```cpp
#include "rs232.h"
#include <QDebug>
#include <QEventLoop>

#define AmountOfCOMPorts 5
//#######FUNKTIONSBESKRIVELSE:
//#######Konstruktor
//############################
rs232::rs232(QObject *parent) :
    QObject(parent)
{
    //Sætter portens indstillinger
    settings.BaudRate = BAUD57600;
    settings.DataBits = DATA_8;
    settings.FlowControl = FLOW_OFF;
    settings.Parity = PAR_NONE;
    settings.StopBits = STOP_1;
    settings.Timeout_Millisec = 10;

    //Allokerer plads til porten og opretter den
    //Fordi den serielle kommunikation bliver tildelt en tilsyneladende
tilfældig COM-port hver gang den tilsluttes tester
    //hvilken COM-port den kan kommunikere med i intervallet COM5-COM9
    QString comArray[] = {"COM5", "COM6", "COM7", "COM8", "COM9"};
    const char *str;
    QByteArray temp;

    int i;
    for(i=0; i < AmountOfCOMPorts-1; i++)
    {
        temp = comArray[i].toLatin1();
        str = temp.data();
        port = new QextSerialPort(str, settings, QextSerialPort::Polling);

        if (port->open(QIODevice::ReadWrite))
        {
            qDebug() << "RS232: Port åbnet på " << comArray[i];
            break;
        }
    }
    if (i == AmountOfCOMPorts-1)
    qDebug() << "RS232: Ingen porte blev åbnet";
}

//#######FUNKTIONSBESKRIVELSE:
//#######Destruktor
//############################
rs232::~rs232()
{
    port->close();
}

//#######FUNKTIONSBESKRIVELSE:
//#######Sende funktion
//############################
void rs232::send(int cmd)
{
    //Opretter et midlertidigt array af bytes
    QByteArray bytes;

    //Sætter første plads til kommando-værdien
    bytes[0] = cmd;

    //Hvis porten er åben sendes det antal bytes der er i arrayet
    if (port->isOpen())
    {
        int n = bytes.size();
        n = port->write(bytes, bytes.length());
```

```
        if (n < 1)
        {
            qDebug() << "Sendte intet";
        }
    }
    qDebug() << "SENDT VIA UART: " << cmd;
}

//#######FUNKTIONSBESKRIVELSE:
//#######Læser den værdi der forventeligt er blevet sent
//##############################
int rs232::readValue()
{
    //opretter et midlertidigt array
    QByteArray bytes;
    bytes.resize(1);
    int size = 0;

    //QEventLoop loop;
    //QObject::connect(port, SIGNAL(readyRead()), &loop, SLOT(quit()));

    port->read(bytes.data(), bytes.size());

    //Sætter value til den første byte vi modtog
    int value = bytes[0];

    size = bytes.length();

    //Udskriver hele arrayet
    if (bytes.length() > 1)
    {
        qDebug() << "Mere end ét element i buffer: " << bytes.length();
        qDebug() << bytes[0] << bytes[1];
    }
    //returnerer den første plads i arrayets værdi
    qDebug() << "Modtog værdi: " << value;
    return value;
}

//#######FUNKTIONSBESKRIVELSE:
//#######Returnerer hvorvidt det er ack eller nack der er modtaget
//##############################
bool rs232::readAck(Kommando cmd)
{
    int buffer = readValue(); //læs buffer

    //Hvis bufferen er negativ
    if (buffer < 0)
    {
        //Lægger 256 til grundet signed int
        buffer = (256 + buffer);
    }

    if (cmd == buffer) //hvis ack modtaget,
    {
        qDebug() << "Modtog ACK: " << buffer;
        emit this->SMConnectionChangedTo(true);
        return 1;
    }
    else
    {
        qDebug() << "ACK ikke modtaget, modtog: " << buffer;
        qDebug() << "Skulle have modtaget: " << cmd;

        emit this->SMConnectionChangedTo(false);
        port->readAll(); //Flush af bufferen
    }
    return 0;
```

```
}

//#######FUNKTIONSBESKRIVELSE:
//#######Checker hvorvidt der er forbindelse til SM vha. INIT
//############################
bool rs232::isConnectionOpen()
{
    send(CMD_INIT);
    if(readAck(ACK_INIT))
    {
        emit this->SMConnectionChangedTo(true);
        return true;
    }
    emit this->SMConnectionChangedTo(false);
    return false;
}
```