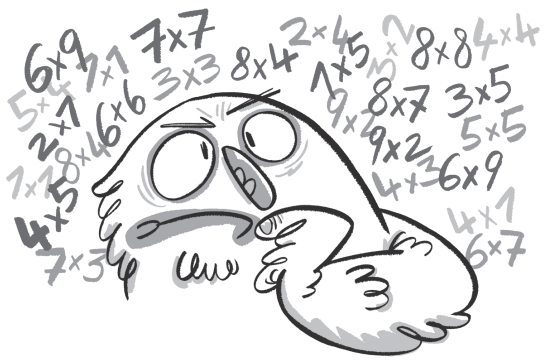
The Calliope mini as multiplication tables trainer

Lio is practicing for the upcoming math exam
Lios math teacher has announced a class test for next week. It is going to be about multiplication. Lio has already solved all tasks in the exercise book. But that is not enough. Lio wants to write a program that keeps creating new exercises.



The calculator

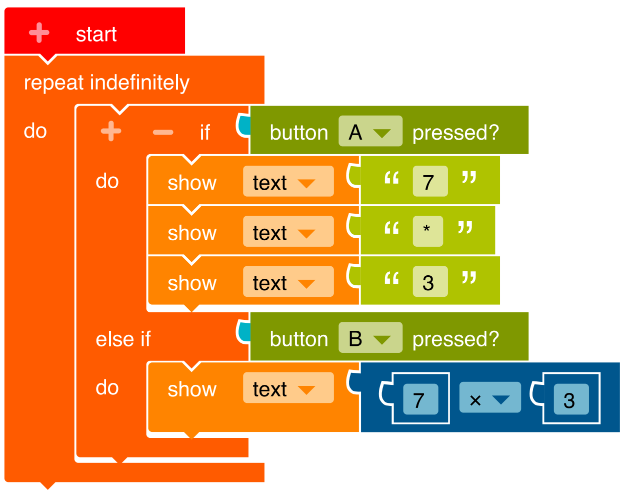
Computers calculate the most difficult math tasks within the blink of an eye. What a computer shall calculate must first be programmed. If the computer is given the correct program, it can even create math tasks for people.

The code

The program shows a single calculation task.

Describe what the Calliope mini generates if you press button A or B.
Go over the program commands carefully.

If button A is pressed, then ...



```
start
repeat indefinitely
do  if   button A pressed?
    do   show text "7"
         show text "*"
         show text "3"
    else if button B pressed?
    do   show text 7 × 3
```

1. a) Program this code in the NEPO® editor ▢ .
   b) Change the numbers to generate a new multiplication task.
   c) SIM Open and ▷ start the simulator. Try the program.
   d) Write down what the Calliope mini displays:

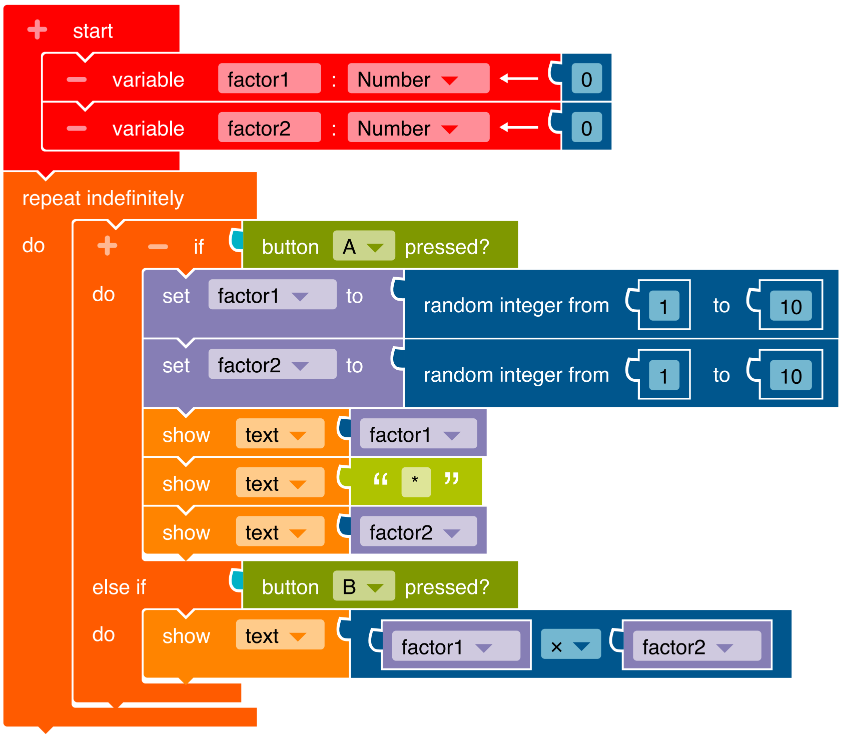
   if button A is pressed: ______________________

