

Undervisningsbeskrivelse

Stamoplysninger til brug ved prøver til gymnasiale uddannelser

Termin	2018-2021
Institution	Rybners Tekniske Gymnasium
Uddannelse	HTX
Fag og niveau	Programmering B
Lærer(e)	David Lindholm
Hold	8HX118D

Oversigt over gennemførte undervisningsforløb

1. år:

Titel 1 - Introduktion til struktureret programmering m. JavaScript

Titel 2 - Introduktion til funktionel programmering m. JavaScript

Titel 3 - Grundlæggende struktureret programmering m. Python

Titel 4 - Grundlæggende funktionel programmering m. Python

Titel 5 - Grundlæggende objekt-orienteret programmering m. Python

2. år:

Titel 6 - Spilprogrammering m. Python

Titel 7 - Struktureret og funktionel programmering m. C#

Titel 8 - Objekt-orienteret programmering m. C#

Titel 9 - Designmønstre m. C#

3. år:

Titel 10 - Videregående struktureret programmering m. Java

Titel 11 - Videregående funktionel programmering m. Java

Titel 12 - Videregående objekt-orienteret programmering m. Java

Titel 13 - Grafiske brugergrænseflader m. Java

Titel 14 - Algoritmer m. Java

Titel 15 - Analyse af selvvalgt sprog

Titel 16 - Eksamensprojekt

1 Introduktion til struktureret programmering m. JavaScript

1.1 Indhold

Kernestof:

- Programmeringssprog og elementer i programmerens opbygning, herunder variable, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmestre

IDE: <https://repl.it/languages/javascript>

Emner:

Variabler, loops, if/else, switch/case, datatyper

Anvendt litteratur:

https://www.w3schools.com/js/js_syntax.asp
https://www.w3schools.com/js/js_comments.asp
https://www.w3schools.com/js/js_variables.asp
https://www.w3schools.com/js/js_operators.asp
https://www.w3schools.com/js/js_arithmetic.asp
https://www.w3schools.com/js/js_datatypes.asp
https://www.w3schools.com/js/js_if_else.asp
https://www.w3schools.com/js/js_loop_for.asp
Egenfremstillede kodeeksempler

1.2 Omfang

5 lektioner

1.3 Særlige fokuspunkter

Kompetencer:

- Redegøre for arkitekturen af programmer på forskellige abstraktionsniveauer, herunder relationen mellem brug og funktion
- Rette, tilpasse og udvide avancerede programmer
- Arbejde inkrementelt og systematisk i programmeringsprocessen
- Demonstrere viden om fagets identitet og metoder

1.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Hands-on projektarbejde
- Opgaveløsning

2 Introduktion til funktionel programmering m. JavaScript

2.1 Indhold

Kernestof:

- programmeringssprog og elementer i programmerers opbygning, herunder variabler, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmemønstre

IDE: <https://repl.it/languages/javascript>

Emner:

Funktioner, array

Anvendt litteratur:

https://www.w3schools.com/js/js_functions.asp

https://www.w3schools.com/js/js_arrays.asp

https://www.w3schools.com/js/js_array_methods.asp

https://www.w3schools.com/js/js_array_sort.asp

https://www.w3schools.com/js/js_array_iteration.asp

Egenfremstillede kodeeksempler

2.2 Omfang

5 lektioner

2.3 Særlige fokuspunkter

Kompetencer:

- Redegøre for arkitekturen af programmer på forskellige abstraktionsniveauer, herunder relationen mellem brug og funktion
- Rette, tilpasse og udvide avancerede programmer
- Arbejde inkrementelt og systematisk i programmeringsprocessen
- Demonstrere viden om fagets identitet og metoder

2.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Hands-on projektarbejde
- Opgaveløsning

3 Grundlæggende struktureret programmering m. Python

3.1 Indhold

Kernestof:

- Programmeringssprog og elementer i programmeres opbygning, herunder variabler, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmestre
- Arkitekturen for programmeres interaktion med omgivelserne med henblik på hændelsesstyret interaktion og interaktion mellem systemer
- Arbejdsgange og systematik i programmeringsprocessen, herunder test og fejlfinding

IDE: PyCharm m. Python 3.6.8

Emner:

Variabler, if/else, switch/case, datatyper, brugerinteraktion

Anvendt litteratur:

