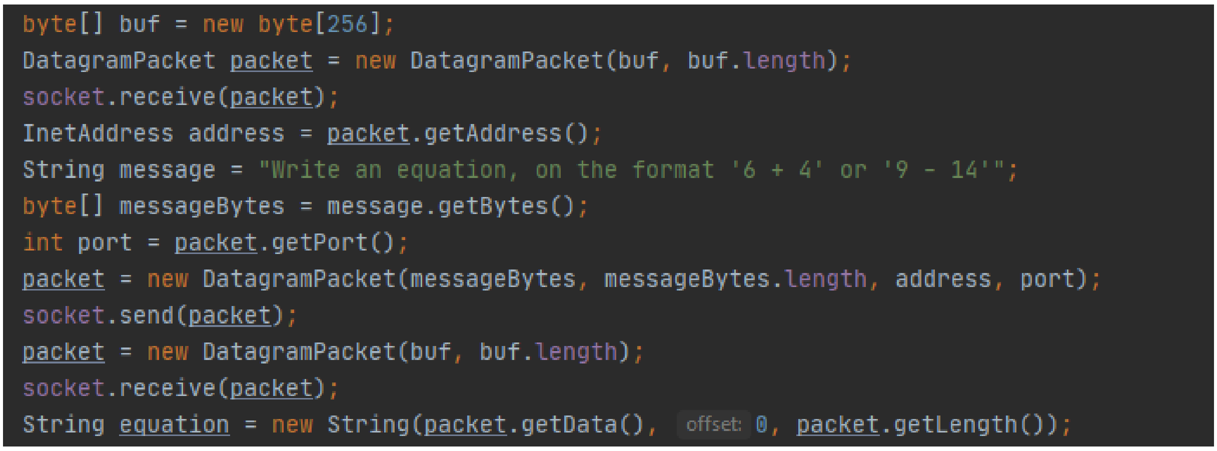
Oblig 2, Nettverksprog – Stian Mogen og Eirik Steira

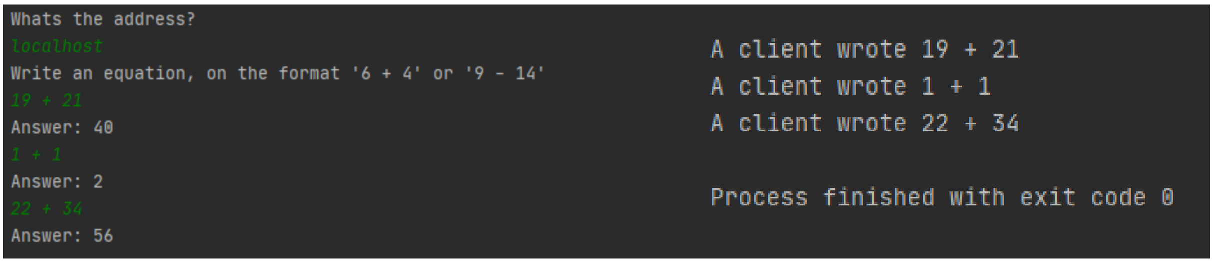
Oppgave 1

I serverprogram tar vi imot DatagramPacket, vi tar i mot og sender bytes ved bruk av adressen og port 4445. Vi aksepterer altså ikke en forespørsel fra klienten, slik som met TCP.



```java
byte[] buf = new byte[256];
DatagramPacket packet = new DatagramPacket(buf, buf.length);
socket.receive(packet);
InetAddress address = packet.getAddress();
String message = "Write an equation, on the format '6 + 4' or '9 - 14'";
byte[] messageBytes = message.getBytes();
int port = packet.getPort();
packet = new DatagramPacket(messageBytes, messageBytes.length, address, port);
socket.send(packet);
packet = new DatagramPacket(buf, buf.length);
socket.receive(packet);
String equation = new String(packet.getData(), offset: 0, packet.getLength());
```

Slik ser kommandolinjen til klienten og serveren ut under kjøring.

