# The Three Cases

****

Daniel Villena Askgaard

&

Magnus Rasch

Data og kommunikation

Indholdsfortegnelse

[The Three Cases 1](#_Toc97807912)

[Indledning 3](#_Toc97807913)

[Spørgsmål og svar 4](#_Toc97807914)

[**• Hvad er et Entry point og hvor mange kan man have i koden:** 4](#_Toc97807915)

[**• Hvad kalder man en variable der er erklæret ved hjælp at ordet ”var”:** 4](#_Toc97807916)

[**o Og hvornår man skal initialiser variablen? – husk begrundelse:** 4](#_Toc97807917)

[**• Dokumenter og forklar hvad hver del gør i initialisering af et loop og hvordan du afvikles. Gør dette for alle loop funktioner.** 4](#_Toc97807918)

[**o while, doWhile, whileDo, foreach and for loop:** 4](#_Toc97807919)

[**• Hvad er for forskellen på ++x og x++:** 5](#_Toc97807920)

[**• Hvad betyder det når noget er null, og hvorfor kan vi ikke lide det:** 5](#_Toc97807921)

[**• Hvornår er string interpolation operator ($) en fordel at bruge:** 5](#_Toc97807922)

[**o Og hvornår er StringBuilder en fordel at bruge:** 5](#_Toc97807923)

[**o Hvorfor skal man være opmærksom på det?** 6](#_Toc97807924)

[**• Hvad er forskelle på value types og reference types** 6](#_Toc97807925)

[**o Hvordan fungere en Value type og hvordan fungere en reference type** 6](#_Toc97807926)

[**o Herunder hvad der skeer når du kopier et objekt (reference types) og når du kopier en værdi (Value type)** 6](#_Toc97807927)

[**• Hvad er Encapsulation / Information hiding og hvorfor ønsker vi at gøre det?** 6](#_Toc97807928)

[**• Forklar de forskellige dele der til sammen skaber en metode. Først generelt og derefter med udgangspunkt i denne metode:** 6](#_Toc97807929)

[**o Forklar hvad der returneres:** 7](#_Toc97807930)

[**• Hvad ligger der bag begrebet ”Operator Overloading”:** At man bruger flere operatører til en type. 7](#_Toc97807931)

[**• Hvad gør en Destructor og hvorfor anvendes den:** En ”Destructor” bruger man til at nedlægge et bestemt tilfælde inde for en klasse. 7](#_Toc97807932)

[**o hvorfor bruger udvikler den ikke særligt ofte i C# kode:** 7](#_Toc97807933)

[**• Hvad er Regular Expressions (RegEx) og hvordan kan man bruge regex:** 7](#_Toc97807934)

[**• Hvis du har flere projekter, så skal du forklar hvad en cirkulær reference er:** 7](#_Toc97807935)

[Konklusion 8](#_Toc97807936)

[Opsætning af kode 9](#_Toc97807937)

[Main program del 9](#_Toc97807938)

[Login del 10](#_Toc97807939)

[Fodbold del 1 11](#_Toc97807940)

[Fodbold del 2 12](#_Toc97807941)

[Dans del 1 **Fejl! Bogmærke er ikke defineret.**](#_Toc97807942)

[Dans del 2 **Fejl! Bogmærke er ikke defineret.**](#_Toc97807943)

[Password del 1 15](#_Toc97807944)

[Password del 2 16](#_Toc97807945)

# Indledning

Vi har fået stillet en opgave ved navn ”The Three Cases”. Opgaven går ud på at vi har fået 3 cases, som vi skal løse i Visual Studio. Hver af cases skal udvikles på følgende måde at vi skal skrive afkoblede kode, som er genanvendelige og uafhængig af alt andet kode. Vi skal samle casene, så vi har et stort program til sidst. Når man starter programmet ind skal man lave en kode og brugernavn, vis alle kravende bliver opfyldt bliver man sendt ind i menuen, hvor man kan vælge hvilken case man vil ind i.