<https://docs.python.org/3/tutorial/introduction.html>

<https://docs.python.org/3/tutorial/datastructures.html>

<https://docs.python.org/3/tutorial/datastructures.html#tuples-and-sequences>

<https://docs.python.org/3/tutorial/datastructures.html#using-lists-as-stacks>

<https://docs.python.org/3/tutorial/datastructures.html#using-lists-as-queues>

<https://docs.python.org/3/tutorial/controlflow.html#if-statements>

Egenfremstillede kodeeksempler

3.2 Omfang

8 lektioner

3.3 Særlige fokuspunkter

Kompetencer:

- Redegøre for arkitekturen af programmer på forskellige abstraktionsniveauer, herunder relationen mellem brug og funktion
- Redegøre for simple specifikationsmodeller og realisere disse i simple velstrukturerede programmer samt teste disse
- Rette, tilpasse og udvide avancerede programmer
- Arbejde inkrementelt og systematisk i programmeringsprocessen
- Demonstrere viden om fagets identitet og metoder

3.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Opgaveløsning

4 Grundlæggende funktionel programmering m. Python

4.1 Indhold

Kernestof:

- programmeringssprog og elementer i programmers opbygning, herunder variabler, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmestre
- Arkitekturen for programmers interaktion med omgivelserne med henblik på hændelsesstyret interaktion og interaktion mellem systemer
- Arbejdsgange og systematik i programmeringsprocessen, herunder test og fejlfinding

IDE: PyCharm m. Python 3.6.8

Emner:

Funktioner, lister, tupel, dictionary, loops

Anvendt litteratur:

<https://docs.python.org/3/tutorial/introduction.html#using-python-as-a-calculator>

<https://docs.python.org/3/tutorial/controlflow.html#for-statements>

<https://docs.python.org/3/tutorial/controlflow.html#break-and-continue-statements-and-else-clauses-on-loops>

<https://docs.python.org/3/tutorial/controlflow.html#defining-functions>

<https://docs.python.org/3/tutorial/controlflow.html#default-argument-values>

Egenfremstillede kodeeksempler

4.2 Omfang

12 lektioner

4.3 Særlige fokuspunkter

Kompetencer:

- Redegøre for arkitekturen af programmer på forskellige abstraktionsniveauer, herunder relationen mellem brug og funktion
- Redegøre for simple specifikationsmodeller og realisere disse i simple velstrukturerede programmer samt teste disse
- Rette, tilpasse og udvide avancerede programmer
- Arbejde inkrementelt og systematisk i programmeringsprocessen
- Demonstrere viden om fagets identitet og metoder

4.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Hands-on projektarbejde
- Opgaveløsning

5 Grundlæggende objekt-orienteret programmering m. Python

5.1 Indhold

Kernestof:

- programmeringssprog og elementer i programmers opbygning, herunder variabler, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmestre
- Arkitekturen for programmers interaktion med omgivelserne med henblik på hændelsesstyret interaktion og interaktion mellem systemer
- Arbejdsgange og systematik i programmeringsprocessen, herunder test og fejlfinding

IDE: PyCharm m. Python 3.6.8

Emner:

Klasser, objekter, metoder, nedarvning, constructor, klassediagram, dokumentation

Anvendt litteratur:

https://www.tutorialspoint.com/python/python_classes_objects.htm

<https://docs.python.org/3/tutorial/classes.html>

<https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/>

<https://www.python.org/dev/peps/pep-0008/>

Egenfremstillede kodeeksempler

5.2 Omfang

12 lektioner

5.3 Særlige fokuspunkter

Kompetencer:

- Redegøre for arkitekturen af programmer på forskellige abstraktionsniveauer, herunder relationen mellem brug og funktion
- Redegøre for simple specifikationsmodeller og realisere disse i simple velstrukturerede programmer samt teste disse
- Anvende avancerede konstruktioner i et programmeringssprog
- Rette, tilpasse og udvide avancerede programmer
- Arbejde inkrementelt og systematisk i programmeringsprocessen
- Demonstrere viden om fagets identitet og metoder
- Abstrakte programmeringsbeskrivelser og dokumentation.

