5.3 Testing programming code

Testing code in Arduino is important because it helps to ensure that the code is working correctly and producing the desired results. Testing can help to catch bugs and errors in the code, and can also help to verify that the code is performing the tasks that it is intended to perform. By thoroughly testing code, you can improve the reliability and functionality of your Arduino projects.

The Serial.println() function is a useful tool for debugging Arduino code because it allows you to print information to the serial monitor, which can help you understand what your code is doing and troubleshoot any problems.

To use Serial.println() for debugging, you will need to include the Serial library at the top of your sketch and initialize the serial monitor using the Serial.begin() function. Then, you can use Serial.println() to print strings or variables to the serial monitor.

Here is an example of how to use Serial.println() for debugging in an Arduino sketch:

```
void setup() {
  // Initialize serial communication at a baud rate of 9600
  Serial.begin(9600);
}

void loop() {
  int x = 10;
  int y = 20;

  // Print the value of x and y to the serial monitor
  Serial.print("x = "); Serial.println(x);
  Serial.print("y = "); Serial.println(y);

  // Print the result of an operation to the serial monitor
  int sum = x + y;
  Serial.print("sum = "); Serial.println(sum);
}
```

To view the output of the Serial.println() statements, you will need to open the serial monitor in the Arduino IDE. You can do this by clicking on the “magnifying glass” icon in the top right corner of the window.

5.3.1 Testing mode

Since testing programming code and hardware is one of the key features in designing a robot it is recommended that testing functions are a part of your main program.

We will write a program which will be triggered by sending specific text over the serial communication. For example if we will send the string `forward` the function `moveForward()` will be executed. The

function `serialEvent()` is trigged when some data are available on serial communication. A basic example of such functionality is shown in prog. 1:

Program 1: Testing programming code.

```
#include "RobotMovingFunctions.h"
//#include <RobDuinoSerialTesting.h>

void setup()
{
  Serial.begin(115200);
  setIOpins();
  moveForward();
  delay(3000);
  stopTheRobot();
}

void loop()
{

}

void serialEvent(){
  String test_string = Serial.readString();
  if      (test_string == "forward")  moveForward();
  else if (test_string == "stop")     stopTheRobot();
  else Serial.println("\n" + test_string + " is not valid command.");
}
```

5.3.2 Task: Try testing workflow

1. Explore the testing functionality of this example.
2. Complete the testing functionality with other functions available to control the robot movement.

In further lectures we will be using more advances `Testing mode` where single digital outputs can be controlled; and inputs will be measured in digital and analog manner. This testing process is available if you have installed `RobDuino Libraty` (see Program Installing chapter). The testing mode will be trigged by the command `testing`. The output will show every output state:

```
******* Testing mode ********************
Dig. Out    Dig. In.    An.In.
D0 = 1      A0 = 0      A0 = 293
D1 = 0      A1 = 0      A1 = 334
D2 = 0      A2 = 0      A2 = 353
D3 = 0      A3 = 0      A3 = 369
D4 = 0      A4 = 0      A4 = 339
D5 = 0      A5 = 0      A5 = 264
D6 = 0
D7 = 0
-----------------------------------------
```

5.3.3 Task: RobDuino module testing

3. Delete `serialEvent()` function in your program and uncomment line 2 in prog. 1: `#include "RobDuinoSerialTesting.h"`.
4. Explore testing functions with command `testing` writing it into Serial Monitor and you will get this respond:

```
*** Testing mode - menu - ******************
*    help - prints this text menu
*    D5   - toggles output state of D5
*    Dx   - toggles output state of any Dx,
*           x is any num. from 0 .. 13.
*    run  - toggles monitoring od I/O pins
*    exit - exits the Testing mode.
*----------------------------------------
Type any command to continue ...
```

5.3.4 Questions:

1. Explain why testing is important.
2. Describe the techniques of testing.
3. What parts of the robot should be tested regularly.

5.3.5 Summary:

5.3.5.1 Testing mode

5.3.6 Issues:

5.3.6.1 How can I get RobDuinoSerialTesting working.

Basicaly you need to do these stps:

1. install RobDuino Library
2. put this code at the top of your porgram:
   ```
   #include <RobDuinoSerialTesting.h>
   ```
3. Compile and write the porgram to your Arduino UNO controller
4. Open `Serial Monitor` window
5. and write `testing` command into prompt.