Internet of Things

Challenge 4 - REPORT

Student 1 : Valeria Maria Fortina **ID 1**: 10537962

Student 2 : Simone Secondari **ID 2**: 10561308

As a group, we developed the TinyOS application for challenge 4. We simulated the program using TOSSIM and the debug statements to check if everything was working according to the request. We used the person code **ID 1**: 10537962
to get the x=3 and y=37 and we booted the mote 1 at time 0, and the mote 2 at time Y=37 as requested. To do so we used the code inside RunSimulationScript.py
`time2 = 37*t.ticksPerSecond(); node2.bootAtTime(time2);`

sendACK.h
In the file sendACK.h we defined the structure of the message, for simplicity we decided to use only one type of message structure for both the requests and the responses.
Every message has three elements, one defining the type, one for the counter, and the last one for the value of the fake sensor, which would be zero in the request messages.

sendAckAppC.nc
In the file sendAckAppC.nc, we defined the components and wired the interfaces. In particular, we used a component for the FakeSensor, one for the Timer, and two components to set the Sender and the Receiver of the messages. As we wired the interfaces we also connect PacketAcknowledgements to use ACKs.

sendAckC.nc
The file sendAckAppC.nc delineates the logic of our application. In the first part of the code we defined the interfaces, and then the implementation.
We used the function `sendReq()` conforming to the instructions to prepare the message and set the ACK inside the packet. Then we send the message using the Unicast of the second mote.
In the `event Boot.booted()` we set the periodic timer of 1000ms as requested.
In the `event AMSend.sendDone` we checked if the packet was sent and if the ACK was received. Here we also incremented the counter at each request and we called the function `MilliTimer.stop()` to stop the timer after the Xth ACK, which in our case was the third as we used the person code 10537962.
In the `event message_t *Receive.receive` we read the content of the message and check the type, if it was a request we sent the response using sendResp().
Finally in the `event Read.readDone` we prepared the response, setting also the ACK and using unicast to send the message to mote 1.