5.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Hands-on projektarbejde
- Opgaveløsning

6 Struktureret og funktionel programmering m. C#

6.1 Indhold

Kernestof:

- programmeringssprog og elementer i programmers opbygning, herunder variabler, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmestre
- Arkitekturen for programmers interaktion med omgivelserne med henblik på hændelsesstyret interaktion og interaktion mellem systemer
- Arbejdsgange og systematik i programmeringsprocessen, herunder test og fejlfinding

IDE: Microsoft Visual Studio

Emner:

Kontrolstrukturer, løkker, variabler, datatyper, collections (især lister), Linq Funktioner, lister, tupel, dictionary, loops

Anvendt litteratur:

Egenfremstillede kodeeksempler og materialer

<https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable?view=netframework-4.8>

<https://www.w3resource.com/csharp-exercises/>

6.2 Omfang

22 lektioner

6.3 Særlige fokuspunkter

Kompetencer:

- Redegøre for arkitekturen af programmer på forskellige abstraktionsniveauer, herunder relationen mellem brug og funktion
- Redegøre for simple specifikationsmodeller og realisere disse i simple velstrukturerede programmer samt teste disse
- Rette, tilpasse og udvide avancerede programmer
- Arbejde inkrementelt og systematisk i programmeringsprocessen
- Demonstrere viden om fagets identitet og metoder

6.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Hands-on projektarbejde
- Opgaveløsning

7 Objekt-orienteret programmering m. C#

7.1 Indhold

Kernestof:

- programmeringssprog og elementer i programmers opbygning, herunder variabler, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmestre
- Arkitekturen for programmers interaktion med omgivelserne med henblik på hændelsesstyret interaktion og interaktion mellem systemer
- Arbejdsgange og systematik i programmeringsprocessen, herunder test og fejlfinding

IDE: Microsoft Visual Studio

Emner:

OOP, indkapsling, arv, polymorfi, lambda, fejlhåndtering, UML, test/use cases

Anvendt litteratur:

Egenfremstillede kodeeksempler og materialer

<https://www.tutlane.com/tutorial/csharp/csharp-encapsulation>

<https://www.tutlane.com/tutorial/csharp/csharp-inheritance>

<https://www.tutlane.com/tutorial/csharp/csharp-polymorphism>

<https://docs.microsoft.com/en-us/dotnet/standard/exceptions>

<https://www.tutorialsteacher.com/csharp/csharp-generics>

<https://csharpindepth.com/articles/Events>

7.2 Omfang

24 lektioner

7.3 Særlige fokuspunkter

Kompetencer:

- Redegøre for arkitekturen af programmer på forskellige abstraktionsniveauer, herunder relationen mellem brug og funktion
- Redegøre for simple specifikationsmodeller og realisere disse i simple velstrukturerede programmer samt teste disse
- Anvende avancerede konstruktioner i et programmeringssprog
- Rette, tilpasse og udvide avancerede programmer
- Arbejde inkrementelt og systematisk i programmeringsprocessen
- Demonstrere viden om fagets identitet og metoder
- Abstrakte programmeringsbeskrivelser og dokumentation.

7.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Hands-on projektarbejde
- Opgaveløsning

8 Designmønstre m. C#

8.1 Indhold

Kernestof:

- Programmeringssprog og elementer i programmets opbygning, herunder variabler, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmemønstre
- Arbejdsgange og systematik i programmeringsprocessen, herunder test og fejlfinding
- Abstrakte programmeringsbeskrivelser og dokumentation.

IDE: Microsoft Visual Studio

Emner:

Model View Controller, Databaser

Anvendt litteratur:

Egenfremstillede kodeeksempler og materialer

<https://docs.microsoft.com/en-us/aspnet/mvc/overview/older-versions-1/models-data/validation-with-the-data-annotation-validators-cs>

https://www.tutorialspoint.com/asp.net_mvc/index.htm

8.2 Omfang

18 lektioner

8.3 Særlige fokuspunkter

Kompetencer:

- Anvende avancerede konstruktioner i et programmeringssprog
- Demonstrere viden om fagets identitet og metoder

8.4 Væsentligste arbejdsformer

- Tavleundervisning
- Opgaveløsning
- Gruppearbejde
- Projektarbejde

9 Spilprogrammering m. Python

9.1 Indhold

Kernestof:

- Arkitekturen for programmets interaktion med omgivelserne med henblik på hændelsesstyret interaktion og interaktion mellem systemer
- Generiske programdele og biblioteksmoduler
- Arbejdsgange og systematik i programmeringsprocessen, herunder test og fejlfinding

IDE: PyCharm m. Python 3.6.8 & Pygame 1.9.4

Emner:

Spiludvikling, spildesign, brugerinteraktion, feedback, grafikprogrammering, basal kunstig intelligens

Anvendt litteratur:

<https://nerdparadise.com/programming/pygame/part1>

<https://pythonprogramming.net/pygame-python-3-part-1-intro/>

<http://thepythongamebook.com/en:pygame:start>

Salen, K & Zimmerman, E. - Rules of Play - Kap 7 - "Defining Games"

9.2 Omfang

30 lektioner

9.3 Særlige fokuspunkter

Kompetencer:

- Bruge programmering til at undersøge et emne eller problemområde, med henblik på – via programmets funktion - at skabe ny indsigt eller til at løse et problem
- Redegøre for arkitekturen af programmer på forskellige abstraktionsniveauer, herunder relationen mellem brug og funktion
- Anvende avancerede konstruktioner i et programmeringssprog
- Arbejde inkrementelt og systematisk i programmeringsprocessen

9.4 Væsentligste arbejdsformer

- Gruppearbejde
- Hands-on projektarbejde

10 Videregående struktureret programmering m. Java

10.1 Indhold

Kernestof:

- Programmeringssprog og elementer i programmers opbygning, herunder variable, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmemønstre
- Arbejdsgange og systematik i programmeringsprocessen, herunder test og fejlfinding

IDE: Netbeans m. Java 1.8.*

Emner:

Variabler, konstanter, datatyper, brugerinput, if/else, switch/case, den ternære operator, kommentarer, while, for, do-while

Anvendt litteratur:

David Lindholm - Bogen om Java v0.16, Kapitel 9-14
Egenfremstillede kodeeksempler

10.2 Omfang

10 lektioner

10.3 Særlige fokuspunkter

Kompetencer:

- Bruge programmering til at undersøge et emne eller problemområde, med henblik på – via programmets funktion - at skabe ny indsigt eller til at løse et problem
- Redegøre for simple specifikationsmodeller og realisere disse i simple velstrukturerede programmer samt teste disse
- Rette, tilpasse og udvide avancerede programmer

10.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Opgaveløsning

11 Videregående funktionel programmering m. Java

11.1 Indhold

Kernestof:

- Programmeringssprog og elementer i programmets opbygning, herunder variable, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmemønstre
- Arbejdsgange og systematik i programmeringsprocessen, herunder test og fejlfinding
- Abstrakte programmeringsbeskrivelser og dokumentation

IDE: Netbeans m. Java 1.8.*

Emner:

metoder, polymorfi, rekursion, paradigmer, array, arraylist, linkedlist, kø, stak, Manåna Princippet

Anvendt litteratur:

David Lindholm - Bogen om Java v0.16, Kapitel 15-17

Egenfremstillede kodeeksempler

11.2 Omfang

12 lektioner

11.3 Særlige fokuspunkter

Kompetencer:

- Bruge programmering til at undersøge et emne eller problemområde, med henblik på – via programmets funktion - at skabe ny indsigt eller til at løse et problem
- Anvende avancerede konstruktioner i et programmeringssprog
- Redegøre for simple specifikationsmodeller og realisere disse i simple velstrukturerede programmer samt teste disse
- Rette, tilpasse og udvide avancerede programmer
- Demonstrere viden om fagets identitet og metoder

11.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Opgaveløsning

12 Videregående objekt-orienteret programmering m. Java

12.1 Indhold

Kernestof:

- Programmeringssprog og elementer i programmets opbygning, herunder variable, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmemønstre
- Generiske programdele og biblioteksmoduler
- Arbejdsgange og systematik i programmeringsprocessen, herunder test og fejlfinding
- Abstrakte programmeringsbeskrivelser og dokumentation

IDE: Netbeans m. Java 1.8.*

Emner:

Klasser, objekter, nedarvning, klassesdiagram, tilstandsdiagram, biblioteker, constructor, accessors, indkapsling, forbindelsestyper og kardinaliteter, abstrakte klasser, indlejrede klasser, interface, enumerator, singleton, command pattern

Anvendt litteratur:

David Lindholm - Bogen om Java v0.16, Kapitel 18-21
Egenfremstillede kodeeksempler

12.2 Omfang

20 lektioner

12.3 Særlige fokuspunkter

Kompetencer:

- Bruge programmering til at undersøge et emne eller problemområde, med henblik på – via programmets funktion - at skabe ny indsigt eller til at løse et problem
- Anvende avancerede konstruktioner i et programmeringssprog
- Redegøre for arkitekturen af programmer på forskellige abstraktionsniveauer, herunder relationen mellem brug og funktion
- Redegøre for simple specifikationsmodeller og realisere disse i simple velstrukturerede programmer samt teste disse
- Rette, tilpasse og udvide avancerede programmer
- Demonstrere viden om fagets identitet og metoder
- Arbejde inkrementelt og systematisk i programmeringsprocessen