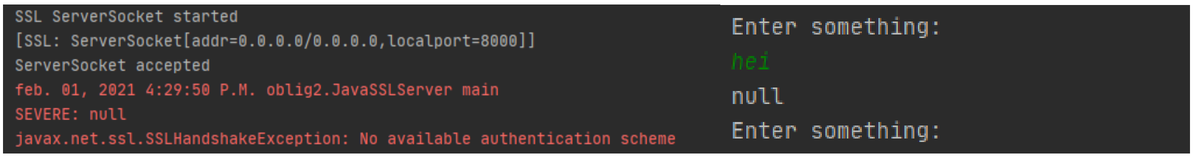


```
Whats the address?
localhost
Write an equation, on the format '6 + 4' or '9 - 14'
19 + 21
Answer: 40
1 + 1
Answer: 2
22 + 34
Answer: 56
```

```
A client wrote 19 + 21
A client wrote 1 + 1
A client wrote 22 + 34

Process finished with exit code 0
```

Oppgave 2

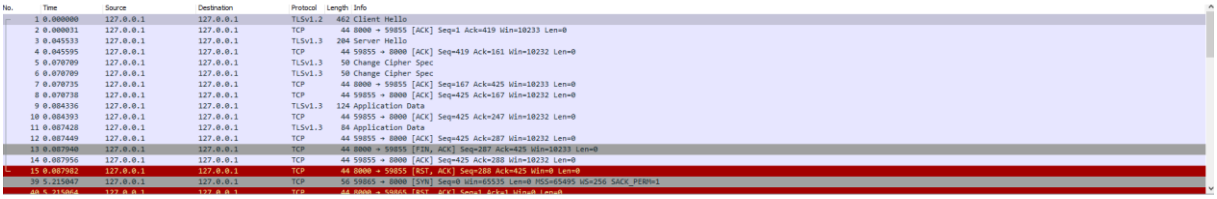
Prøver først å kjøre, det funker naturligvis ikke



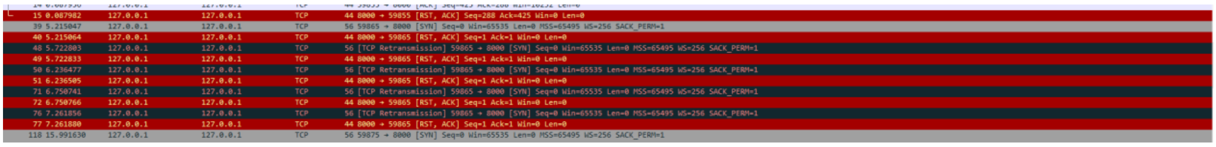
```
SSL ServerSocket started
[SSL: ServerSocket[addr=0.0.0.0/0.0.0.0,localport=8000]]
ServerSocket accepted
feb. 01, 2021 4:29:50 P.M. oblig2.JavaSSLServer main
SEVERE: null
javax.net.ssl.SSLHandshakeException: No available authentication scheme
```

```
Enter something:
hei
null
Enter something:
```