   if button B is pressed: ______________________

2. The program shown on page 28 always generates the same calculation task.

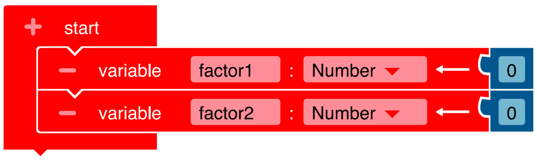
Now create a program that does the following :
If button A is pressed, generate a multiplication with random numbers. If button B is pressed, display the result of the task.

This is the code of the finished program:



```
+ start
  – variable factor1 : Number ← 0
  – variable factor2 : Number ← 0
repeat indefinitely
do
  + – if button A pressed?
  do
    set factor1 to random integer from 1 to 10
    set factor2 to random integer from 1 to 10
    show text factor1
    show text " * "
    show text factor2
  else if button B pressed?
  do
    show text factor1 × factor2
```

Program this code in the NEPO® editor ☆ .
Proceed step by step.



```
+ start
  – variable factor1 : Number ← 0
  – variable factor2 : Number ← 0
```

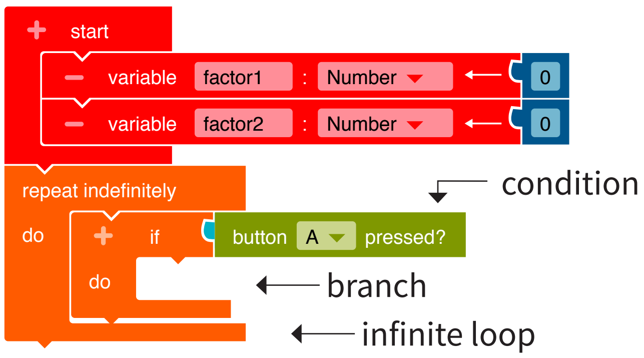
• In order to generate new tasks again and again, two variables* must be created in the beginning. Click on the "+" next to "start". Click on the word "item" and type in the new variable name "factor1".
Repeat both steps.
Give the new variable the name "factor2".

- To create multiplication tasks as many times as you want, you need an infinite loop*.

  Control ▶ Loops ➞ "Repeat indefinitely/do"

  If button A is pressed (if), a multiplaction task shall be displayed (do):
  For this you need a branch*.

  Control ▶ Decisions ➞ "if/do"
  Sensors ➞ "button A pressed?"

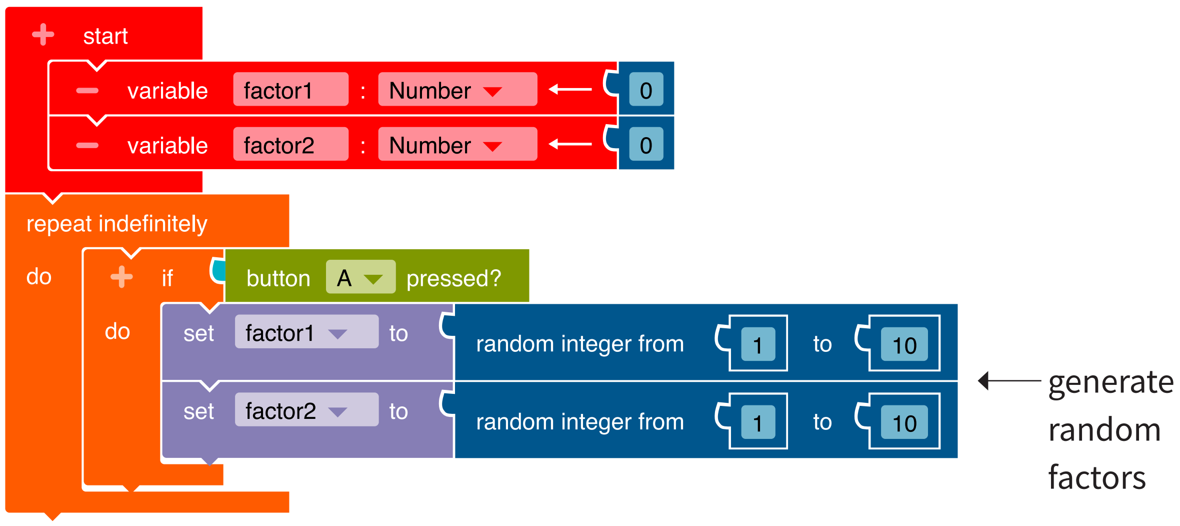


- In order to create the factors for the task, random numbers must be generated. The numbers are stored in the variables "factor1" and "factor2".

