PROGRAMING MICRO:BIT WITH MICRO-PYTHON ON HIGH SCHOOL

The aim of the workshop is to present the BBC Micro: bit development board primarily intended for teaching programming in primary and secondary school. In our course, we will focus on the MicroPython programming language as a subset of the Python language. We will start from the very basics, but gradually we will move on to the possibility of communication between two Micro: bits and we will show our work in a pair or even in a larger group. The final product should be the creation of a security device consisting of two Micro: bits, one of which has the function of a sensor and the other a detector.

BASIC INSTRUCTION

This course is recommended for first- or second-year high school students or higher grades of eight-year high schools. It is recommended especially for students of technical schools and grammar schools. Especially in technical schools, it is good to count on an even more complex platform, such as Arduino.

Here I present the recommended course and time allocation. The course can be divided into four parts:

1) Working with outputs - output to the display and audio playback.
   a) If a MicroPython course has already taken place on the Micro: bit, this section can be skipped or minimized.
   b) If students already had some programming, then one lesson (45 minutes) will suffice.
   c) If students have not yet had programming and do not know Micro: bit, then I recommend two lessons.
2) Working with inputs - button press, motion detection, magnetic field, sound, light, temperature measurement.
   a) If students know Microbit and have programmed, one lesson will suffice. Be careful, if they know the Microbit V1, there are some extra things for the new type.
   b) In all other cases, I recommend two lessons. Conditional statements must be discussed.
3) Communication between two Micro: bits - sending messages between two Micro: bits using the built-in radio module. Security of this communication.
   a) If students have already worked with this, it is possible to skip this passage completely.
   b) Otherwise, I recommend one lesson.
4) The creation of the security device itself is the same for everyone. I recommend at least two lessons. In the first lesson, the teacher chooses one of the security options and constructs it together with the students. In the second and possibly other students, they work independently on their projects. There is an option to add one or more lessons, where students will present their projects to others.

   The minimum time for which this learning package can be completed is therefore two lessons - the first lesson of parts 1 to 3, the second lesson of part 4. The recommended number is eight hours (2, 2, 1, 3), including the presentation of projects. It is necessary to consider the possibility that it will be necessary to add or subtract hours during the run.

For parts 1 to 3 and the first lesson of part 4, a suitable teaching method is the interpretation associated with the independent work of students in the lesson. The teacher should always explain part of the material and then let the students try everything and come up with other options if necessary.

It should also be considered that from part 3 onwards, pupils should work in ideally two-member teams if there are few Micro-bits so multi-member that there are at least two Micro-bits in each team.

In the final lessons, project teaching can be successfully used, where students either get an assigned topic or choose it completely. In the last hour, each team should present its functional security equipment and demonstrate its strengths and weaknesses.

Micro: bit version 2 (Micro: bit V2) is recommended for teaching. Best for each student one. I recommend equipping you with additional USB cables and battery holders to run the Micro: bit outside the computer. It can be purchased in a set for approx. 20 Euro. At the same time, I recommend equipping with spare AAA batteries.

If we have Micro: bit V1 it is necessary to deal with the audio output as described in the textbook below. In that case, you will need two cables with alligator clips for each and some sound source, such as headphones.

We will use the programming language MicroPython, which is a subset of Python. We have two options for the Editor in which we will write programs:

1. Thonny - I think it's a better choice. It only allows the program to run remotely on the Micro: bit without saving to the Micro: bit. But micro: bit storage is also possible. This is useful, for example, when testing the sound - after pressing reset the sound will end. On the other hand, there is no syntax checker.

2. Mu - Allows you to check the syntax, but the program must be saved to micro: bit before running.

From the point of view of the described programs, it does not matter which of the editors you choose.

As for the teaching materials, we have several options.

There are the following pages about MicroPython on micro: bit: BBC micro: bit MicroPython documentation.

Many more information and instructions; including many interesting projects can also be found at the original BBC Micro: bit project website.