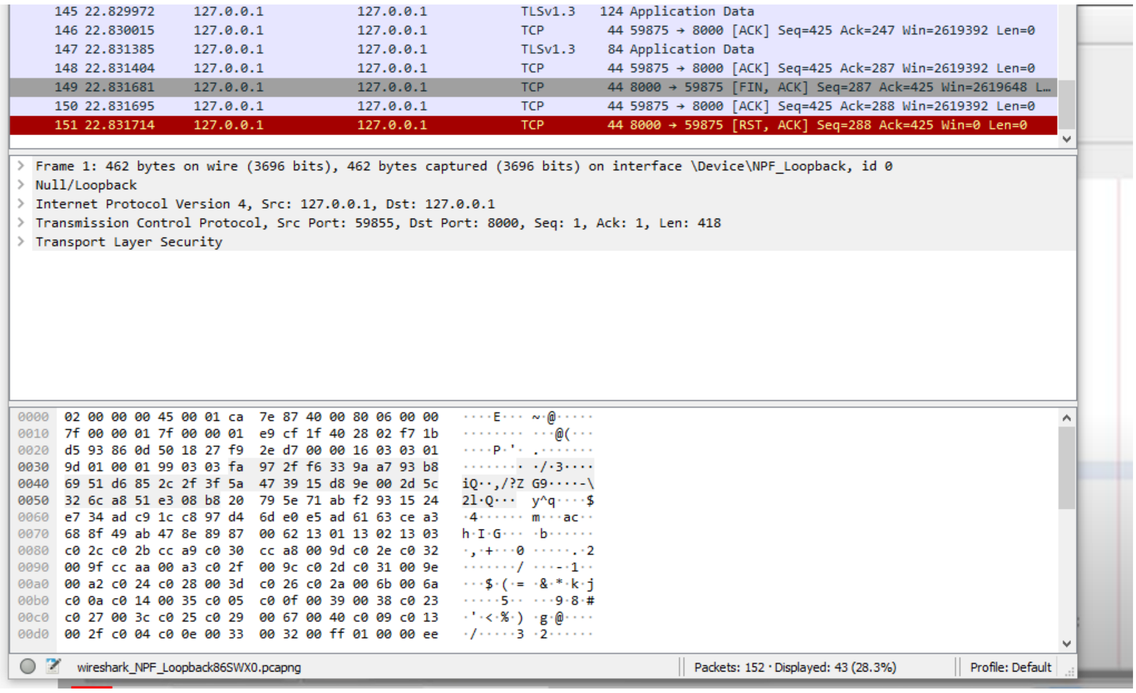
Vi ser på Wireshark og fanger pakken på port 8000.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	127.0.0.1	127.0.0.1	TLSv1.2	462	Client Hello
2	0.000031	127.0.0.1	127.0.0.1	TCP	44	8000 → 59855 [ACK] Seq=1 Ack=419 Win=10233 Len=0
3	0.045533	127.0.0.1	127.0.0.1	TLSv1.3	204	Server Hello
4	0.045595	127.0.0.1	127.0.0.1	TCP	44	59855 → 8000 [ACK] Seq=419 Ack=161 Win=10232 Len=0
5	0.070709	127.0.0.1	127.0.0.1	TLSv1.3	50	Change Cipher Spec
6	0.070709	127.0.0.1	127.0.0.1	TLSv1.3	50	Change Cipher Spec
7	0.070735	127.0.0.1	127.0.0.1	TCP	44	8000 → 59855 [ACK] Seq=167 Ack=425 Win=10233 Len=0
8	0.070738	127.0.0.1	127.0.0.1	TCP	44	59855 → 8000 [ACK] Seq=425 Ack=167 Win=10232 Len=0
9	0.084336	127.0.0.1	127.0.0.1	TLSv1.3	124	Application Data
10	0.084393	127.0.0.1	127.0.0.1	TCP	44	59855 → 8000 [ACK] Seq=425 Ack=247 Win=10232 Len=0
11	0.087428	127.0.0.1	127.0.0.1	TLSv1.3	84	Application Data
12	0.087449	127.0.0.1	127.0.0.1	TCP	44	59855 → 8000 [ACK] Seq=425 Ack=287 Win=10232 Len=0
13	0.087940	127.0.0.1	127.0.0.1	TCP	44	8000 → 59855 [FIN, ACK] Seq=287 Ack=425 Win=10233 Len=0
14	0.087956	127.0.0.1	127.0.0.1	TCP	44	59855 → 8000 [ACK] Seq=425 Ack=288 Win=10232 Len=0
15	0.087982	127.0.0.1	127.0.0.1	TCP	44	8000 → 59855 [RST, ACK] Seq=288 Ack=425 Win=0 Len=0
39	5.215047	127.0.0.1	127.0.0.1	TCP	56	59865 → 8000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
40	5.215064	127.0.0.1	127.0.0.1	TCP	44	8000 → 59865 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

