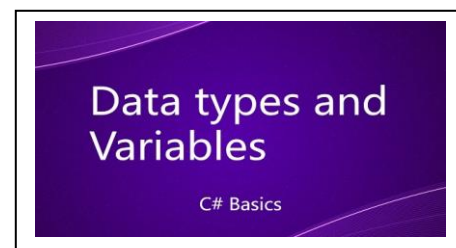


02 – Variabler og Datatyper

S1



Opgavens fokusområder og læringsmål

- ☐ Du kan redegøre for hvad en variabel er
- ☐ Du kan redegøre for de i C# indbyggede datatyper, samt Date og String
- ☐ Du kan demonstrere anvendelsen af variabler og datatyper
- ☐ Du kan løse opgaver hvor du anvender og modificere variabler og deres data
- ☐ Du kan skelne mellem de enkelte datatyper og konkludere hvilken datatype der er bedst egnet

Kort fortalt

Dette er introduktionen til begreberne *variabler* og *datatyper* i C#.

Begrebet *variabel* går igen i mange andre programmeringssprog og vil have samme funktion.

I andre programmeringssprog vil *datatyper* måske være anderledes end i C#, men grundlæggende vil du finde de samme *datatyper* i alle programmeringssprog.

Praktiske oplysninger

Materialet er baseret på Visual Studio 2022 og .NET Framework 4.8

Ressourcer

Hvis du ikke har Visual Studio 2022 installeret, kan den hentes og installeres ved at følge denne guide - [LINK](#)

Variabler

I programmeringssproget C# er en variabel en navngiven "beholder", der bruges til at lagre data af forskellige typer, såsom tal, tekst eller objekter.

Variable i C# er grundlæggende enheder, der bruges til at gøre arbejdet med data i et computerprogram nemmere og hurtigere.

I den ovenstående tekst, er en variabel beskrevet som en "beholder" – det er en måde at beskrive en variabel på. Jeg har mange beskrivelser når jeg har læst bøger om programmering: Container, Kasse og flere andre mere eller mindre egnede begreber der skulle gøre det nemmere at forstå hvad en variabel er.

Jeg vil nu prøve at forklare det med mine egne ord.

En variabel i dit C# program, bliver oprettet af dig som skriver koden.

En variabel består grundlæggende af to ting:

- Navn - et sigende navn/ord som fortæller hvilken data den holder
- Datatype – hvilken form for data kan denne variabel indeholde (vi kommer til Datatyper om lidt)



Det er de to ting du som programmør skal angive, ved at du skriver det i din C# kode. Når din kode omsættes til noget der kan afvikles på din PC, så bliver der reserveret plads i arbejdshukommelsen på din PC, så der altid vil være plads til både den mindste og den største værdi din variabel kan indeholde.

Det navn du giver din variabel, er kun til glæde for dig og andre programmører, så man har nemmere ved at skille de forskellige variabler fra hinanden og får et bedre overblik over hvor hvilke data er gemt. Når du starter dit program, bliver alt hvad du har skrevet i dit program lavet om til maskinkode, og dine variabelnavne laves om til "adresser" i Pc'ens RAM-lager.

En variabels indhold eksisterer kun så længe dit program kører.

Navngivning af variabler

Når du skriver din kode, vil din kode efterhånden blive mere og mere kompleks efterhånden som du bliver mere erfaren og skal lave kode der løser større og mere krævende opgaver.

Derfor er yderst vigtigt med et standardiseret regelsæt for hvordan man navngiver sine variabler. Det vil gøre det nemmere at identificere variablen og samtidig sige noget om hvad den indeholder.

Når vi taler om navngivning taler man overordnet om to typer:

- **camelCase** – Navnet starter med en *lowercase* karakter og hvert nyt ord i navnet angives med en *uppercase* karakter.
- **PascalCase** - Navnet starter med en *uppercase* karakter og hvert nyt ord i navnet angives med en *uppercase* karakter.

Når du navngiver din variabel, kan det være nyttigt at bruge flere ord, for at lave et beskrivende navn for variablen. Herunder kan du se et par eksempler:

- **ansatFuldeNavn** – Gennem navngivningen, angiver du at denne variabel indeholder det fulde navn for en ansat i et firma, du laver et program til.
- **kundeFornavn** - Gennem navngivningen, angiver du at denne variabel indeholder fornavnet på en kunde der er knyttet til det firma du laver et program til.
- **kundeAdresse** - Gennem navngivningen, angiver du at denne variabel indeholder adressen på en kunde der er knyttet til det firma du laver et program til.