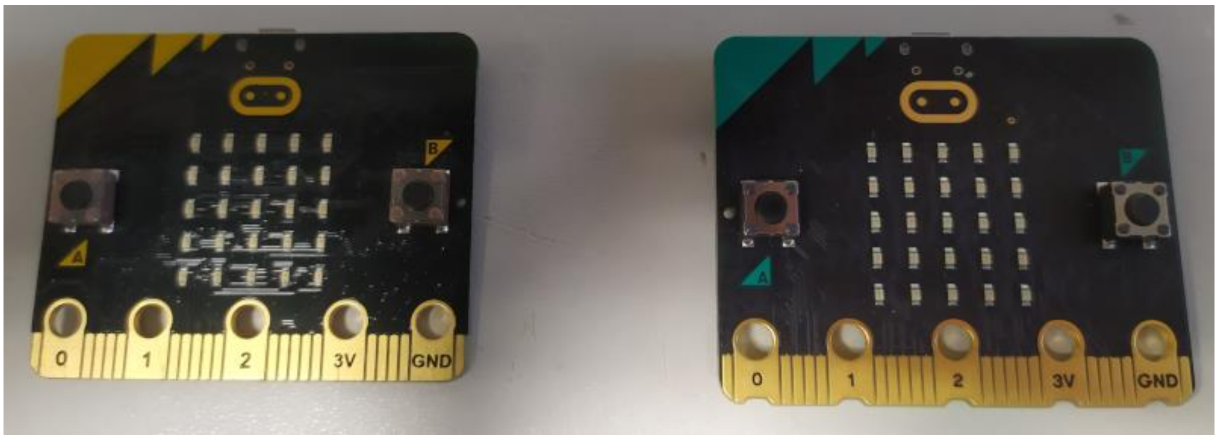
THEORETICAL PART TO THE PROBLEM

Micro: bit is a handheld computer that introduces you to the world of software and hardware and their interaction. It includes an LED display, sensor buttons and many programmable functions. The new micro: bit version also includes a microphone and buzzer. (http://www.microbit.org)

Features of Micro: bit (V2 means only V2 version):

- Nut 5x5 LEDs
- Two programmable buttons
- One touch button (V2)
- Accelerator, allowing you to respond to movement
- Light sensor, magnetic field and thermometer
- Microphone (V2)
- Buzzer (primitive speaker) (V2)
- Possibility of mutual communication via radio
- Bluetooth
- Connectors for connecting other devices, three of them for convenient connection using cables with alligator clips
- Easy connection to expansion devices via connector strip

Existují dvě verze micro:bitu. Starší verze prodávaná od roku 2016, nyní označovaná jako V1 a novější od října 2020 označovaná jako V2.



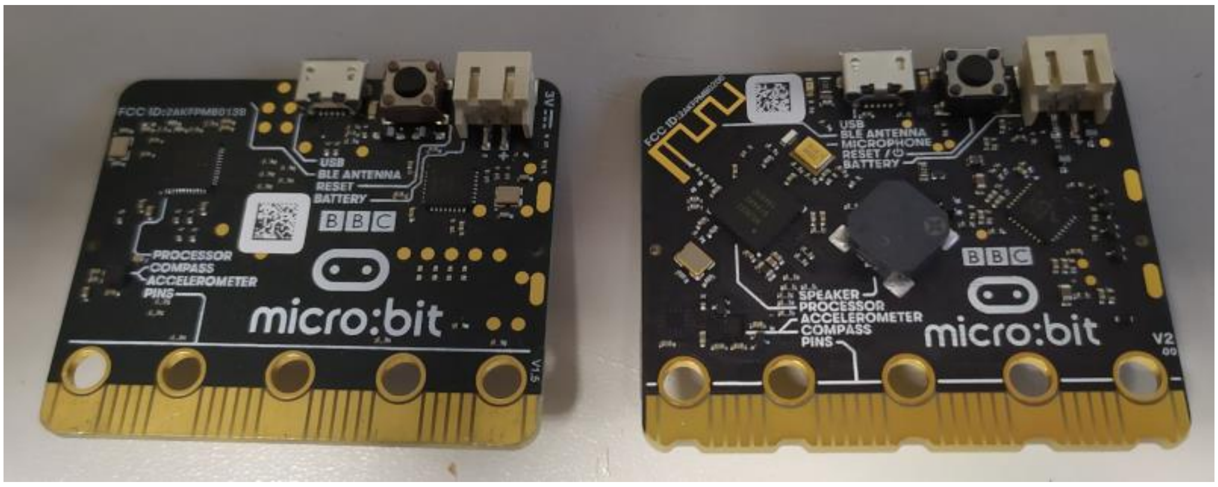
Picture 1 front side micro:bit, left V1, right V2