I denne opgave har vi brugt:

* Youtube
* Visuel Studio
* Google search engine

# Spørgsmål og svar

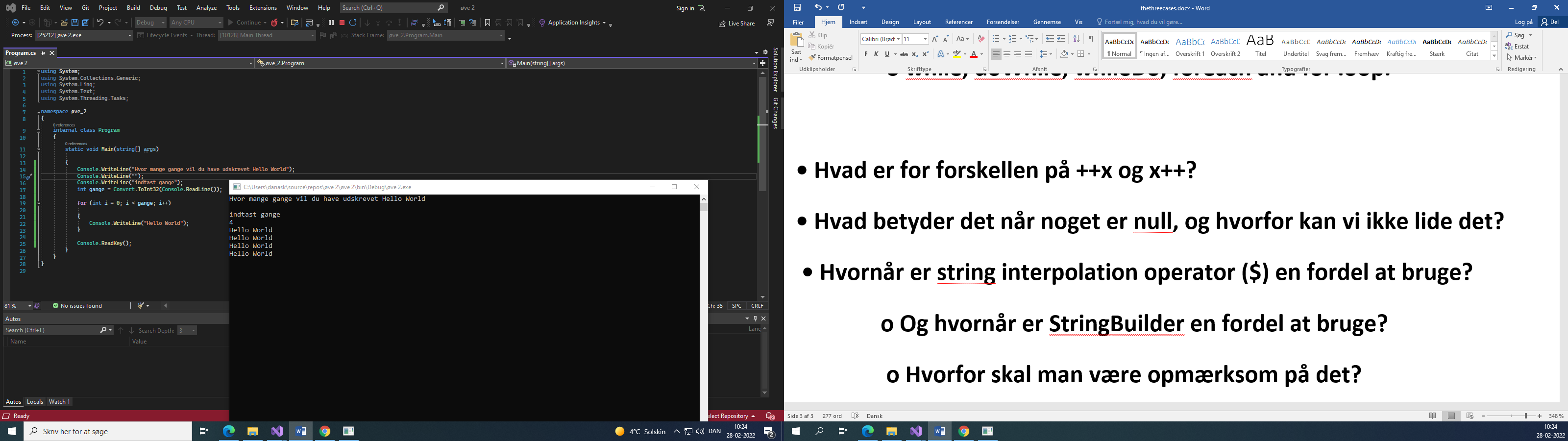
**• Hvad er et Entry point og hvor mange kan man have i koden:** et entry point er et sted i koden hvor en overførsel af programstyring (udførelse) finder sted.

**• Hvad kalder man en variable der er erklæret ved hjælp at ordet ”var”:** var finder slev ud af datetypen som fx er int og string

## **o Og hvornår man skal initialiser variablen? – husk begrundelse:**

det kan man bruge i stedet for fx int oh string så finder den selv af det

## **• Dokumenter og forklar hvad hver del gør i initialisering af et loop og hvordan du afvikles. Gør dette for alle loop funktioner.**

**** **o while, doWhile, whileDo, foreach and for loop:** for den gentager så mange gange brugeren indtaster

## **• Hvad er for forskellen på ++x og x++:**

**’++x’ er Prefix:** At sige ++x betyder at øge (++) først og derefter returnere værdien af ​​x.

**’x++’ er Postfix:** postfix returnere værdien af en variabel før den er blevet forøget

**• Hvad betyder det når noget er null, og hvorfor kan vi ikke lide det:** Null er i programmering- og database-sammenhæng et udtryk for en værdi, der hverken er sand eller falsk, tom eller ikke defineret

**• Hvornår er string interpolation operator ($) en fordel at bruge:** string er når du gir en værdi

men det kan kun værre tekst.

**o Og hvornår er StringBuilder en fordel at bruge:** det er en fordel, fordi man kan modificere Stringbuilder.

#### **String Example:**

string colors;

colors += "red";

colors += "blue";

colors += "green";

#### **StringBuilder Example:**

StringBuilder sb = new StringBuilder("");

sb.Append("red");

sb.Append("blue");

sb.Append("green ");

string colors = sb.ToString();

## **o Hvorfor skal man være opmærksom på det?**

Man skal være opmærksom i forhold til at man vil bruge mindre resurser i ens program hvilket altid er ideen bag det et velfungerende program.

## **• Hvad er forskelle på value types og reference types**

En ”value type” indeholder data inde i dens egen hukommelses tildeling. ”Reference type” gemmer på en reference til dens egen data.

## **o Hvordan fungere en Value type og hvordan fungere en reference type**

”value type” bruger man til bestemme en type for noget data. ”Reference type” bruger man til at refererer til en bestemt data type eller en bestemt klasse i programmet.