  Variables ➞ "set factor1 to"
  Add the block to the branch.

  Math ➞ "random integer from 1 to 100"
  Add this block and change the value 100 to the number 10.
  By doing so, you determine that the random number cannot be greater than 10.

  Repeat the two steps for the variable"factor2".


start
variable factor1 : Number 0
variable factor2 : Number 0
repeat indefinitely
do
if
button A pressed?
do
set factor1 to random integer from 1 to 10
set factor2 to random integer from 1 to 10
generate random factors

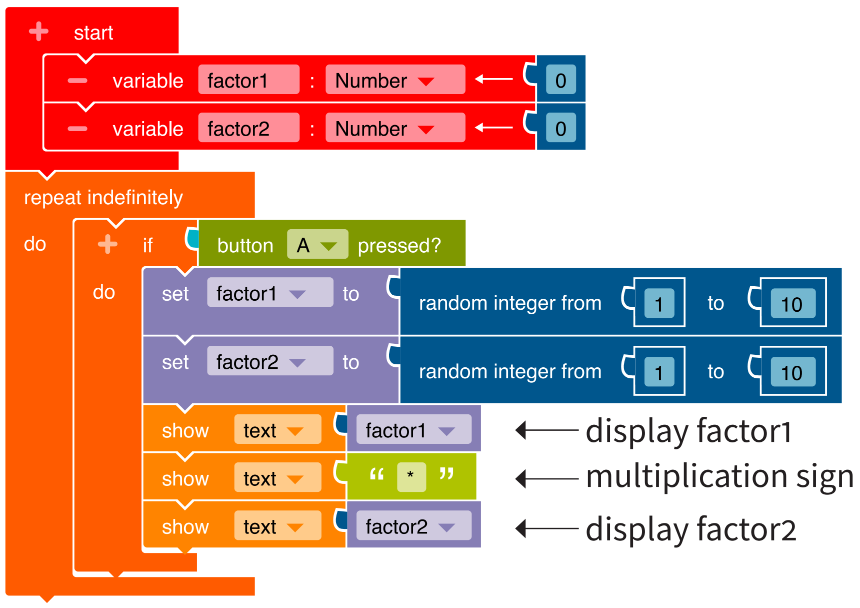
- The Calliope mini should display the task on the LED screen*
  Program the output of the first factor.

  Action ▶ Display → "show text"
  Delete the block "Hello".
  Variables → "factor1"
  Attach the block to the "show text" block.

SIM Open and ▷ start the simulator. Try the programm.

- In the next step, the multiplication sign should be displayed.

  Action ▶ Display → "show text"
  Click on the word "Hello" and type in an asterisk (*) as a multiplication sign.



