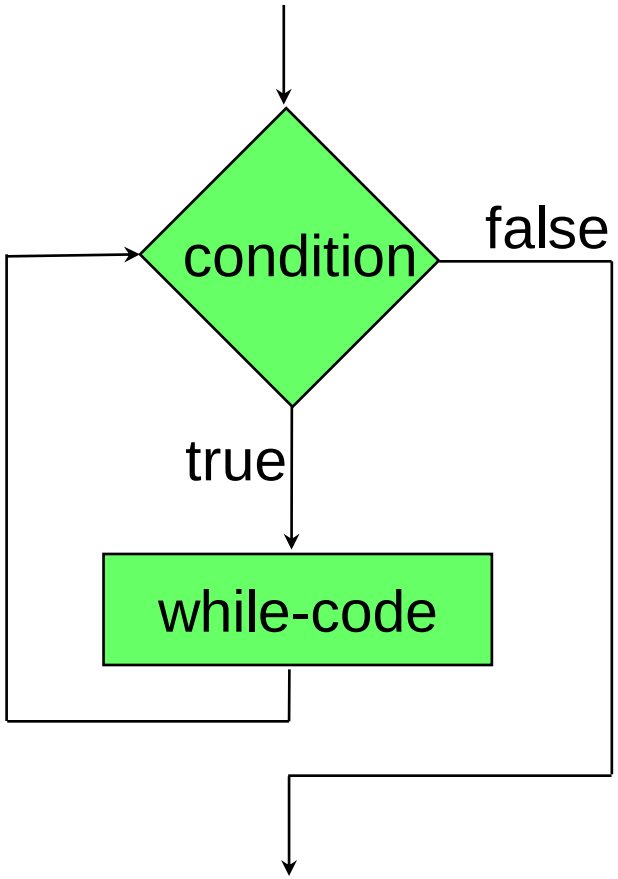


Επαναλήψεις - While statement

- 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.

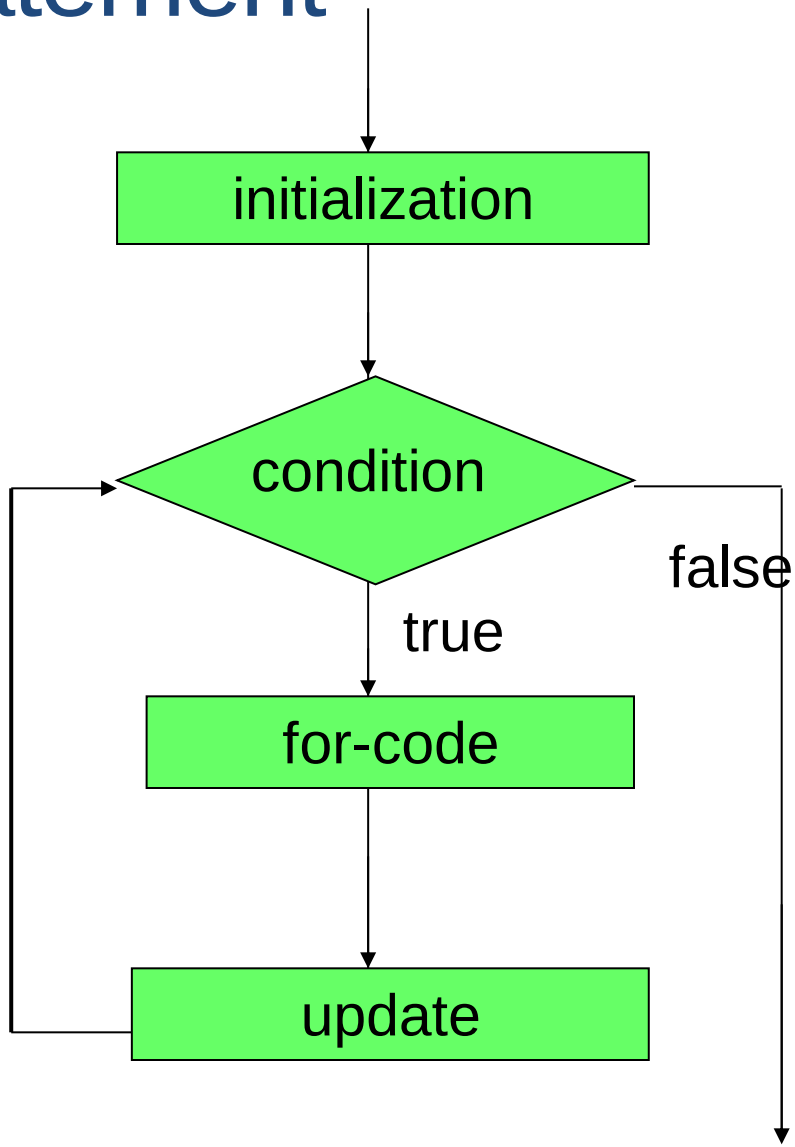


Επαναλήψεις – for statement

- 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.