## **o Herunder hvad der skeer når du kopier et objekt (reference types) og når du kopier en værdi (Value type)**

Hvis man kopiere en ”value type” eller ”reference type” så risikere man at der opstår en fejl i programmet for der kan ikke være den samme definerede værdi i et program.

## **• Hvad er Encapsulation / Information hiding og hvorfor ønsker vi at gøre det?**

”Encapsulation” er en metode hvor man kan undgå en uønsket data korruption, og man gør også på samme tid sådan så programmet ikke bliver nemt manipuleret med af de kommende brugere af programmet.

## **• Forklar de forskellige dele der til sammen skaber en metode. Først generelt og derefter med udgangspunkt i denne metode:**

”private static” er returtypen, ”MyMethod” er navnet og alt hvad der inde i tuborgklammen er kroppen for metoden.



<access level><valg af retur type><Navngivning af metode><Input paramter> <tuborgklamme><return><returtype>< array med tre pladser >< output paramter>< tuborgklamme>

## **o Forklar hvad der returneres:**

Den returnere ”x, 20 og null”.

## **• Hvad ligger der bag begrebet ”Operator Overloading”:** At man bruger flere operatører til en type.

## **• Hvad gør en Destructor og hvorfor anvendes den:** En ”Destructor” bruger man til at nedlægge et bestemt tilfælde inde for en klasse.

## **o hvorfor bruger udvikler den ikke særligt ofte i C# kode:**

Fordi det ikke er optimalt i et program. Der er bedre måder at gøre det på.

## **• Hvad er Regular Expressions (RegEx) og hvordan kan man bruge regex:**

Den repræsentere en ikke modificer bar regulært udtryk. Den tjekker efter gentagelser.

## **• Hvis du har flere projekter, så skal du forklar hvad en cirkulær reference er:**

Cirkulær reference opstår, når to eller flere indbyrdes afhængige ressourcer forårsager låsetilstand.

# Konklusion

Med denne her opgave har vi lært hvad ”class library” gør og hvad man skal bruge det til når man laver programmer. Vi har lært nogle grundlæggende ting om metoder og klasser, og vi har lært en masse ny forskellig kode som blandt andet er: stringbuilder, operator overload, encapsulation og destructor. Vi synes at det er nogle rigtige gode værktøjer til at bygge fremtidige programmer som kan udrette store ting.

# Opsætning af kode

## Main program del

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Tre\_Cases

{

internal class Program

{

static void Main(string[] args)

{

string indtast, brugernavn\_login, adgangskode\_login;

Console.WriteLine("Starten af programmet");

Console.ReadKey();

Console.WriteLine("Indtast brugernavn: ");

brugernavn\_login = Console.ReadLine().ToLower();

Console.WriteLine("Indtast adgangskode: ");

adgangskode\_login = Console.ReadLine();

///Ovenfor kan man se starten af programmet. Det er det alleførste man ser inden man for tilladelse til a komme ind på de

///forskellige cases.

ClassLibrary1.Program\_login Program\_login\_Objekt = new ClassLibrary1.Program\_login();

string Program\_loginResultat = Program\_login\_Objekt.LoginSåDuKanGåIndICases(brugernavn\_login, adgangskode\_login);

Console.WriteLine(Program\_loginResultat);

Console.ReadKey();

///Linjerne ovenfor bliver brugt til at gå ind i login casen for at tjekke om man har indtastet de rigtige informationer

///til at kunne blive godkendt af programmet for at få adgang til de andre cases.

if (Program\_loginResultat == "Login er godkendt") ///Det er til hvis man bliver godkendt.

{

do

{

///Menuen til at vælge de andre cases.

Console.Clear();

Console.WriteLine("Vælg en case");

Console.WriteLine("");

Console.WriteLine("Fodbold");

Console.WriteLine("");

Console.WriteLine("Dans");

Console.WriteLine("");

Console.WriteLine("Password");

Console.WriteLine("");

Console.WriteLine("Sluk");

Console.Write(": ");

indtast = Console.ReadLine().ToLower();

if (indtast == "fodbold")

{

var OpeningAfFodbold = new Fodbold();

OpeningAfFodbold.Run();

}

///Ovenfor er der koden til at kunne gå ind i fodbold casen og det samme gælder for de nedenstående koder indtil "While" linjen.

else if (indtast == "dans")

{

var OpeningAfDans = new Dans();

OpeningAfDans.Run();

}

else if (indtast == "password")

{

var OpeningAfPassword = new Password();

OpeningAfPassword.Run();

