

e.g.:

```
String[] clothes = {"gloves", "shoes", "shirts"};  
for (String i : clothes) {  
    System.out.println(i);  
}
```

Note: for "for" loops we are usually using i, j, k, l, m, n, \dots (i_1, i_2, i_3, \dots) as a loop variable

for a "for each" loop we usually use an easily understood name

e.g. `for(String line : book)`

`for(String fruit : fruits)`

`for(Integer i : numbers)`

Homework: are \forall for \Leftrightarrow for each?
(harder exercise)

It is easy to prove that by using
 \exists for each we can get \forall for,
however by using \forall for and
getting \exists for each is harder.

Nested loops:

```
for (int i=0; i<4 ; i= i+1) {
```

```
    for ( int j=0; j<4 ; j=j+1) {
```

.

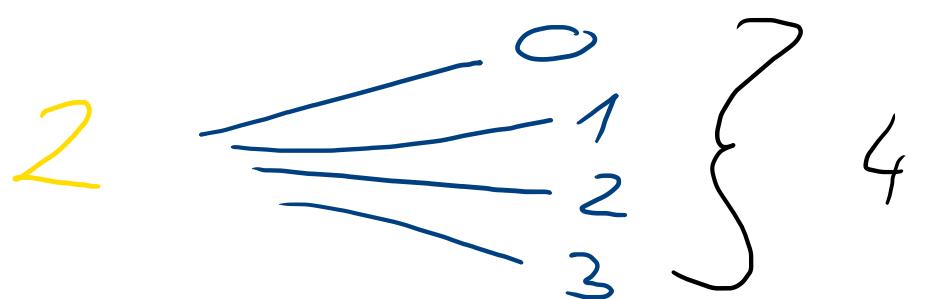
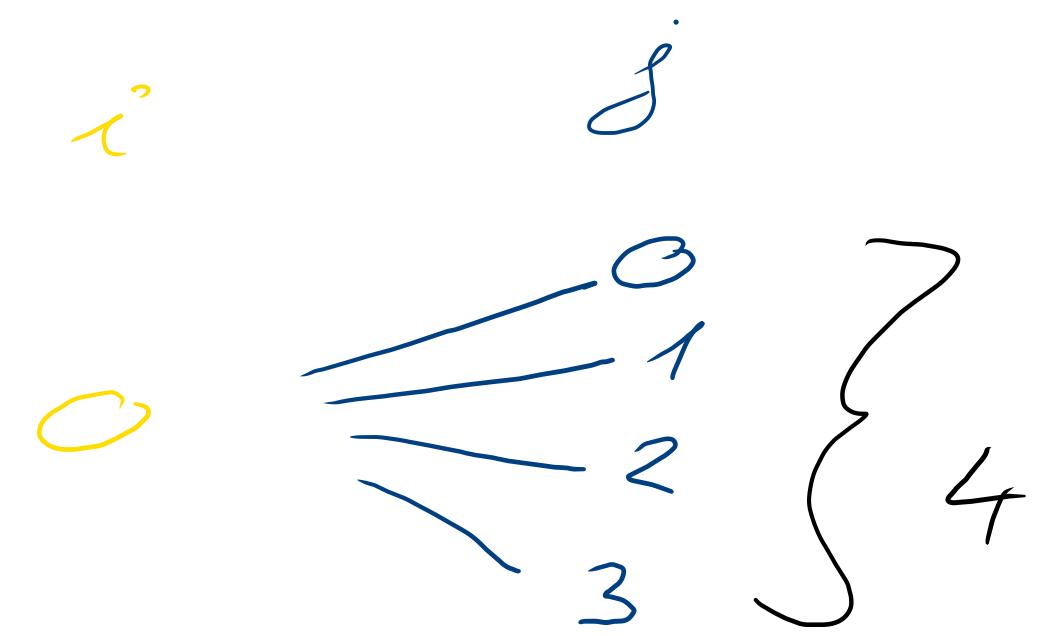
.

.

```
}
```

```
{
```

how many iterations do we have
in total?



For \Leftrightarrow For each equivalence

FOR EACH with FOR only

for $i=0$ to array.length by 1 do

use $array[i]$ in the code

(in a for each we use i)

end for

FOR with FOR EACH

$i = 0$

for each element in array do

work with i

⋮

$i = i + 1$

end for each

Until now:

- 1) if - else (else if) statements
- 2) loops

Now: functions

$$\hookrightarrow g(x) = x^2 + 2x - 3$$

\hookrightarrow in IT they are similar,
but different as well.

Math. functions:

domain, range

$$f: \mathbb{N} \rightarrow \mathbb{R}^+$$

dom \rightarrow range

$$f(x) = \sqrt{x}$$

def.

In programming every func. has a head(er) and a body.

In statically typed languages:

* head: ~~access~~ ~~extra info~~ ~~returns~~ ~~name~~ ~~parameters~~
modifiers extra info returns type name parameters

(programming 1,2)

↳ return type: type of the data which we get back after the function finishes.

It can be int, double, String, ...