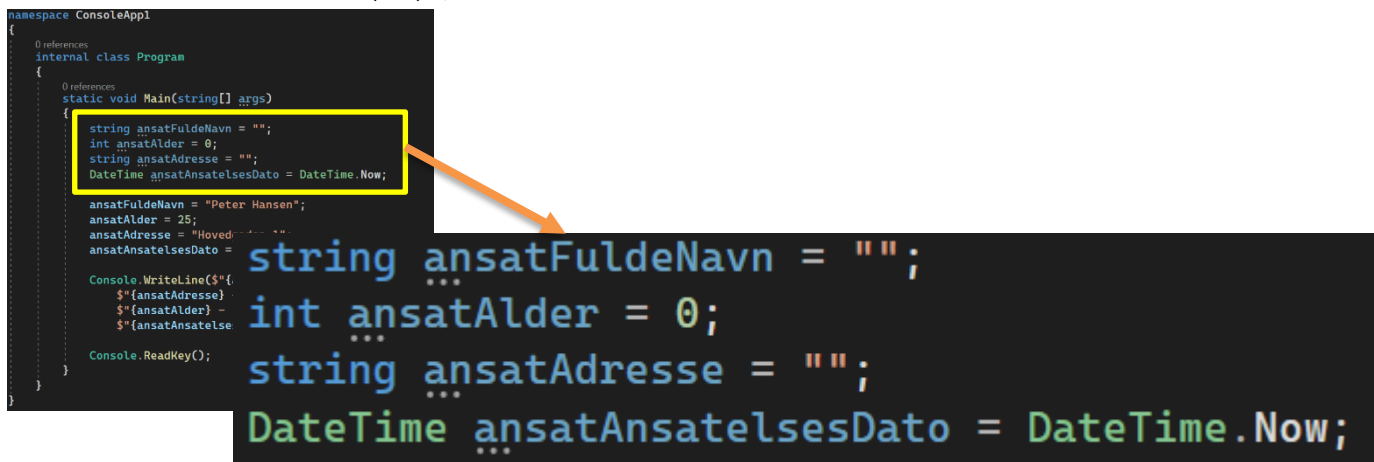
Det er **god programmeringsskik**, at benytte *camelCase* i forbindelse med navngivning af variabler.

Datatyper

Programmeringssproget C# er af en type der kaldes *Strongly Typed*.

Det betyder, at man kun kan tildele værdier til en variabel, der stemmer overens med den datatype som den er erklæret med – Hvad betyder det?

For at definere en variabel i C#, skal du angive dens datatypen, give den et navn og eventuelt initialisere den med en værdi. Her er et eksempel på, hvordan du kan definere en variabel i C#:



```

namespace ConsoleApp1
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            string ansatFuldeNavn = "";
            int ansatAlder = 0;
            string ansatAdresse = "";
            DateTime ansatAnsattelsesDato = DateTime.Now;

            ansatFuldeNavn = "Peter Hansen";
            ansatAlder = 25;
            ansatAdresse = "Hovedvej 123";
            ansatAnsattelsesDato = DateTime.Now;

            Console.WriteLine($"{
                $"{ansatAdresse}
                $"{ansatAlder} -
                $"{ansatAnsattelsesDato}
            Console.ReadKey();
        }
    }

```

string ansatFuldeNavn = "";

int ansatAlder = 0;

string ansatAdresse = "";

DateTime ansatAnsattelsesDato = DateTime.Now;

Som du kan se, starter man altid med at angive *datatypen* og herefter *variabelnavnet*.

I dette eksempel sker der også det man kalder *Initialisering* af *variablen*. Det betyder, at man giver *variablen* en startværdi, da den ellers vil være i en tilstand der kaldes *null*.

Da man skal undgå *variabler* med værdien *null*, er det ligeledes god *programmeringsskik* at man *initialisere* sine variabler når man *erklærer* (opretter) dem.

C# understøtter forskellige datatyper, som kan opdeles i kategorier:

Heltal		
byte	8-bit unsigned heltal	0 – 255
sbyte	8-bit signed heltal	-128 – 127
short	16-bit signed heltal	-32.768 – 32.767
ushort	16-bit unsigned heltal	0 – 65.535
int	32-bit signed heltal	-2.147.483.647 – 2.147.483.647
uint	32-bit unsigned heltal	0 – 4.294.967.295
long	64-bit signed heltal	-9.223.372.036.854.775.808 – 9.223.372.036.854.775.808
Ulong	64-bit unsigned heltal	0 – 18.446.744.073.709.551.615
Decimaltal		
float	32-bit Enkeltpræcision flydende kommatype	-3.402823 ³⁸ til 3.402823 ³⁸

double	32-bit Single-precision floating point type	-1.79769313486232 ³⁰⁸ til 1.79769313486232 ³⁰⁸
decimal	128-bit decimaltype til finansielle og mone-tære beregninger	(+ or -)1.0 x 10 ⁻²⁸ to 7.9 x 10 ²⁸ Et meget stort tal.