Buy V2 now, it is better equipped and has more memory for programs at de facto the same purchase price. Its advantage is the integrated buzzer and microphone, so you don't need anything other than two microbits to work with this text.

Micro: bit can be programmed using both graphical programming and classical text languages.

Graphic programming - such programming, where we do not create a program by typing code on the keyboard, but by moving graphic blocks across the screen. These tools include Scratch, MakeBlock or Robert Lab's Open

Microsoft MakeCode - A graphical programming language developed by Microsoft for Micro: bit programming.



Picture 2 Back side micro:bit, left V1, right V2

In the case of micro: bit, the following "classic" text programming languages JavaScript and MicroPython can be used.

- JavaScript can be used in the online editor at microbit.org if we switch to it in the program environment of the graphical programming language MakeCode.
- We can program MicroPython again in the online environment of the micro: bit homepage or with the help of the already mentioned editors Mu and Thonny.

EXAMPLES FROM PRACTICE

The author was created to create this material by an incident from practice, which happened when he verified his textbook in primary school as part of testing in the first year.

Two brothers who own two micro-bits created primitive security devices at home. They hid one microbit on the porch of the house and programmed it to respond to a change in light by sending a signal to another micro: bit located in their room. He had it connected to the speaker, and he alerted them to this fact. So, when one of the parents came home, they were informed about it in time and could put away tablets, books, etc. and take the textbooks in their hands.

Although from a pedagogical point of view, of course, I do not agree with that, from a didactic point of view, this is an interesting case associated with spontaneous teamwork and the use of knowledge gained in the circle.

Thus, it could be concluded that by completing this course, students will acquire or deepen basic programming habits. They will also learn information from the world of hardware and software and hardware integration. Here it only depends on the teacher, who has to estimate the level of his students and how deep he can allow himself to dive into the given issue.

Furthermore, possible security problems of wireless communication can be well explained on this issue - eg eavesdropping, man in the middle attack, etc.

Finally, this task can be used to develop group work - two micro: bits are required for the vast majority of cases to work properly. In the end, students can be successfully assigned a project in a group, which they will then have to present and defend before classroom.

METHODOLOGICAL AND DIDACTIC PART

As explained in the first chapter, the course is divided into four parts. Gradually we will stop at all four and say what to learn in them.

Working with outputs

- In this section, you need to master the following tasks:
- Get acquainted with micro: bit
- Familiarize yourself with the Mu or Thonny editor environment
- Learn to load the generated code into a micro: bit
- Working with the display - display of icon and text
- Working with sound

At the beginning, the teacher hands out micro: bits to the students and briefly introduces them to them. Alert students to the fact that the micro: bit has labels of the individual components. Explain the meaning of the individual buttons - A, B programmable buttons, Reset is used to restart the program from the beginning or to turn off the micro: bit while holding for 3 seconds.

Students will now launch the selected editor. Familiarize them with the environment and the basic principles of using the selected editor. In the next course, I will assume the use of my recommended editor Thonna.

