

Mock eksamen 1

GitHub link: <https://github.com/HelenaGraff/Mock1-HelenaGraff-1997>

Azure link: https://portal.azure.com/?Microsoft Azure Education correlationId=234e1458-08c0-496b-b499-6c9c6e685875#@edu.easj.dk/resource/subscriptions/0c6ebe46-f222-4706-ab7b-efe5b048ba45/resourceGroups/Mock_1/providers/Microsoft.Web/sites/BilhusetHelenaG/appServices

Opgave 2. b:

På nedenstående billede, kan code coverage over min kode i SKAT projektet ses.

Code Coverage Results					
helen_LAPTOP-G000B3LP 2019-10-23 14_1					
Hierarchy	Not Covered (Blocks)	Not Covered (% Blocks)	Covered (Blocks)	Covered (% Blocks)	
helen_LAPTOP-G000B3LP 20...	2	4,26 %	45	95,74 %	
skat.dll	0	0,00 %	18	100,00 %	
Skat	0	0,00 %	18	100,00 %	
Afgift	0	0,00 %	18	100,00 %	
BilAfgift(int)	0	0,00 %	9	100,00 %	
ElBilAfgift(int)	0	0,00 %	9	100,00 %	
testskat.dll	2	6,90 %	27	93,10 %	
TestSkat	2	6,90 %	27	93,10 %	
UnitTest1	2	6,90 %	27	93,10 %	
TestBilAfgiftFor...	1	14,29 %	6	85,71 %	
TestBilAfgiftOve...	0	0,00 %	3	100,00 %	
TestBilAfgiftUnd...	0	0,00 %	3	100,00 %	
TestElbilAfgiftFo...	1	14,29 %	6	85,71 %	
TestElbilAfgiftO...	0	0,00 %	3	100,00 %	
TestElbilAfgiftU...	0	0,00 %	3	100,00 %	
UnitTest1()	0	0,00 %	3	100,00 %	

Udfra code coverage analyse kan det konkluderes, at min test har god code coverage, da 95,74% af koden er covered. Ligeledes er de resterende 4,26% af koden, som ikke er covered, tuborgklammer, hvorfor koden alt andet lige må have god code coverage.

```

[TestMethod]
0 references
public void TestElbilAfgiftForNegativ()
{
    int pris = -10000;

    try
    {
        afgift.ElBilAfgift(pris);
    }
    catch (Exception e)
    {
        Assert.AreEqual(expected: "Pris må ikke være mindre end eller lig med 0", actual: e.Message);
    }
}

```

Opgave 3. d:

Jeg implementerer en multi-trådet TCP-server ved at implementere et while-loop i min main, i min server, som indeholder en task, som afventer en klient. Jeg har valgt at bruge factory til at gøre min server multi-trådet, da factory opretter en task objekt til hvert objekt, som bliver opfanget.

```
Task.Factory.StartNew(() => afgift.DoIt());
```

Opgave 4. c:

The left screenshot shows a command prompt window titled 'C:\WINDOWS\system32\cmd.exe'. The text displayed is:

```
Klienten er klar
Indtast bil eller elbil
elbil
Server: Indtast prisen på bilen
200000
Server: Afgiften på bilen er: 34000
stop
Tak for denne gang. Tryk enter for at lukke vinduet
```

The right screenshot shows a command prompt window titled 'C:\Program Files\dotnet\dotnet.exe'. The text displayed is:

```
Serveren er startet
Serveren er aktiveret
Klient: elbil
Klient:
Serveren er aktiveret
Klient: elbil
Klient: 200000
Klient: stop
```

Som det fremgår af ovenstående billede, er klienten i kontakt med serveren, samtidig med, at serveren er i kontakt med de metoder, som findes i SKAT, da serveren kan give klienten det korrekte svar på bilens afgift. Ligeledes fremgår det af billederne, at klienten søger en elbil som koster 200000 kr.

The screenshot shows a Wireshark packet capture for interface 127.0.0.1. The packet list table is as follows:

No.	Time	Source	Destination	Protocol	Length	In
11	1.141809	127.0.0.1	127.0.0.1	TLSv1.2	70	Ap
12	1.141888	127.0.0.1	127.0.0.1	TCP	44	51
13	3.053372	127.0.0.1	127.0.0.1	TCP	51	52
14	3.053410	127.0.0.1	127.0.0.1	TCP	44	76
15	3.053679	127.0.0.1	127.0.0.1	TCP	70	76
16	3.053706	127.0.0.1	127.0.0.1	TCP	44	52
17	5.301858	127.0.0.1	127.0.0.1	TLSv1.2	76	Ap
18	5.301940	127.0.0.1	127.0.0.1	TCP	44	88
19	5.421464	127.0.0.1	127.0.0.1	TLSv1.2	72	Ap
20	5.421540	127.0.0.1	127.0.0.1	TCP	44	52
21	6.727071	127.0.0.1	127.0.0.1	TLSv1.2	244	Ap
22	6.727144	127.0.0.1	127.0.0.1	TCP	44	88
23	6.813774	127.0.0.1	127.0.0.1	TLSv1.2	98	Ap
24	6.813861	127.0.0.1	127.0.0.1	TCP	44	52