Tegn		
char	16-bit single Unicode character	Alle valide karakter, e.g. a,*, \x0058 (hex), eller \u0058 (Unicode)
string	En samling af char	Det har altid været lidt af en religiøs diskussion, om string er en datatype eller ej. Årsagen til denne diskussion er, at string er en samling af datatypen char, hvilket gør, at nogen mener den ikke er sin egen datatype. Vi vil dog som udgangspunkt betragte den som en datatype.

Andre		
bool	8-bit logical true/false værdi	True eller False er de to tilstande en variabel af typen bool kan antage.
object	Basistype af alle andre typer.	Denne datatype er i princippet en repræsentant for alle datatyper. Du vil senere i S-forløbet blive præsenteret for hvordan man kan benytte object typen i sin kode.
DateTime	64-bit unsigned heltal	0:00:00 1/1/01 til 23:59:59 12/31/9999 Et tal der repræsenterer antal millisekunder fra d. 01/01 - 0001 kl. 0:00:00:000 DD/MM – ÅÅÅÅ kl. TT:MM:SS:MS

Du skal være opmærksom på en bestemt ting omkring decimaltal.

Når du skal skrive decimaltal til en variabel i din kode, skal du i koden altid benytte '.' (punktum) som decimal tegn i dit tal. Det skal du gøre, fordi compileren kun kan forstå amerikansk standard.

I Danmark benytter vi ',' som decimal tegn og '.' som tusindtals separator, i USA, Sverige og flere andre lande benyttes det omvendt.

Da C# er udviklet i USA, er den kun i stand til at benytte amerikansk standard.

Det betyder dog ikke, at du skal benytte denne standard når du afvikler din kode. Her vil styresystemet sikre at den nationale standard du benytter, bliver omsat til amerikansk standard i koden.



Opgaver

De følgende opgaver vil give dig mulighed for at få afprøvet den overstående teori i noget kode som du skal skrive.

1. Opret en ny Blank Solution
 - a. Giv den navnet *VariablerOgDatatyper*
 - b. Placer den i mappen: *C:\CodeMappe\S1\Console*
2. Via Solution Explorer skal du tilføje et nyt projekt
 - a. Vælg typen *Console App (.Net Framework)*
 - b. Giv den navnet: *Opgave_1*

Til hver af de følgende opgaver, tilføjer du et nyt projekt, af typen *Console App (.Net Framework)*, til *solution VariablerOgDataTyper*.

Opgave1

Erklæring og Initialisering

Til hver af nedenstående delopgaver, skal du i samme projekt (*Opgave_1*), erklærer, initialisere og udskrive indholdet af variablen til konsollen.

1. Erklær og initialiser en variabel, der indeholder dit fulde navn.
2. Erklær og initialiser en variabel, der indeholder din adresse.
3. Erklær og initialiser en variabel, der indeholder din AspIT mail.
4. Erklær og initialiser en variabel, der angiver stien til denne solution.
5. Erklær og initialiser en variabel, der indeholder din alder.
6. Erklær og initialiser en variabel, der indeholder dit fødselsår.
7. Erklær og initialiser en variabel, der indeholder temperaturen (uden decimaler) i Vejle den 19. marts 2019, om morgenen.
8. Erklær og initialiser en variabel, der repræsenterer 0,3 promille.
9. Erklær og initialiser en variabel, der repræsenterer 100%.
10. Erklær og initialiser en variabel, der repræsenterer 125%.
11. Erklær og initialiser en variabel, der repræsenterer 25%
12. Erklær og initialiser en variabel, der repræsenterer den mindst mulige værdi der er større end 0.

Opgave2

Datatyper

Til hver af nedenstående delopgaver, skal du i samme projekt (Opgave_2), erklære, initialisere og udskrive indholdet af variablen til konsollen.

1. Erklær og initialiser en variabel af hver datatype som du finder i tabellerne Heltal og Decimaltal. Du skal ligeledes initialisere dem til en passende startværdi. [Link til INFO](#)
2. Du skal nu få dit program til at udskrive den mindste og den største værdi, de enkelte variabler kan antage. Her skal du udnytte, at du kan tilgå disse værdier via datatypens funktioner. [Link til INFO](#)
3. Du skal oprette 3 variabler med de 3 datatyper Float, Double og Decimal. Du skal initialisere dem med en passende startværdi.
Du skal nu benytte klassen *Math* som indeholder metoden *PI*. Den metode vil give dig værdien af *PI*. For at du kan indsætte værdien i *Float* og *Decimal* datatyperne, skal du benytte *casting*. Det gøres ved, at du foran kaldet til metoden skriver i en parentes hvilken datatype den skal bruges som.
F.eks. `int myInt = (int)Math.PI;` vil give dig værdien af *PI* som en *int* (heltal);

Opgave3

Datatypes string

Datatypes string er den datatype som giver flest udfordringer for programmøren og den datatype der "koster" mest at benytte på en computer.