Try to create a simple program:

```
from microbit import*

display.scroll("Ahoj svete")
```

and upload it to micro: bit. Show students the difference between Save (the program is saved to disk) and Run (the program will run directly to the micro: bit). Explain that once connected, the micro: bit is accessible in the same way as a flash drive, and a stored program can be loaded into it by dragging it to that drive. Show students that the yellow LED on the back flashes during recording. Also show them that they can start the terminated program again by pressing the Reset button.

For other programs in this section, the source code will usually no longer be inserted. You will find them in the attached worksheets.

Have students modify the program by adding a block for endless repetition. Try adding a time delay. Try displaying the prepared images, again with suitable time delays. If necessary, try a cycle for a fixed number of repetitions. Explain to the students the differences between the while and for loops.

Try audio playback. If you have micro: bit V1 you must first connect an output device. Try the "play tone", "play melody" and "play sound" blocks (V2 only). You can let the students compose their own melody, but in that case, I recommend equipping them with ear flaps and soundproofing the classroom.

Working with inputs

• In addition to the needs of the previous chapter, I recommend one or more strong magnets per class.

The simplest type of input are two buttons marked A and B. Their easiest use is using the block "After pressing the button A (B)" in the input category. Program the action (output to display, play tone "after pressing the selected button. Assign the activity to both buttons. For example:

```
from microbit import *
while True:
    if button_a.is_pressed():
        display.show(Image.HAPPY)
    if button_b.is_pressed():
        display.show(Image.SAD)
    sleep(100)
    display.clear()
```

- Try other input options now:
- microphone.sound_level() – test for sound level at micro: bit V 2
- pin_logo.is_touched() – pressed logo on micro: bit V 2
- accelerometer.was_gesture ("shake") – micro: bit shake
- temperature() – temperature
- display.read_light_level() – lighting level
- compass.get_field_strength () – magnetic field strength
- compass.heading() – azimuth in degrees, initialization required using compass.calibrate()

Write a program that displays the temperature after pressing button A, the intensity of light after pressing button B and the intensity of the magnetic field at the same time pressing both buttons. Experiment with turning off and on and lighting up and approaching the magnet.

You can try magic on the uninitiated. When you bring your clenched fist closer to the micro: bit, a smiley will appear. Don't say you have a strong magnet in that fist and challenge them to try it on you. You can improve the task by saying that the image will only show to those who have magical potential. You can then replace the word magical with the word magnetic and watch when the person notices.

You can try another magic program, when shaken it will give you a micro: bit random answer yes / no. You can ask him yes / no questions. You can also add options I don't know, can't decide, etc..

Communication between two micro:bits

• To meet this hour, at least two-member groups equipped with two micro-bits must already be created

To begin with, program such a simple program on both micro: bits in a group:

```
from microbit import *
import radio
chnl = 23
radio.on()
radio.config(channel = chnl)
display.clear()
while True:
    if button_a.is_pressed():
        radio.send("Zprava")
    mess = radio.receive()
    if (zprava):
        display.scroll(mess)
        mess = ""
    sleep(100)
radio.off()
```

It is important that different groups have a different radio group (here 23), otherwise they will crawl into each other's cabbages. We select the group from the interval <0.83>. It is best to let students first see what happens when they do not have a good idea of wireless security.

After the students have divided the radio groups, let them try to send a signal to each other. If possible (and safe), drop them out of the classroom to test the range of micro: bits in free space and over walls. This will come in handy when creating a security device.

Edit the more complicated case where a different number is sent when you press A and another when you press B. If you have V2, the third is sent when you press the logo. The receiving micro: bit then decides on its activity according to the received number.

Pupils can modify the code so that the meaning when sending a signal from one micro: bit can be eg "Let's go to the playground?", "Let's go to learn", "We have tasks" and answers back eg "Yes", "No" and " I don't know. " Let the students choose and test the solution. Micro: bits can now be marked as interrogative and matching.