}

} while (indtast != "sluk");

}

else

{

Console.WriteLine("Login mislykkedes. Nu lukker programmet");

Console.ReadKey();

}

///En fejlmeddelelse

}

}

}

## Login del

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ClassLibrary1

{

public class Program\_login

{

public string LoginSåDuKanGåIndICases(string brugernavn\_login, string adgangskode\_login)

{

var Login\_Resultat = HvisLogin(brugernavn\_login, adgangskode\_login);

if (!string.IsNullOrEmpty(Login\_Resultat))

{

return Login\_Resultat;

}

else

{

return "Fejl indtast";

}

}

private string HvisLogin(string brugernavn\_login, string adgangskode\_login)

{

string bruger = System.IO.File.ReadAllText(@"C:\Users\magras\Desktop\Opgave\bruger.txt"); // dette er koden som læser det som står inde i stien det gælder også for den nedenstående kode

string adgangskode = System.IO.File.ReadAllText(@"C:\Users\magras\Desktop\Opgave\adgangskode.txt");

if (bruger == brugernavn\_login && adgangskode == adgangskode\_login) // Denne kode tjekker om brugernavn og kode er korrekt

{

return "Login er godkendt"; // hvis koden er korrekt så returner den teksten

}

else // fejlmeddelelse

{

return "Forkert brugernavn eller kode";

}

}

}

}

## Fodbold del 1

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Tre\_Cases

{

internal class Fodbold

{

internal void Run()

{

string Mål, input;

int Afleveringer;

// menu

Console.Clear();

Console.WriteLine("Velkommen til Fodbold case");

Console.ReadKey();

Console.WriteLine("Skriv (mål) hvis der er mål: ");

Mål = Console.ReadLine().ToLower();

Console.WriteLine("Skriv antal afleveringer: ");

input = Console.ReadLine();

bool StringProof = int.TryParse(input, out Afleveringer); // Denne linje bliver brugt til at forhindre fejl ved at trykke på enter uden at skrive noget i feltet først.

if (Afleveringer >= 0)

{

ClassLibrary1.Fodbold Fodbold\_Objekt = new ClassLibrary1.Fodbold();

string FodboldResultat = Fodbold\_Objekt.ErDerMålOgAfleveringer(Mål, Afleveringer); // denne kode tager den ind i logikken

Console.WriteLine(FodboldResultat);

}

else

{

Console.WriteLine("Fejl indtast");

Console.ReadKey();

}

Console.ReadKey();

}

}

}

## Fodbold del 2

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ClassLibrary1

{

public class Fodbold

{

public string ErDerMålOgAfleveringer(string Mål, int Afleveringer)

{

var Resultat\_string = HvisMål(Mål);

if (!string.IsNullOrEmpty(Resultat\_string))

{

return Resultat\_string;

}

else

{

return HvisAflevering(Afleveringer);

}

}

private string HvisMål(string Mål)

{

if (Mål == "mål") // hvis der er mål udskriver den "Olé Olé Olé Olé"

{

return "Olé Olé Olé Olé";

}

return string.Empty;

}

private string HvisAflevering(int Afleveringer)

{

if (Afleveringer < 1)

{

return "shh"; // mindre end 1 aflevering udskiver den "shh"

}

else if (Afleveringer >= 1 && Afleveringer < 10) // hvís der mere eller lig med 1 og mindre en 10 udskriver den "Huh"

{

var resultat = "";

for (int i = 0; i < Afleveringer; i++)

{

resultat += "Huh! ";

}

return resultat;

}

else

return "High Five - Jubel!!!"; // hvis der er mere end 10 afleveringer eller mere

}

}

}

## Dans del 1

public class Danse

{

public void Run()

{

Console.Clear();

//Indtast danserens navn

Console.WriteLine("Indtast Danser 1 Navn: ");

string name = Console.ReadLine();

//Indtast points

Console.WriteLine("Indtast point: ");

int points1 = Convert.ToInt32(Console.ReadLine());

//Indtast danser 2 navn

Console.WriteLine("Indtast danser2 navn: ");

string name2 = Console.ReadLine();

//Indtast danser 2 points

Console.WriteLine("Indtast point: ");

//den laver talene om til text

int points2 = Convert.ToInt32(Console.ReadLine());

Danse\_class dans1 = new Danse\_class(name, points1);

Danse\_class dans2 = new Danse\_class(name2, points2);