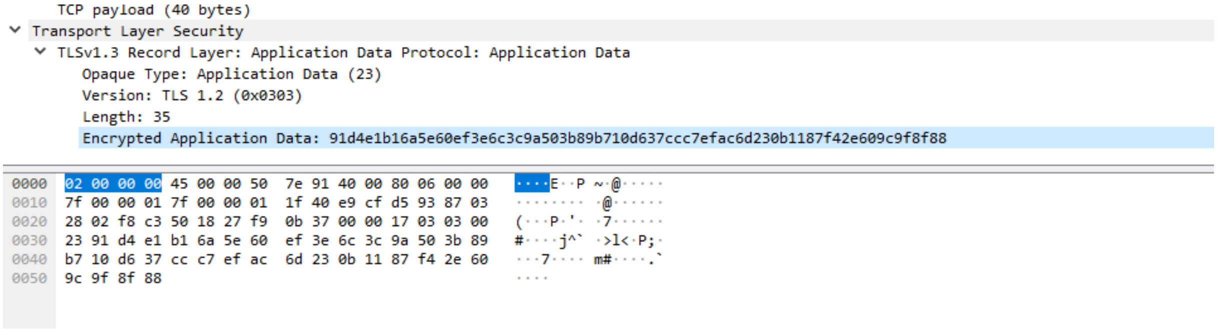


No.	Time	Source	Destination	Protocol	Length	Info
14	0.087956	127.0.0.1	127.0.0.1	TCP	44	59855 → 8000 [ACK] Seq=425 Ack=288 Win=10232 Len=0
15	0.087982	127.0.0.1	127.0.0.1	TCP	44	8000 → 59855 [RST, ACK] Seq=288 Ack=425 Win=0 Len=0
39	5.215047	127.0.0.1	127.0.0.1	TCP	56	59865 → 8000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
40	5.215064	127.0.0.1	127.0.0.1	TCP	44	8000 → 59865 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
48	5.722803	127.0.0.1	127.0.0.1	TCP	56	[TCP Retransmission] 59865 → 8000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
49	5.722833	127.0.0.1	127.0.0.1	TCP	44	8000 → 59865 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
50	6.236477	127.0.0.1	127.0.0.1	TCP	56	[TCP Retransmission] 59865 → 8000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
51	6.236505	127.0.0.1	127.0.0.1	TCP	44	8000 → 59865 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
71	6.750741	127.0.0.1	127.0.0.1	TCP	56	[TCP Retransmission] 59865 → 8000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
72	6.750766	127.0.0.1	127.0.0.1	TCP	44	8000 → 59865 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
76	7.261856	127.0.0.1	127.0.0.1	TCP	56	[TCP Retransmission] 59865 → 8000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
77	7.261880	127.0.0.1	127.0.0.1	TCP	44	8000 → 59865 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
118	15.991630	127.0.0.1	127.0.0.1	TCP	56	59875 → 8000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1



```
145 22.829972    127.0.0.1    127.0.0.1    TLSv1.3   124 Application Data
146 22.830015    127.0.0.1    127.0.0.1    TCP        44 59875 → 8000 [ACK] Seq=425 Ack=247 Win=2619392 Len=0
147 22.831385    127.0.0.1    127.0.0.1    TLSv1.3    84 Application Data
148 22.831404    127.0.0.1    127.0.0.1    TCP        44 59875 → 8000 [ACK] Seq=425 Ack=287 Win=2619392 Len=0
149 22.831681    127.0.0.1    127.0.0.1    TCP        44 8000 → 59875 [FIN, ACK] Seq=287 Ack=425 Win=2619648 L…
150 22.831695    127.0.0.1    127.0.0.1    TCP        44 59875 → 8000 [ACK] Seq=425 Ack=288 Win=2619392 Len=0
151 22.831714    127.0.0.1    127.0.0.1    TCP        44 8000 → 59875 [RST, ACK] Seq=288 Ack=425 Win=0 Len=0

> Frame 1: 462 bytes on wire (3696 bits), 462 bytes captured (3696 bits) on interface \Device\NPF_Loopback, id 0
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 59855, Dst Port: 8000, Seq: 1, Ack: 1, Len: 418
> Transport Layer Security

0000  02 00 00 00 45 00 01 ca  7e 87 40 00 80 06 00 00   ····E··· ~·@·····
0010  7f 00 00 01 7f 00 00 01  e9 cf 1f 40 28 02 f7 1b   ········ ···@(···
0020  d5 93 86 0d 50 18 27 f9  2e d7 00 00 16 03 03 01   ····P·'· .·······
0030  9d 01 00 01 99 03 03 fa  97 2f f6 33 9a a7 93 b8   ········ ·/·3····
0040  69 51 d6 85 2c 2f 3f 5a  47 39 15 d8 9e 00 2d 5c   iQ··,/?Z G9····-\
0050  32 6c a8 51 e3 08 b8 20  79 5e 71 ab f2 93 15 24   2l·Q···  y^q····$
0060  e7 34 ad c9 1c c8 97 d4  6d e0 e5 ad 61 63 ce a3   ·4······ m···ac··
0070  68 8f 49 ab 47 8e 89 87  00 62 13 01 13 02 13 03   h·I·G··· ·b······
0080  c0 2c c0 2b cc a9 c0 30  cc a8 00 9d c0 2e c0 32   ·,·+···0 ·····.·2
0090  00 9f cc aa 00 a3 c0 2f  00 9c c0 2d c0 31 00 9e   ·······/ ···-·1··
00a0  00 a2 c0 24 c0 28 00 3d  c0 26 c0 2a 00 6b 00 6a   ···$·(·= ·&·*·k·j
00b0  c0 0a c0 14 00 35 c0 05  c0 0f 00 39 00 38 c0 23   ·····5·· ···9·8·#
00c0  c0 27 00 3c c0 25 c0 29  00 67 00 40 c0 09 c0 13   ·'·<·%·) ·g·@····
00d0  00 2f c0 04 c0 0e 00 33  00 32 00 ff 01 00 00 ee   ·/·····3 ·2······

wireshark_NPF_Loopback86SWX0.pcapng    Packets: 152 · Displayed: 43 (28.3%)    Profile: Default
```

