Eksamen B

I fagene

Programmering og Teknik

2020, juni, kl. 9.00 - 14.00

Alle skriftlige materialer, pc'er, bærbare computere og internet ressourcer er tilladt til eksamen.

Mobiltelefoner og kommunikation med andre personer (inkl. chatbots), bortset fra kommunikation med eksaminator og censor, er forbudt.

Du må ikke gemme dine løsninger på eksterne netværksdrev/værter som GitHub, Facebook o. lign. Overtrædelse af dette vil medføre bortvisning fra eksamen og passende sanktioner vil senere blive tildelt både til afsender/uploader og modtager.

Ved eksamens afslutning skal du lægge din løsning op i Wiseflow.

Eksamen varer 4 timer og efterfølges af 1 times evaluering. I disse tidsrum kan du kun forlade eksamenslokalet med henblik på toilet besøg. Rygning er ikke tilladt.

Eksamenssættet består af 8 opgaver.

Ud over disse opgaver, kan du kan blive bedt at besvare nogle ekstra spørgsmål omkring dine svar og eventuelle andre valg.

Kontroller at dette sæt indeholder 4 sider inklusive forsiden.

# Introduktion

Du skal programmere et system der på forskellig vis kan konvertere beløb mellem diverse valutaer.

* 1 svensk krone = 0.7041 dansk krone
* 1 dansk krone = 1.0/0.7041 svensk krone

Udover at programmere dette system, er der også teoretiske spørgsmål undervejs, hvor du bliver bedt om at redegøre for overvejelser og nedskrive forklaringer. Dette skal du skrive i et dokument, som uploades til Wiseflow sammen med dine programmer i en zip fil.

# Opgave 1 Class library

1. Du skal lave et projekt af typen Class Library (.NET core).
Dit projekt skal indeholde en klasse med 2 simple *static* metoder
 public double TilSvenskeKroner(double danskeKroner)
 public double TilDanskeKroner(double svenskeKroner)
2. Du skal lave en unit test af din klasse.

# Opgave 2 TCP-server

Du skal nu gøre brug af en TCP Server, som kan tilbyde de to udregninger som er udviklet i klassen i opgave 1 (*TilSvenskeKroner, TilDanskeKroner*). Dertil skal der designes en brugerdefineret protokol. Protokollen skal kunne skelne mellem de to udregninger og protokollen skal overføre de nødvendige tal-informatoner for de to udregninger. Protokollen skal derfor kunne:

1. Modtage hvilket udregning der ønskes: “*TilSvenskeKroner”* eller *“TilDanskeKroner”* Derudover modtager en beløb - Derefter
2. Udregn konvertering afhængig af den ønskede konvertering.
3. Sende resultatet af udregningen tilbage til klienten.

Det kan anbefales først at lave en meget simpel TCP Server (denne opgave – Opgave 2), som kan testes med SocketTest tool og en meget simpel TCP Client (Opgave 3). Herefter kan programmet (programmerne) udvides sideløbende.
 Din TCP Server må meget gerne fra start af udnytte/udstille dine metoder i Library, men det er ikke et krav i første omgang.

1. På denne daggrund, overvej og design en meget simple protokol, dvs. Hvordan du adskiller om udregningen er “*TilSvenskeKroner” eller “TilDanskeKroner”,* samt hvordan data overføres.
2. Lav en simple Consol Application projekt. Der gør brug af en TcpListener og TcpClient sockets, og som lytter (listening) på port f.eks. 7000.
3. Udvid programmet til at:
	* Læse beskederne (defineret ved din protokol) fra klienten og udskrive beskeden.
	* Fortolke beskederne og udregne det ønskede resultat.
	* Sende et svar med det udregnede resultat tilbage til klienten.
	* Test dit program med SocketTest
4. Din TCP-server skal nu kunne bruge dit Library og derfor skal du skal nu udstille disse metoder i dette projekt. Du skal bruge din DLL fra opgave 1.
 Udnyt derefter metoderne i dit Library til udregningen af resultatet, hvis du ikke allerede har gjort det.
5. Hvis du ikke har programmeret en multi-trådet (concurrent) TCP-server, skal du nu redegøre for hvordan dette kunne gøres.
Udvid derefter programmet, så TCP-serveren kan håndtere hver klient i en separat task/tråd.

# Opgave 3 TCP client

Lav et Console Application projekt, som kan forbinde til din TCP Server og som kan sende beskeder til Serveren.

1. Vis hvordan man kan sende beskeder til serveren, modtage og udskrive serverens svar.
2. Hvad betyder det, at TCP er en forbindelsesorienteret (connection oriented) protokol?
Hvordan kan et TCP-segment finde vej til den korrekte modtager?

# Opgave 4 REST service provider

1. Du skal lave en REST service provider.
Controller-klassen skal have 2 metoder svarende til de 2 metoder i dit class library.
Du skal bruge dit class library og anvende metoderne herfra til udregningerne.
Hvilken HTTP metode (GET, POST, PUT osv.) er passende til dine metoder?
Hvilke routes (URI’er) har du valgt til dine metoder?
2. Kør din service lokale (*localhost*).
Afprøv din service med en browser og/eller Postman.
3. Publicer din REST service til Microsoft Azure.
4. Indstil din REST service til at understøtte CORS.
Det kan gøres på flere måder. Du bestemmer selv hvilken måde du vil benytte.

# Opgave 5 Web applikation

Lav en web-applikation med HTML, Typescript og Vue.js.

Brugeren vælger om der skal konverteres fra danske til svenske kroner, eller vice versa.
Brugeren indtaster et beløb.
Web-applikationen skal bruge din REST service til at udregne beløbet i den ønskede valuta.

# Opgave 6 Web-applikation i Azure

Publicer din web-applikation til Azure.

Vis i en browser at applikationen virker.

# Opgave 7 UDP server og UDP client

1. Lav en Udp server som kan tilbyde udregningerne (“*TilSvenskeKroner” og “TilDanskeKroner”*)
2. Test Udp-serveren med SocketTest program.
3. Forklar forskellen mellem en TCP-forbindelse og en UDP-forbindelse.

# Opgave 8: REST service provider, udvidet

1. Indtil videre har der været en fast valutakurs mellem danske og svenske kroner.
Du skal udvide din REST service provider med en metode, der kan ændre valutakursen mellem danske og svenske kroner.
Hvilken HTTP metode (GET, POST, PUT osv.) er passende til den tilføjede metode?
Hvilke route (URI) har du valgt til den tilføjede metoder?
2. Kør din service lokale (*localhost*).
Afprøv din service med en browser og/eller Postman.