Explain the dangers of this communication to the students. If an attacker knows the group used and the meaning of the codes, he can eavesdrop on the conversation. In addition, it can enter the communication and cause confusion for the participants.

Creation of security equipment

We will create a simple security device that will respond to light intensity.

First, try a difference in the value of light intensity between switching off and switching on (closed and open blinds) on a simple program.

```
from microbit import *

while True:

    display.scroll(display.read_light_level())
```

We need two micro: bits:

- Transmitter - monitors the light intensity and sends a signal when turned on (off). The code could look like this:

```
from microbit import *

import radio

chnl = 23

radio.on()

radio.config(channel = chnl)

while True:

    display.scroll(display.read_light_level())

    if (display.read_light_level()>127):

        radio.send("Alarm - light on")
```

- Receiver - receives info that an incident has occurred and issues an alarm. For example, code as follows:

```
from microbit import *
import music
import radio
chnl = 23
radio.on()
radio.config(channel = chnl)
display.clear()
while True:
    msg = radio.receive()
    if (msg):
        display.scroll(msg)
        msg = ""
        music.play(music.NYAN)
    sleep(100)
radio.off()
```

Now give students a project to create a security device using two micro: bits. Tell them what they can use - pressing a button (on the contrary, releasing the button), changing the intensity of light, temperature, magnetic field, audio signal, movement.

Recommended aids

First, we need enough micro: bits. The ideal situation is when each student has their own micro: bit. It is more ideal if it has its personal micro: bit, if the school or the parents of the pupils can afford it.

We need a USB connection cable for every micro: bit, and I also recommend a portable battery holder. It is ideal to buy everything together as a convenient package, as described in the introduction.

We also need a computer that has the Mu or Thonny editor installed, or probably both. I definitely recommend the second option; we will not be given that much to the whims of the internet connection.

We also need a stronger magnet to explain the reaction to the magnetic field. E.g., "Borrowed" from the bulletin board.

If we have an older micro: bit version one then we still need external speakers or headphones.

WORKSHEETS

Worksheets are attached to this material. These worksheets are available in an editable electronic form for each teacher to edit, eg according to what they have already discussed with the pupils.

There are four worksheets for each of the above topics, one. In the worksheets, I anticipate that the school has Micro: bit V2 at its disposal. If only the older version V1 is available, these sheets must be modified. The differences are mainly in the fact that with V1 it is necessary to connect an external device for audio playback and further V1 does not have a microphone and a touch button, so these two devices cannot be used as an alarm sensor.

WORKSHEET 1

WORKING WITH OUTPUTS MICRO:BITU

What we need

- A computer with the Thonny or Mu editor installed
- Micro: bit V2
- USB connection cable

Let's go

Take your micro: bit and take a good look at it. On the front you have an array of 5x5 LEDs that can glow with different intensities of red. On their sides you have two programmable buttons marked A and B. Above the diodes there is also a programmable touch button and at the top right above the LED is a small dot, which is a microphone. Now turn the micro: bit and look at the back. Above we have two microUSB ports for connecting to a PC and a connector for connecting a battery pack. Among them is the RESET button to restart the program from the beginning. Note that other sections are described here. For completeness, let's add that from both sides below we see pins for connecting other peripherals. For example, the entire micro: bit can be plugged into an external device.

Start the editor Thonny or Mu and get to know him.

Try to create a simple program:

```
from microbit import*

display.scroll("Hello world")
```

and upload it to micro: bit. Try to explain between the options to run and save as with Thonna. Note that the yellow LED on the back flashes during recording. Also test that the program can be started repeatedly using the RESET button (or by disconnecting and connecting the micro: bit to the power supply).

Try adding a while block to True (note that True must be a capital letter) and add a timeout. Try displaying the prepared images, again with suitable time delays.

```
from microbit import*

while True:

    display.scroll("Hello world")

    sleep(100)
```

We will modify the program so that the text is displayed three times:

```
from microbit import*
for i in range(1, 4):
    display.scroll("Hello world")
sleep(1000)
display.clear()
```

Note that for three views, the range really must be 1 to 4. This is actually a half-closed (left) interval.