// ligger total af begge classer sammen

Danse\_class total = dans1 + dans2;

//Cleans the screen so it can look pretty

Console.Clear();

//Shows it in the screen

Console.WriteLine("Danser1: {0} {1}", name, points1);

Console.WriteLine("Danser2: {0} {1}", name2, points2);

Console.WriteLine("indtast for at summere points");

Console.Clear();

Console.WriteLine(total.navn);

Console.WriteLine(total.point);

Console.ReadKey();

}

}

}

# Dans del 2

namespace LogicCases

{

public class Danse\_class

{

public string navn;

public int point;

//A string inside DancePoints that equals name and points

public Danse\_class(string navn, int point)

{

this.navn = navn;

this.point = point;

}

public static Danse\_class operator +(Danse\_class a, Danse\_class b)

{

string navne = a.navn + " & " + b.navn;

int points = a.point + b.point;

return new Danse\_class(navne, points);

}

}

}

## Password del 1

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Tre\_Cases

{

internal class Password

{

internal void Run() // internal void Run den starter case inde fra hovedprogrammet

{

string brugernavn, kodeord;

// dette er menuen

Console.Clear();

Console.WriteLine("Velkommen til Password case");

Console.ReadKey();

Console.Clear();

Console.WriteLine("Indtast brugernavn");

brugernavn = Console.ReadLine().ToLower();

Console.WriteLine("Indtast kodeord");

kodeord = Console.ReadLine();

ClassLibrary1.Password Password\_Objekt = new ClassLibrary1.Password();

string PasswordResultat = Password\_Objekt.BrugernavnOgKodeord(brugernavn, kodeord); // den kode sørger for at vi kommer ind i logikken

Console.WriteLine(PasswordResultat);

Console.ReadKey();

}

}

}

## Password del 2

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ClassLibrary1

{

public class Password

{

public string BrugernavnOgKodeord(string brugernavn, string kodeord)

{

var Login\_Resultat = HvisGodkendt(brugernavn, kodeord);

if (!string.IsNullOrEmpty(Login\_Resultat))

{

return Login\_Resultat;

}

else

{

return "Fejl indtast";

}

}

private string HvisGodkendt(string brugernavn, string kodeord)

{

string MinFejlMeddelelse = "Kodeordet skal mindst være 12 tegn langt, der skal anvendes både store og små bogstaver, et tal mindst, benyttes mindst et specialtegn, tal må ikke være i starten og slutningen af koden, og der må ikke være mellemrum";

bool ErDerMindstEtStortBogstav = kodeord.Any(char.IsUpper); // mindst have et stort bogstav

bool ErDerMindstEtSmåtBogstav = kodeord.Any(char.IsLower);// mindst et småt bogstav

bool ErDerEtSpecialtegn = kodeord.Any(ch => !char.IsLetterOrDigit(ch)); // mindst et specielt teng

bool ErDerEtTalMindst = kodeord.Any(char.IsDigit); // mindst et tal

bool DerMåIkkeVæreTalIStarten = !string.IsNullOrEmpty(kodeord) && char.IsLetter(kodeord[0]); // ingen tal i starten

bool DerMåIkkeVæreTalISlutningen = !string.IsNullOrEmpty(kodeord) && char.IsLetter(kodeord.LastOrDefault()); // ingen tal i slutningen

bool ErDerMellemrum = kodeord.Contains(" "); // ingen mellemrum

if (ErDerMindstEtStortBogstav == false || ErDerMindstEtSmåtBogstav == false || ErDerEtTalMindst == false || ErDerEtSpecialtegn == false || DerMåIkkeVæreTalIStarten == false || DerMåIkkeVæreTalIStarten == false || ErDerMellemrum == true)

// vis en af kravene ikke opfyldt giver den er en fejlmeddelelse

{

return MinFejlMeddelelse; // den returner string

}

else

{

string brugerfil = @"C:\Users\magras\Desktop\Opgave\bruger.txt"; // det er en string til en sti

string adgangskodefil = @"C:\Users\magras\Desktop\Opgave\adgangskode.txt";

System.IO.File.WriteAllText(brugerfil, brugernavn); // disse to koder overskriver alt der er derinde som er inde i tekstdokumentet

System.IO.File.WriteAllText(adgangskodefil, kodeord);

return brugernavn + " " + kodeord; // viser hvad brugernavn og kodeord

}

}

}

}