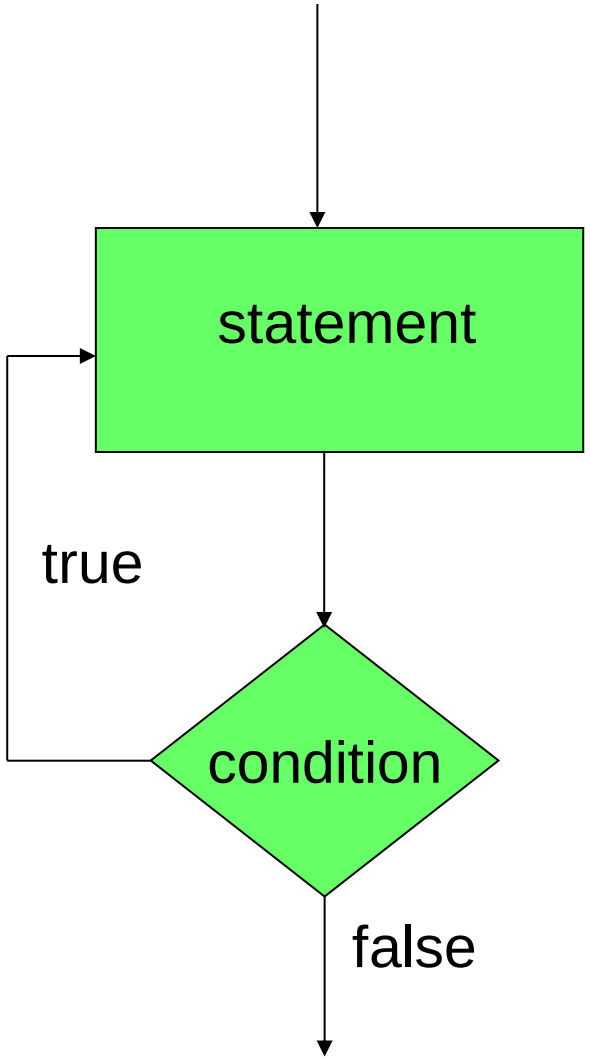


Το Do-While statement

- 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.

```
import java.util.Scanner;

class FlowTest
{
    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 FlowTest2
{
    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
  - Π.χ., πώς θα τυπώναμε μόνο τους άρτιους αριθμούς?

- break: Μας βγάζει έξω από την εκτέλεση του βρόχου από οποιοδήποτε σημείο μέσα στον κώδικα.
  - Κάποιοι θεωρούν ότι χαλάει το μοντέλο του δομημένου προγραμματισμού.
  - Βολικό για να σταματάμε το βρόγχο όταν κάτι δεν πάει καλά.

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 FlowTestContinue
{
    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();
        }
    }
}
```

Τυπώνει μόνο τους περιττούς αριθμούς

```
import java.util.Scanner;

class FlowTest2
{
    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)
    }
}
```