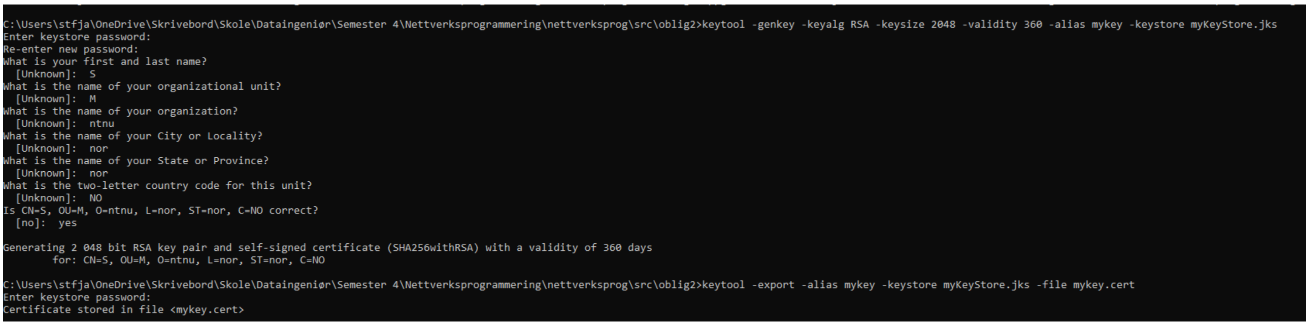
Vi må se litt nærmere, og velger TLS med Application Data:



```
        TCP payload (40 bytes)
  v Transport Layer Security
      v TLSv1.3 Record Layer: Application Data Protocol: Application Data
            Opaque Type: Application Data (23)
            Version: TLS 1.2 (0x0303)
            Length: 35
            Encrypted Application Data: 91d4e1b16a5e60ef3e6c3c9a503b89b710d637ccc7efac6d230b1187f42e609c9f8f88

0000  02 00 00 00 45 00 00 50  7e 91 40 00 80 06 00 00   ····E··P ~·@·····
0010  7f 00 00 01 7f 00 00 01  1f 40 e9 cf d5 93 87 03   ········ ·@······
0020  28 02 f8 c3 50 18 27 f9  0b 37 00 00 17 03 03 00   (···P·'· ·7······
0030  23 91 d4 e1 b1 6a 5e 60  ef 3e 6c 3c 9a 50 3b 89   #····j^` ·>l<·P;·
0040  b7 10 d6 37 cc c7 ef ac  6d 23 0b 11 87 f4 2e 60   ···7···· m#····.`
0050  9c 9f 8f 88                                        ····
```