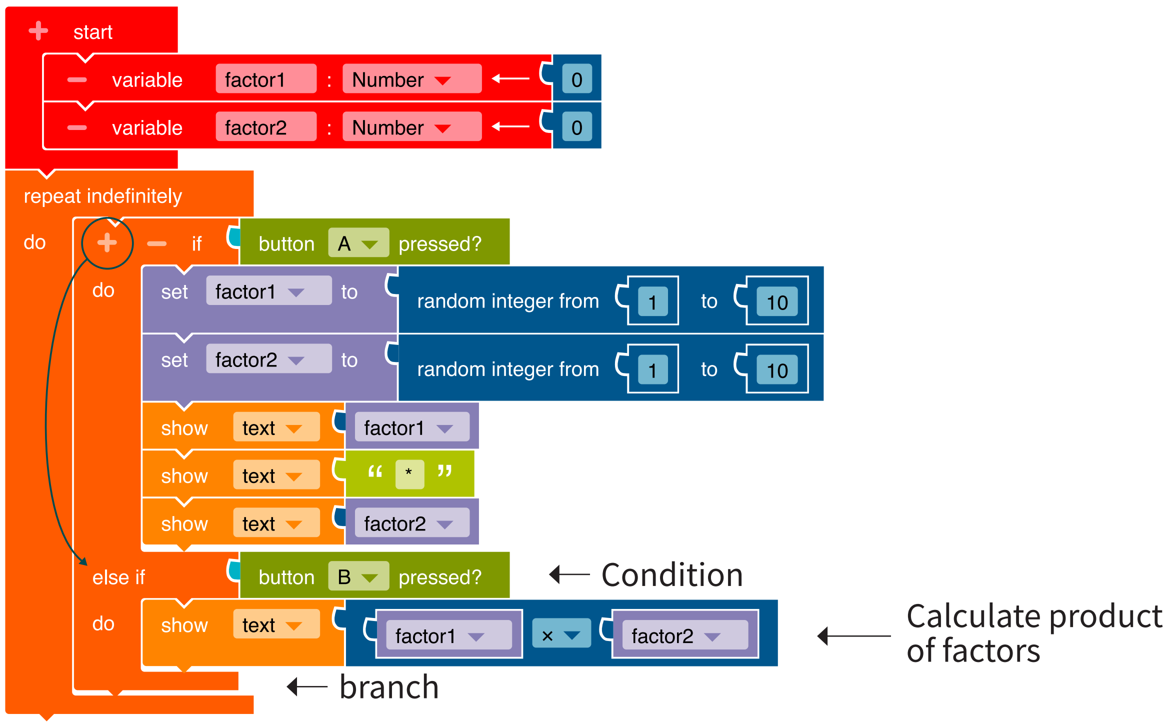
```
start
    variable factor1 : Number ← 0
    variable factor2 : Number ← 0
repeat indefinitely
do  if button A pressed?
    do  set factor1 to random integer from 1 to 10
        set factor2 to random integer from 1 to 10
        show text factor1        ← display factor1
        show text "*"            ← multiplication sign
        show text factor2        ← display factor2
```

- Now program the output of the second factor.
  Proceed as you did for the first factor. But use the variable "factor2".

SIM Open and ▷ start the simulator. Try the program.

- If button B is pressed (if), the result of the multiplication task should be displayed (do). Click on the "+" next to "if". Another branch appears.

Sensors ➞ "button A pressed?"
Add the block as a condition to the branch.
Click "A" and select "B".



```
+ start
    − variable factor1 : Number ← 0
    − variable factor2 : Number ← 0
repeat indefinitely
do  + − if  button A pressed?
        do  set factor1 to  random integer from 1 to 10
            set factor2 to  random integer from 1 to 10
            show text  factor1
            show text  " * "
            show text  factor2
        else if  button B pressed?
        do  show text  factor1 × factor2
```

- The Calliope mini calculates the product of the two randomly selected factors and displays it on the LED screen.

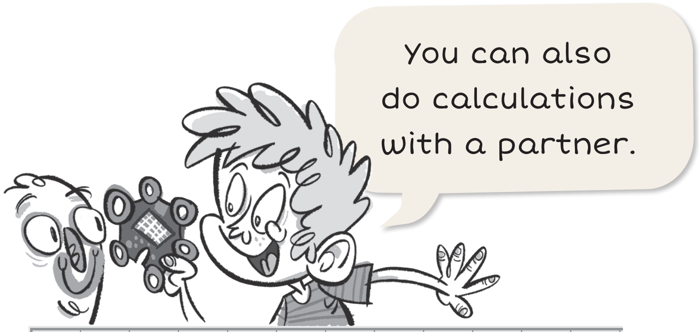
Action ▶ Display ➞ "show text"
Add the block to the second branch. Delete the block "Hello".

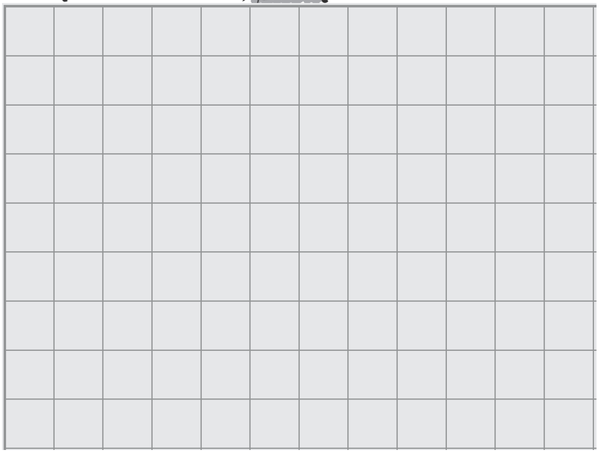
Math ➞ " "
Change the arithmetic character to a multiplication character by clicking the "+" and selecting the "x" in the drop-down menu.