Try audio playback:

```
from microbit import *
import music
display.show(Image.HAPPY)
while True:
    music.play(music.POWER_UP)
    sleep(1000)
```

When your ears hurt, press the reset button on the micro: bit (Thonna only). You can even let micro: bit talk:

```
from microbit import *
import speech
speech.say("Attention please",speed=100)
```

What have you learned?

Display images and text on the micro: bit display. You know how to run part of the code once, repeatedly, several times. You can also play a tone on the micro: bit and let it speak.

WORKSHEET 2

WORKIN WITH OUTPUT MICRO:BIT

What we need

A computer with the Thonny or Mu editor installed

• Micro: bit V2

• USB connection cable

Let’s go

The simplest type of input is two buttons marked A and B. Micro: bit V2 additionally touch logo. The following example demonstrates its use well:

```
from microbit import *
import music
while True:
    if button_a.is_pressed():
        display.show(Image.HAPPY)
    elif button_b.is_pressed():
        display.show(Image.SAD)
    elif pin_logo.is_touched():
        music.play(music.DADADADUM)
    sleep(100)
    display.clear()
```

This is a conditional command. If the first condition applies, it executes and jumps to the end of the block, and so on. The part is missing here.

Now try to measure other quantities and display their value: light intensity, magnetic field and temperature. You can also try the azimuth, don't forget the micro: bit must be calibrated before use (the micro: bit will write a prompt, then a dot will appear in the middle of the display and by turning the micro: bit we will have to fill the whole display).

Use the following commands:

- microphone.sound_level () – (only V2) sound test
- pin_logo.is_touched () – if was pressed logo on micro: bit (onlyV 2)
- accelerometer.was_gesture ("shake") – if was shake
- temperature () - temperature
- display.read_light_level () - lighting level
- compass.get_field_strength () - magnetic field strength
- compass.heading () - azimuth in degrees, initialization required using command compass.calibrate()

Write a program that displays the temperature after pressing button A, the intensity of light after pressing button B and the intensity of the magnetic field at the same time pressing both buttons. Experiment with turning off and on and lighting up and approaching the magnet.

Try experimenting similarly with the volume level around.

You can try magic on the uninitiated:

When you bring your clenched fist closer to the micro: bit, a smiley will appear. Don't say you have a strong magnet in that fist and challenge them to try it on you. You can improve the task by saying that the image will only show to those who have magical potential. You can then replace the word magical with the word magnetic and watch when the person notices it.

```
from microbit import *
level = 5000
compass.calibrate()
startLevel = compass.get_field_strength()
while True:
    sleep(100)
    strngth = compass.get_field_strength()
    if abs(strngth - startLevel) > level:
        display.show(Image.HAPPY)
        sleep(3000)
        display.clear()
```

You can try another magic program, when shaken it will give you a micro: bit random answer yes / no. You can ask him about I / I don't have to.

What have you learned

Program input events such as button press or microbit shake. You have learned to subtract physical quantities from the real world - temperature, light, noise and magnetic field. You will use all this later in the creation of security devices.

WORKSHEET 3

KOMUNIKACE MEZI DVĚMA MICRO:BITY

What we need

- Computer with any Python editor installed
- 2 Micro: bits preferably V2
- USB connection cable
- I recommend using a Battery Box for both Micro: bits - to test the connection in the "field"

Let´s go

At least two groups of at least two micro:bits must be created in this and the following lesson

To begin with, program such a simple program on both micro: bits in a group:

```
from microbit import *
import radio
chnl = 23
radio.on()
radio.config(channel = chnl)
display.clear()
while True:
    if button_a.is_pressed():
        radio.send("Zprava")
    msg = radio.receive()
    if (msg):
        display.scroll(msg)
        zprava = ""
    sleep(100)
radio.off()
```

It is important that different groups have a different radio group (here 23) from the interval <0.83>, otherwise you will climb into each other's cabbage. All micro: bits with the same group should receive the signal.