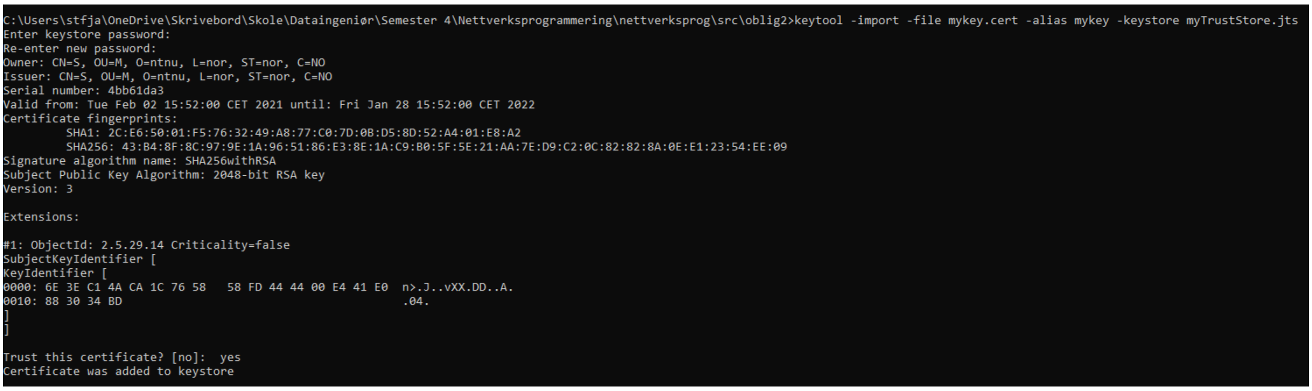
Vi ser under Encrypted Application Data at det er kryptert.

Vi må ordne en key, som beskrevet i dokumentasjonen. Bruker genkey og eksporterer denne.



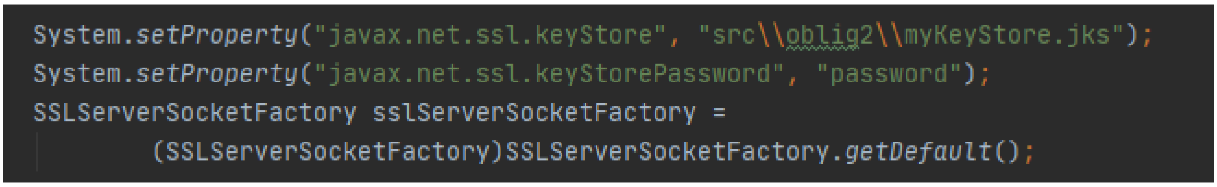
```
C:\Users\stfja\OneDrive\Skrivebord\Skole\Dataingeniør\Semester 4\Nettverksprogrammering\nettverksprog\src\oblig2>keytool -genkey -keyalg RSA -keysize 2048 -validity 360 -alias mykey -keystore myKeyStore.jks
Enter keystore password:
Re-enter new password:
What is your first and last name?
  [Unknown]:  S
What is the name of your organizational unit?
  [Unknown]:  M
What is the name of your organization?
  [Unknown]:  ntnu
What is the name of your City or Locality?
  [Unknown]:  nor
What is the name of your State or Province?
  [Unknown]:  nor
What is the two-letter country code for this unit?
  [Unknown]:  NO
Is CN=S, OU=M, O=ntnu, L=nor, ST=nor, C=NO correct?
  [no]:  yes

Generating 2 048 bit RSA key pair and self-signed certificate (SHA256withRSA) with a validity of 360 days
        for: CN=S, OU=M, O=ntnu, L=nor, ST=nor, C=NO

C:\Users\stfja\OneDrive\Skrivebord\Skole\Dataingeniør\Semester 4\Nettverksprogrammering\nettverksprog\src\oblig2>keytool -export -alias mykey -keystore myKeyStore.jks -file mykey.cert
Enter keystore password:
Certificate stored in file <mykey.cert>
```

Importerer deretter.