Add the variables:
Variable ➞ "factor1" and "factor2"
Insert the blocks to the left and right of the "x" sign.

3. a) ▶ Transfer the code to the Calliope mini and run the program.

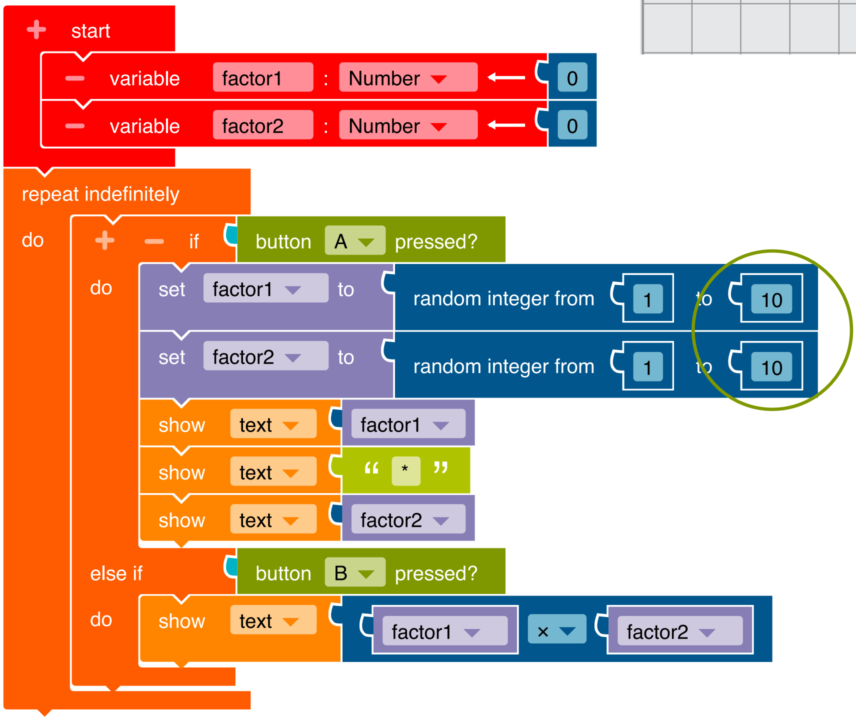
b) Press button A on the Calliope mini to get a new task. Solve the task. Check your solution by pressing button B.





4. You can change the code by following these steps:

a) Use the Calliope mini to calculate tasks of the great multiplication table. The green circle shows you, where you need to do changes.



```
start
    variable factor1 : Number ← 0
    variable factor2 : Number ← 0
repeat indefinitely
do  if button A pressed?
    do  set factor1 to random integer from 1 to 10
        set factor2 to random integer from 1 to 10
        show text factor1
        show text "*"
        show text factor2
    else if button B pressed?
    do  show text factor1 × factor2
```

b) Calculate addition problems with the Calliope mini.
Mark the blocks in red where you need to change something.

c) SIM Open and ▷ start the simulator. Try the programm.

5. ▶ Transfer the changed code to the Calliope mini and run the program.

The small coding encyclopedia

instruction
(= command)

When you receive an instruction, you can execute it. For example:
"Hang the wet socks on the clothes horse to dry."
The same is true for the computer. It executes instructions that clearly describe what it should do. A code/program is built from instructions.

loop
with a
condition

A loop allows a sequence of instructions to be executed over and over again.
For example:
"Hang up socks as long as there‘s laundry in the basket."
The **loop** is: "Hang up socks as long as (repeat) …"
The **condition** of the loop is: "Is there still laundry in the basket?"
Answer: "Yes!"
In the loop, **four instructions** are executed one after the other:
1. Take a wet piece of laundry
2. Hang the piece of laundry on the clothes horse
3. Use two clothespins
4. Fasten the piece of laundry with the clamps

If the answer to the condition "Is there still laundry in the basket?"
is "No!", the program continues behind the loop:
"Bring the basket to the bathroom."

infintite loop

An infinite loop **has no condition** und and will run until the Calliope mini is switched off.

variable

A variable is a container for a specific value (number, word, etc.), image or something else that is set at the beginning of the program. Each variable needs a unique name and you have to decide if the variable should store a number, a word (→ string), an image or something else.

branch with
a condition

Every branch in a program needs a condition.
The condition defines the next instructions in the program.
There are two ways of doing this, for example:
Condition: "Is the laundry on the clothes horse still wet?"

branch

If yes -
then: "Wait an hour"

If no -
then: "Take off the laundry"