(if we do not want to get anything then the type is "void")

↳ name can be "anything" (remember to the naming conventions at the start)

↳ isPrime

↳ addXY

↳ calculateNumberOfDays

⋮

↳ no á,é,í,ó,ö,--

↳ no "space"

↳ parameters:

- a) always given inside a bracket: (...)
- b) separated by "," : (par1, par2, par3, ...)
- c) if we have 0 parameters we STILL put an empty bracket ()
- d) every param. has 2 parts: type name
(see before at variables: int x)

⇒ a param. is a data what the function gets from the main program

- body: actual description of the function (so the steps what it performs)

Compared to math:

$$f : N \rightarrow R^+$$

$$f(x) = \sqrt{x}$$

N : type of the param.

R^+ : type of the return value

x : name of the param.

f : name of the function

\sqrt{x} : body (return value)

Java function from

$$f: N \rightarrow R^+$$

$$f(x) = \sqrt{x}$$

(~~unsigned~~) double ~~f (unsigned int x)~~ {
 return ~~root(x);~~

}

Problem: in programming we usually work with integers instead of positive integers.

So to calculate the square root we need to make sure that our number is \oplus or 0 .

```
double f(int x){  
    if (x < 0) {  
        print("Error, number is negative!");  
        return -1;  
    } else {  
        return root(x);  
    }  
}
```

What can we do in the "body" part?

↳ the body is basically an isolated program, so we can use any programming tools (if-else, variables, loops, functions, etc.)

BUT: • Since it is isolated, it cannot "see" the data defined elsewhere.

(e.g. if in our main program we have
int num=1234 then we cannot access
it from other functions)

- also because of the isolation almost everything in the function gets deleted after it ends.

Functions are the "Las Vegas" of programming.

("everything happens in the functions stay in the functions")

Communicating with the "outside" program :



How to give back a value:

return keyword

↳ return variable name

↳ return result

↳ return x

⋮

↳ we can only return a variable
which has the same type as
the one what we gave in the head

↳ we can only return 1 value
in C, C++, Java, C#, ...

(however in Python, LUA, ... we can
return more than 1 values as well)

↳ in Python: return $\textcircled{1} \textcircled{2} \textcircled{3}$
 x, y, num

↳ after the function arrives to a
return statement, it immediately
stops.

↳ we can have multiple return statements : if sun is shining then
return walk
else
return gaming
end if

Examples:

```
public static double addTwoNumbers(double x1, double x2){  
    double result = x1 + x2;  
    return result;  
} I used oo return → after function and I'm  
//MAIN PROGRAM BELOW forgetting the result var.  
public static void main(String[] args) {  
    double x1 = 10.5;  
    double x2 = 9.3;  
    addTwoNumbers(x1,x2);  
    System.out.println(result);  
  
elvezek: build failed At 2021-03-18 19:07 with 1 error  
src\com\company 1 error  
ot find symbol variable result :15  
3 sec, 743 ms  
D:\Dropbox\Matek\Tanítás\2020\Programm  
java: cannot find symbol  
symbol: variable result  
location: class com.company.Main
```

How to understand this "return"
Statement?

- Hey brother, can you download a movie
for me?
- Yes.



- So why didn't you download it?
- You only asked me if I could download it
not to actually download.

If I simply run a function, the return value just simply gets calculated and then "forgotten".

If I want to keep it, I need to assign it to a variable!

"Please calculate 5×13 !"

Done.

```
public static double addTwoNumbers(double x1, double x2){  
    double result = x1 + x2;  
    return result;  
}
```

//MAIN PROGRAM BELOW

```
public static void main(String[] args) {  
    double x1 = 10.5;  
    double x2 = 9.3;  
    double result = addTwoNumbers(x1,x2);  
    System.out.println(result);
```

Main X

```
"C:\Program Files\Java\jdk-15.0.1\bin\java.exe" "-javaagent:C:\Pro  
19.8
```

```
public static void subtractTwoNumbers(double y1, double y2){  
    double newResult = y1 - y2;  
    // return newResult; -> since the return type is void  
    // I cannot return a variable  
}  
  
//MAIN PROGRAM BELOW  
public static void main(String[] args) {  
    double y1 = 10.5;  
    double y2 = 9.3;  
  
    // since the subtractTwoNumbers function is a void func.  
    // it is not possible to access newResult from the outside  
}
```

```
public static double addTwoNumbers(double x1, double x2){  
    double result = x1 + x2;  
    return result;  
}  
  
// having same var names in the parameters or inside the function  
// of multiple functions is possible, since they are isolated  
// from each other  
public static double subtractTwoNumbers(double x1,double x2){  
    double result = x1 - x2;  
    return result;  
}
```

```
public static double addTwoNumbers(double x1, double x2){  
    double result = x1 + x2; 10.5 9.3  
    return result; 10.5 9.3
```

}

↳ just a name what I only
use is the function (internal use
only)

//MAIN PROGRAM BELOW

```
public static void main(String[] args) {  
    double y1 = 10.5; → y1 means 10.5  
    double y2 = 9.3; → y2 means 9.3
```

// will this one work? YES! 10.5 9.3

```
    double number = addTwoNumbers(y1, y2);  
    System.out.println(number);  
}
```

x_1 , x_2 , $result$ are just "symbols"
which we substitute with numbers
in the program.