Hvorfor nu det ?

En string er:

- En samling/række af individuelle elementer af datatypes Char
 - Datatypes Char holder en værdi der repræsenterer den valgte karakter
 - Den valgte værdi omsættes til en karakter gennem en reference til en tegntabel
 - Den valgte karakter vises på skærmen, i et dokument eller på printeren

En *string* kan illustrer på følgende måde:

Lad os se på en *string* der indeholder følgende tekst: "Hej med dig."

I tabellen herunder er denne tekst indsat, så hvert element(*char*) i *string* står i hver sin kolonne.

I rækken "Char kode", står den intiger værdi som datatypes char indeholder.

I rækken "UTF-8 Hex værdi", står den hexadecimale værdi "Char kode" omsættes til i en UTF-8 tegntabel.

I de to nederste rækker kan du se hvad værdien omsættes til, alt efter hvilken skrifttype du har valgt.

Rå tekst	H	e	j		m	e	d		d	i	g	.
Char kode	72	101	106	32	109	101	100	32	100	105	103	46
UTF-8 Hex værdi	48	65	6A	20	6D	65	64	20	64	69	67	2E
Algerian	H	E	J		M	E	D		D	I	G	.
Brush Script MT	ℋ	ℯ	ℵ		ℓ	ℯ	ℓ		ℓ	ℯ	ℵ	.

Link til en UTF-8 ASCII tegntabel - [LINK](#)

Data til de kommende opgaver

Denne tekst skal benyttes i nogle af opgaverne vedrørende string, kopier teksten ind i disse opgaver:

I C# er "string" en datatype, der bruges til at repræsentere en sekvens af tegn. En "string" i C# er en reference type, hvilket betyder, at den er baseret på en objektreference snarere end at være en værditype som "int" eller "float". Strengene i C# er uundgåeligt en vigtig del af mange programmer, da de bruges til at håndtere tekst og karakterdata. C# tilbyder også en række nyttige metoder og egenskaber til at arbejde med strenge, såsom at finde længden af en streng, ændre tegn i en streng, opdele en streng i substrings, og meget mere.

Erklær en variabel med navnet *fruit* og initialiser den med værdien "Pære". Udskriv variabelen. Overskriv bagefter variabelen med værdien "Banan". Udskriv igen.

Til de følgende opgaver, kan du med fordel undersøge hvilke metoder der findes i datatypen *String*, som kan hjælpe dig med at manipulere med datatypens indhold [LINK](#)

1. Erklær to variabler, der henholdsvis skal indeholde dit fornavn og efternavn. Husk reglerne for navngivning af variabler.
2. Udskriv de to variabler i samme `Console.WriteLine()` kommando, så der er mellemrum mellem fornavn og efternavn. Du skal benytte " + " metoden. Link til inspiration [LINK](#)
3. Udskriv de to variabler i samme `Console.WriteLine()` kommando, så der er mellemrum mellem fornavn og efternavn. Du skal benytte *String Interpolation*. Link til inspiration [LINK](#)
4. Udskriv de to variabler i samme `Console.WriteLine()` kommando, så der er mellemrum mellem fornavn og efternavn. Du skal benytte *String.Format()*. Link til inspiration [LINK](#)
5. Opret en variabel der kan initialiseres med den tekst der står øverst på siden og navngiv den *opgaveTekst*.
6. Udskriv indholdet af *opgaveTekst* til konsollen.
7. Udskriv indholdet af *opgaveTekst* til konsollen, men hvor alt er skrevet med store bogstaver (upper-case)
8. Udskriv indholdet af *opgaveTekst* til konsollen, men hvor alt er skrevet med små bogstaver (lower-case)
9. Udskriv til konsollen, antallet af karakterer der forekommer i variabelen *opgaveTekst*.
10. Opret en variabel der initialiseres med en kopi af variabelen *opgaveTekst*, kald den nye variabel *temp*.
11. Tag indholdet i *temp* og udskift alle mellemrum i teksten med karakteren `ø`, og udskriv indholdet af *temp* til konsollen.



12. I variablen *temp*, skal du finde placeringen af første og sidste forekomst af karakteren æ

Udskriv placeringerne til konsollen.

13. Benyt variablen *opgaveTekst* og udskriv til konsollen, længden af indholdet i variablen *opgaveTekst*.

14. Benyt variablen *opgaveTekst* og udskriv til konsollen, placeringen af første og anden forekomst af ordet "at".

15. Benyt variablen *opgaveTekst* og udskriv til konsollen, indholdet af *opgaveTekst* som er placeret mellem første og anden forekomst af ordet "at".