Test assignments
1. make two assignments, one of them must be boolean
2. make a while loop with AND and OR and equal logical statements.
3. Make a while loop with a boolean variable as expression.
4. define a function and call a function
5. make a void function with and without return
6. make a if else in a if else.

how to start a program
Every program must have an setup and loop function declared.

```
func setup(){
        statements
}
func loop(){
        statements
}
```

outside of loop and setup function definitions are made

1.
variable declarations and assignment:
a variable is a value that is saved with another name.
When you first tell that an value exist it must be "declared" so it later can get an value, which is called a assignment.
The right side of an assignemt can be a expression or a function call
example:

```
int someNumber              #declaration
someNumber := 1             #assignment
sumeNumber := 2/2           #expression
someNumber := returnTwo()   #function call
```

statements:
- variable assignments
- function calls
- while loops
- if else statements

code: all text that is programmed is called code.

a value can be a:
- int (which is whole a number ex. 1)
- double (which is a floating point number ex. 1.1)
- string (text: "hello")

boolean (data type for logic, it has the values true and false and can handle
expressions that compare to each other
ex.
a:= true
a:= false
a:= 1<2
)

2.
while loop:
a while loop repeats the code inside it until the condition no longer is met.
example:
int a
a := 0
while(a<3) {
  a = a - 1
}

3.
AND OR EQUAL
AND, OR and equal “written =” are used for logic
both arguments have to be true for an and statement to be true
ex.
1<2 AND 2<3
one of them must be true for OR
ex.
1<2 OR 2<1
and equal returns true if the two compared values are the same
ex.
1 = 1
example with a variable:
int a
int b
boolean c
a:= 1
b:=1
d := 2
c := a = b (evaluates to c := true)
c := 2 = 2 AND 1 = 1 (will be true)
c := 2 = 2 OR 2 = 1 (will be true)

4.
function definition and function call:
A function executes the code inside {} brackets when called.
A function can have a type. If it have a type it must return a value.
example:

function definition with type:

```
func int returnANumber(){
        return 1
}
```

function call:

runs the statements inside the function definition and might return a value

```
helloWorld()
returnOne()             #might return the value 1 depending on the code inside it
```

example

```
int a
a := returnOne()
```

5.

function definition without type (also called void):

```
func helloWorld(){
        string hello := "hello world"
}
```

if you want to end a function before it has run all code it can use return without a value.

ex.

```
func int returnANumber(){
        if(error){
                return
        }
        int a
        a:= 5 + 13
        return a
}
```

(stops after repeating 3 times since a = 0)

6.

if else statements:

if decides if some code will be reaches if what inside them is true. If it is not true else will be executed instead.

ex where the if statement is true and else will not be reached.

```
if(1<2){
        int a
        a := 5
} else{
        int a
        a := 1
}
```