Lesson 55 (search)

In this lesson we want to write a program that calculates the number of digits in a given number. For example 5896 contains 4, 52 contains 2.

What's the difference between **/** and **%** ?

Division mark /: 85 **/** 10 = 8.5. But 85 is declared as an int, so the result is going to be 8, 429 **/** 10 the result is going to be 42.

Remainder of the division % :

4123 **%** 10 = 3

412 **%** 10 = 2

10 **%** 4 = 4

```
#include <stdio.h>
int main() {
int x,r=0;
printf("Enter number: ");
scanf("%d", &x);
while (x) {
x /= 10;
r++;
}
printf("%d", r);
}
```

input:

```
369852741
```

output:

```
9
```

Try the code : Click here!

This program works only one time, if we want to change that:

Either we write **while**(1) { }.

Either we use the **goto**;

```
goto label ;
..
..
label :
statements;
```

The label is an identifier. When the goto statement is encountered, the control of the program jumps to label: and starts executing the code.

```
#include <stdio.h>
int main() {
int x,r;
start:
```

```
printf("Enter number: ");
scanf("%d", &x);
r=0;
//start counting from 0
while (x) {
//Because we don't know the number of digits
x /= 10;
r++;
}
printf("%d", r);
goto start;
}
```

Each time the computer executes the print statement, he will find the **goto**, so he returns to the code after the label **start**.

But here in this program there is an error that some people fall into, which is the definition of r = 0 above start. Here, in each cycle, the program collects the old value in the old roll with the new value. But here, for each cycle of the **goto** command, we need r to be zero.

There is a note for the **goto** command to work correctly on any C compiler

It must be under the variables definition

Because the compiler in the video is for **C++**