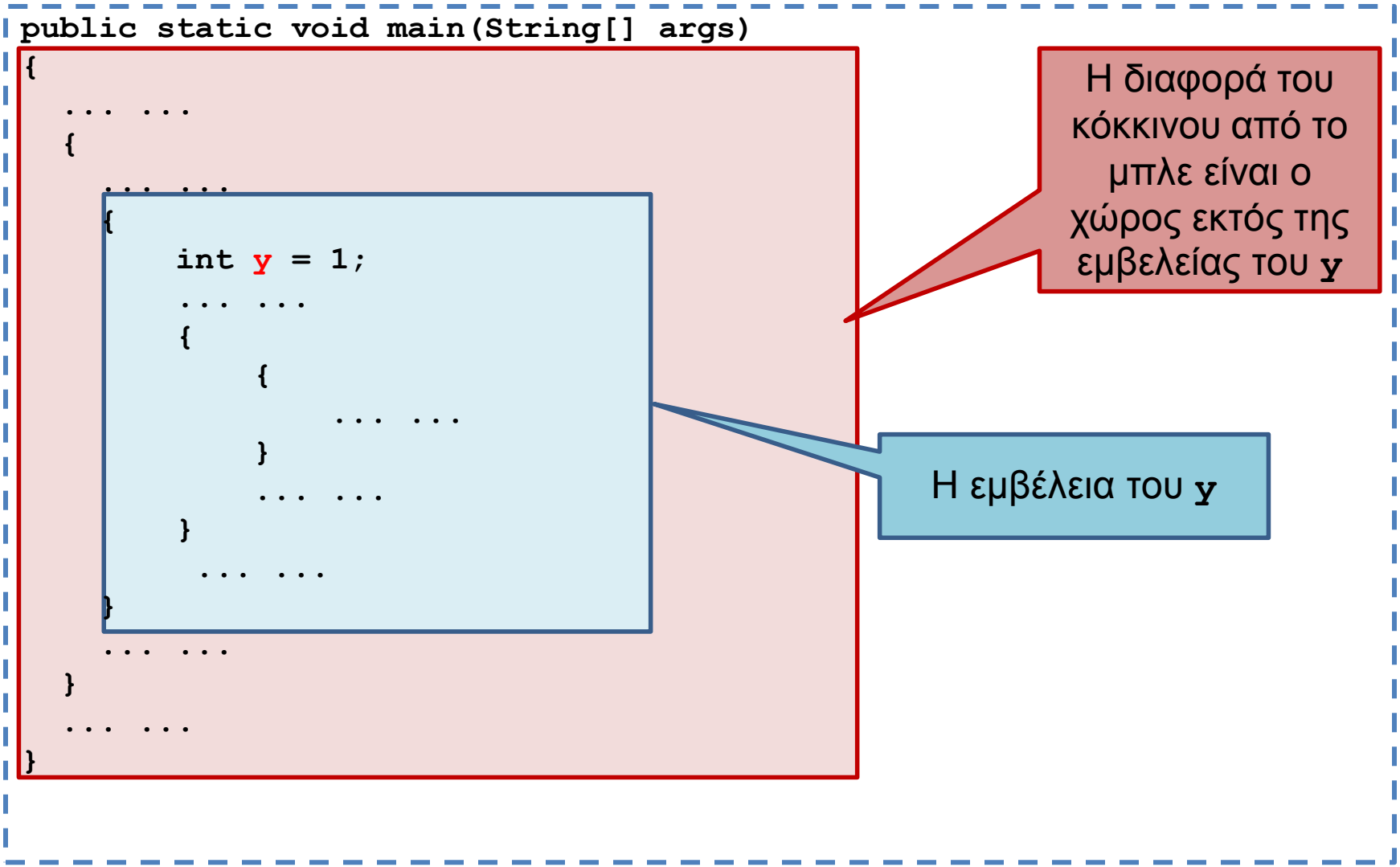
Scope μεταβλητών

- 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 στο stack ένα χώρο για το block το οποίο μετά 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;
            int 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


public static void main(String[] args)
{
... ...
{
... ...
{
int y = 1;
... ...
{
{
... ...
}
... ...
}
... ...
}
... ...
}
... ...
}
Η διαφορά του κόκκινου από το μπλε είναι ο χώρος εκτός της εμβελείας του y
Η εμβέλεια του y

Strings

- Η κλάση String είναι προκαθορισμένη κλάση της Java που μας επιτρέπει να χειριζόμαστε αλφαριθμητικά.
- Ο τελεστής “+” μας επιτρέπει την συνένωση
- Υπάρχουν πολλές χρήσιμες μέθοδοι της κλάσης String.
  - length(): μήκος του String
  - equals(String x): τσεκάρει για ισότητα
  - trim(): αφαιρεί κενά στην αρχή και το τέλος του string.
  - split(char delim): σπάει το string σε πίνακα από strings με βάσει το χαρακτήρα delim.
  - Μέθοδοι για να βρεθεί ένα υπο-string μέσα σε ένα string.
  - Κλπ.

Παράδειγμα με Strings

```
public class StringProcessingDemo
{
    public static void main(String[] args)
    {
        String sentence = "I hate text processing!";
        int position = sentence.indexOf("hate");
        String ending =
                sentence.substring(position + "hate".length( ));

        System.out.println("012345678901234567890123");
        System.out.println(sentence);
        System.out.println("The word \"hate\" starts at index "
                                                + position);

        sentence = sentence.substring(0, position) + "love"
                                                + ending;
        System.out.println("The changed string is:");
        System.out.println(sentence);
    }
}
```

Τα Strings είναι αμετάβλητα (immutable) αντικείμενα
Όταν κάνουμε ανάθεση δημιουργούνται και αντιγράφονται από την αρχή