Try to send a signal to each other. Connect the micro: bits to the battery pack and vice versa, disconnect them from the USB cable and take them for a walk outside and test the signal range.

For testing, you can also record the following program on one micro: bit, which you leave connected to the computer, and go "for a walk" with only one micro: bit on which you leave the previous program and find out the range. Or modify the program on the micro: bit to send messages automatically after two seconds and examine the range of the micro: bit from the classroom.

Attention. If you connect the Micro: bit to a computer, always disconnect the Baterry Pack first to avoid damaging the Micro: bit by higher voltage.

Edit a more complicated case where another number is sent when A is pressed and another when B is pressed. The receiving micro: bit then decides on its action according to the received number (eg which message to display).

If you have V2, edit the example so that the third number is sent when you press the logo.

Modify the code so that the meaning when sending a signal from one micro: bit can be eg "Let's go to the field?", "Let's go to learn", "We have tasks" and answers back eg "Yes", "No" and "I do not know “. In this case, it will be necessary to have a different code in micro: bits.

Be aware of the dangers of this communication. If an attacker knows the group used and the meaning of the codes, he can eavesdrop on the conversation. In addition, it can enter the communication and cause confusion for the participants.

What have you learned?

How two micro:bits communicate.

Send a message from one micro: bit to another and decrypt its meaning.

You also learned something about wireless security.

WORKSHEET 4

TVORBA ZABEZPEČOVACÍHO ZAŘÍZENÍ

What we need

- A computer with the Thonny or Mu editor installed
- Micro: bits preferably V2
- USB connection cable
- I recommend using the Battery Box for both Micro: bits - the possibility of placement outside the computer

Let's go

We will create a simple security device that will respond to light intensity. We will work in pairs again.
First, try a difference in the value of light intensity between switching off and switching on (closed and open blinds) on a simple program.from microbit import *

```
while True:
    display.scroll(display.read_light_level())
```

Set a value that will mean the dividing value between light and dark for you to sound the alarm.

We will now create a simple security device. You can replace the value 128 with the observed value from the previous program. This time we will record different programs for each Micro: bit.

- Transmitter - monitors the intensity of light and sends a signal - alarm when turned on (off). The code could look like this:

```
from microbit import *
import radio
chnl = 23
radio.on()
radio.config(channel = chnl)
while True:
    display.scroll(display.read_light_level())
    if (display.read_light_level()>127):
        radio.send("Alarm - light on")
```

- Receiver - receives info that an incident has occurred and issues an alarm. For example, the code might look like this:

```
from microbit import *
import music
import radio
chnl = 23
radio.on()
radio.config(channel = chnl)
display.clear()
while True:
    msg = radio.receive()
    if (zprava):
        display.scroll(msg)
        msg = ""
        music.play(music.NYAN)
    sleep(100)
radio.off()
```

Try to modify the receiver code so that it repeats the warning until, for example, the button is pressed. (alarm acknowledgment)

Now create any security device using two micro: bits.

You can try different types of events:

- pressing the button (on the contrary, releasing the button)
- change in light intensity
- temperature change
- change in magnetic field strength
- audio signal (sound)
- micro movement: bit (shake)

What have you learned?

The knowledge gained in this course is actually knowledge from the world of robotics. You can now monitor your surroundings and detect its changes - this is an analogy with the human senses. On the one hand, you do not have the senses of "taste and sight", but on the contrary, you have a sense of the intensity of the magnetic field. You can now respond to these events with display messages or sound. You can also communicate with another micro: bit remotely.

It is important to realize that although this course has taught you a lot, it is up to you how you handle the acquired skills and whether you will further deepen them with the help of other courses or other literature.