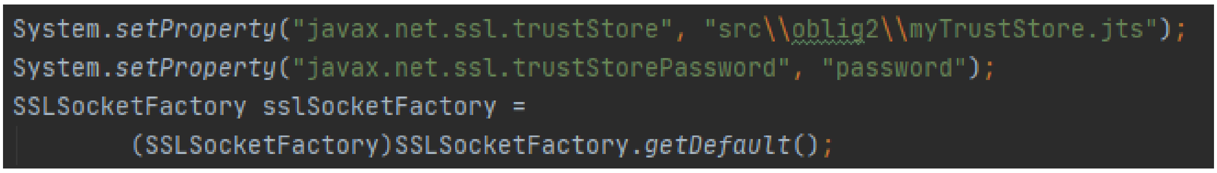


```
C:\Users\stfja\OneDrive\Skrivebord\Skole\Dataingeniør\Semester 4\Nettverksprogrammering\nettverksprog\src\oblig2>keytool -import -file mykey.cert -alias mykey -keystore myTrustStore.jts
Enter keystore password:
Re-enter new password:
Owner: CN=S, OU=M, O=ntnu, L=nor, ST=nor, C=NO
Issuer: CN=S, OU=M, O=ntnu, L=nor, ST=nor, C=NO
Serial number: 4bb61da3
Valid from: Tue Feb 02 15:52:00 CET 2021 until: Fri Jan 28 15:52:00 CET 2022
Certificate fingerprints:
         SHA1: 2C:E6:50:01:F5:76:32:49:A8:77:C0:7D:0B:D5:8D:52:A4:01:E8:A2
         SHA256: 43:B4:8F:8C:97:9E:1A:96:51:86:E3:8E:1A:C9:B0:5F:5E:21:AA:7E:D9:C2:0C:82:82:8A:0E:E1:23:54:EE:09
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 6E 3E C1 4A CA 1C 76 58   58 FD 44 44 00 E4 41 E0  n>.J..vXX.DD..A.
0010: 88 30 34 BD                                        .04.
]
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
```

For å få programmet til å kjøre måtte vi legge inn disse to kodesnuttene for i henholdsvis server- og klientprogrammet for keystore og truststore.

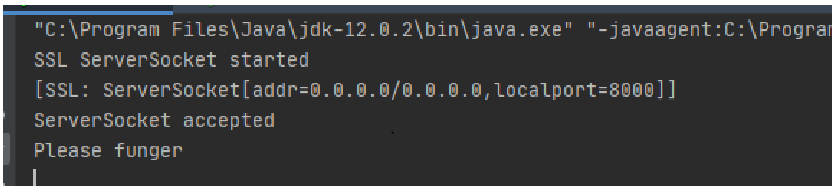


```java
System.setProperty("javax.net.ssl.keyStore", "src\\oblig2\\myKeyStore.jks");
System.setProperty("javax.net.ssl.keyStorePassword", "password");
SSLServerSocketFactory sslServerSocketFactory =
        (SSLServerSocketFactory)SSLServerSocketFactory.getDefault();
```

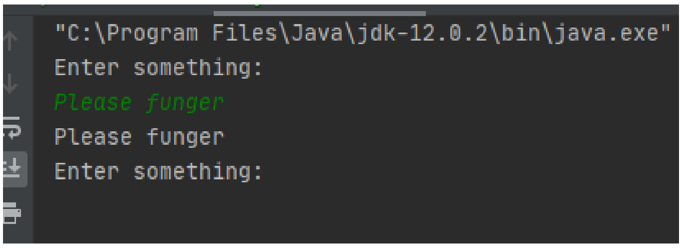


```java
System.setProperty("javax.net.ssl.trustStore", "src\\oblig2\\myTrustStore.jts");
System.setProperty("javax.net.ssl.trustStorePassword", "password");
SSLSocketFactory sslSocketFactory =
        (SSLSocketFactory)SSLSocketFactory.getDefault();
```

Vi prøver igjen. Ser man det, nå blir ServerSocket accepted!



```
"C:\Program Files\Java\jdk-12.0.2\bin\java.exe" "-javaagent:C:\Progra
SSL ServerSocket started
[SSL: ServerSocket[addr=0.0.0.0/0.0.0.0,localport=8000]]
ServerSocket accepted
Please funger
```



```
"C:\Program Files\Java\jdk-12.0.2\bin\java.exe"
Enter something:
Please funger
Please funger
Enter something:
```