Below the table, the details of frame 13 are shown:

```
> Frame 13: 51 bytes on wire (408 bits), 51 bytes captured (408 bits) on interface
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 52602, Dst Port: 7000, Seq: 1, Ack: 1
```

At the bottom, the raw packet data is shown in hexadecimal and ASCII:

```
0000 02 00 00 00 45 00 00 2f 17 b3 40 00 80 06 00 00 ...E../ ..@.....
0010 7f 00 00 01 7f 00 00 01 cd 7a 1b 58 87 7d 64 bd .....z.X}d-
0020 d0 e4 7a c4 50 18 27 f9 2b 30 00 00 65 6c 62 69 ..z.P.' +0-elbi
0030 6c 0d 0a 1...
```

Som det kan ses på dette billede, opfanger WireShark at klienten indtaster "elbil". Og ligeledes fremgår det af de to nedenstående billeder, at Wireshark også fanger at klienten indtaster bilens pris, samt at serveren returnerer bilens afgift.

127.0.0.1					
No.	Time	Source	Destination	Protocol	Length
14	3.053410	127.0.0.1	127.0.0.1	TCP	4
15	3.053679	127.0.0.1	127.0.0.1	TCP	7
16	3.053706	127.0.0.1	127.0.0.1	TCP	4
17	5.301858	127.0.0.1	127.0.0.1	TLSv1.2	7
18	5.301940	127.0.0.1	127.0.0.1	TCP	4
19	5.421464	127.0.0.1	127.0.0.1	TLSv1.2	7
20	5.421540	127.0.0.1	127.0.0.1	TCP	4
21	6.727071	127.0.0.1	127.0.0.1	TLSv1.2	24
22	6.727144	127.0.0.1	127.0.0.1	TCP	4
23	6.813774	127.0.0.1	127.0.0.1	TLSv1.2	9
24	6.813861	127.0.0.1	127.0.0.1	TCP	4
25	6.814435	127.0.0.1	127.0.0.1	TLSv1.2	289
26	6.814509	127.0.0.1	127.0.0.1	TCP	4
27	6.942051	127.0.0.1	127.0.0.1	TCP	5

> Frame 27: 52 bytes on wire (416 bits), 52 bytes captured (416 bits) on interface
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 52602, Dst Port: 7000, Seq: 8, Acc

```
0000 02 00 00 00 45 00 00 30 17 c1 40 00 80 06 00 00  ....E..0 ..@.....
0010 7f 00 00 01 7f 00 00 01 cd 7a 1b 58 87 7d 64 c4  ....z.X.}d.
0020 d0 e4 7a de 50 18 27 f9 c9 56 00 00 32 30 30 30  ..z.P.'..V..2000
0030 30 30 0d 0a                                00..
```

127.0.0.1					
No.	Time	Source	Destination	Protocol	Length
18	5.301940	127.0.0.1	127.0.0.1	TCP	4
19	5.421464	127.0.0.1	127.0.0.1	TLSv1.2	7
20	5.421540	127.0.0.1	127.0.0.1	TCP	4
21	6.727071	127.0.0.1	127.0.0.1	TLSv1.2	24
22	6.727144	127.0.0.1	127.0.0.1	TCP	4
23	6.813774	127.0.0.1	127.0.0.1	TLSv1.2	9
24	6.813861	127.0.0.1	127.0.0.1	TCP	4
25	6.814435	127.0.0.1	127.0.0.1	TLSv1.2	289
26	6.814509	127.0.0.1	127.0.0.1	TCP	4
27	6.942051	127.0.0.1	127.0.0.1	TCP	5
28	6.942081	127.0.0.1	127.0.0.1	TCP	4
29	6.942519	127.0.0.1	127.0.0.1	TCP	7
30	6.942535	127.0.0.1	127.0.0.1	TCP	4
31	6.974362	127.0.0.1	127.0.0.1	TLSv1.2	33

> Frame 29: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 7000, Dst Port: 52602, Seq: 27, Acc

```
0000 02 00 00 00 45 00 00 46 17 c3 40 00 80 06 00 00  ....E..F ..@.....
0010 7f 00 00 01 7f 00 00 01 1b 58 cd 7a d0 e4 7a de  ....z.X.z..z.
0020 87 7d 64 cc 50 18 27 f9 fc 90 00 00 41 66 67 69  ..}d.P.'..Afgi
0030 66 74 65 6e 20 70 c3 a5 20 62 69 6c 65 6e 20 65  ften p.. bilen e
0040 72 3a 20 33 34 30 30 30 0d 0a                                r: 34000 ..
```