Exercises on Conditionals and Logical Operators

The tea party

We are having a party with amounts of tea and candy.

- A party is good if both tea and candy are at least 5
- However, if either tea or candy is at least double the amount of the other one, the party is great.
- However, in all cases, if either tea or candy is less than 5, the party is always bad.

Write a program which prints the state of a party to *stdout.* Start with the following variable definitions. Try to change the numbers and check if your program behaves correctly then.

```
int tea = 5;
int candy = 2;
```

Sum limit

Given 2 non-negative ints, a and b, print their sum, so long as the sum has the same number of digits as a. If the sum has more digits than a, just print a.

```
int a = 5;                           int a = 5;
int b = 2;                           int b = 8;
→ "7"                                → "5"
```

Exercises on Loops

Print birthday and age

For each of your birthdays print the the date and how old you turned at that day. For example the output could look like this:

```
→ "03.03.1996: Johannes turns 1"
→ "03.03.1997: Johannes turns 2"
...
```

Odd sum

Sum up all odd numbers from 0 to 100 excluding the number 13.

DoubleChar

Given a string, print a string where for every char in the original, there are two chars.

```
String doubleThat = "The";
→ "TThhee"
```

Exercises on Data Types

Cross-Assigning

Declare and initialize (that is, to assign a value to it) a variable of each kind (`int, float, double, long, short, byte, char, boolean`). Now try to save each variable in a variable of a different kind, e.g. for `int` to `long`:

```
int i = 5;
long l = i;
```

If it does not work, try to *cast* it:

```
long l = 14;
int i = (int) l;
```

Draw a table in which, for each pair of data types, you note if the assignment works right away (like from `int` to `long`), if it works only with casting (like from `long` to `int`) or if it does not work at all.

How big is int?

Remember how we printed all the powers of 2 from 1 to 128?:

```
1, 2, 4, 8, 16, 32, 64, 128,
```

Right.
Now, do the same, but do not only print 8 numbers. Instead, print *50* such numbers beginning at 1. What do you see? Do all of the numbers really always become bigger? What happens? Can you find an explanation on the internet?

How big is long?

Do the previous example again, but this time, instead of using `int`, use `long` as the data type for the variable. What can you see? Try to print more than 50 numbers.