12.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Opgaveløsning

13 Grafiske brugergrænseflader m. Java

13.1 Indhold

Kernestof:

- Programmeringssprog og elementer i programmers opbygning, herunder variable, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmestre
- Arkitekturen for programmers interaktion med omgivelserne med henblik på hændelsesstyret interaktion og interaktion mellem systemer
- Generiske programdele og biblioteksmoduler
- Arbejdsgange og systematik i programmeringsprocessen, herunder test og fejlfinding
- Abstrakte programmeringsbeskrivelser og dokumentation

IDE: Netbeans m. Java 1.8.* & Swing

Emner:

Grafikprogrammering, brugergrænsefladedesign, grafiske elementer, brugerinteraktion, model view controller

Anvendt litteratur:

David Lindholm - Bogen om Java v0.16, Kapitel 23-25
Egenfremstillede kodeeksempler

13.2 Omfang

10 lektioner

13.3 Særlige fokuspunkter

Kompetencer:

- Bruge programmering til at undersøge et emne eller problemområde, med henblik på – via programmets funktion - at skabe ny indsigt eller til at løse et problem
- Anvende avancerede konstruktioner i et programmeringssprog
- Redegøre for arkitekturen af programmer på forskellige abstraktionsniveauer, herunder relationen mellem brug og funktion
- Redegøre for simple specifikationsmodeller og realisere disse i simple velstrukturerede programmer samt teste disse
- Arbejde inkrementelt og systematisk i programmeringsprocessen.

13.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Opgaveløsning

14 Algoritmer m. Java

14.1 Indhold

Kernestof:

- Programmeringssprog og elementer i programmers opbygning, herunder variabler, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmemønstre
- Arbejdsgange og systematik i programmeringsprocessen, herunder test og fejlfinding
- Abstrakte programmeringsbeskrivelser og dokumentation

IDE: Netbeans m. Java 1.8.*

Emner:

Søgning, sortering, tidskompleksitet

Anvendt litteratur:

<https://www.toptal.com/developers/sorting-algorithms>

David Lindholm - Bogen om Java v0.16, Kapitel 28-31

14.2 Omfang

10 lektioner

14.3 Særlige fokuspunkter

Kompetencer:

- Anvende avancerede konstruktioner i et programmeringssprog
- Redegøre for arkitekturen af programmer på forskellige abstraktionsniveauer, herunder relationen mellem brug og funktion
- Rette, tilpasse og udvide avancerede programmer
- Demonstrere viden om fagets identitet og metoder

14.4 Væsentligste arbejdsformer

- Tavleundervisning
- Opgaveløsning

15 Analyse af selvvalgt sprog

15.1 Indhold

Kernestof:

- Generiske programdele og biblioteksmoduler
- Arbejdsgange og systematik i programmeringsprocessen, herunder test og fejlfinding
- Abstrakte programmeringsbeskrivelser og dokumentation

Emner:

C/C++ ELLER Elm/Ruby on Rails ELLER Fortran/COBOL ELLER Go/Scala ELLER Android

Anvendt litteratur:

Tutorialsæt til de valgte sprog

15.2 Omfang

15 lektioner

15.3 Særlige fokuspunkter

Kompetencer:

- Bruge programmering til at undersøge et emne eller problemområde, med henblik på – via programmets funktion - at skabe ny indsigt eller til at løse et problem
- Demonstrere viden om fagets identitet og metoder
- Arbejde inkrementelt og systematisk i programmeringsprocessen

15.4 Væsentligste arbejdsformer

- Gruppearbejde
- Hands-on projektarbejde

16 Eksamensprojekt

16.1 Indhold

Kernestof:

- Programmeringssprog og elementer i programmers opbygning, herunder variabler, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmestre
- Arkitekturen for programmers interaktion med omgivelserne med henblik på hændelsesstyret interaktion og interaktion mellem systemer
- Generiske programdele og biblioteksmoduler
- Arbejdsgange og systematik i programmeringsprocessen, herunder test og fejlfinding
- Abstrakte programmeringsbeskrivelser og dokumentation.

Emner:

Projektarbejde med IT-system baseret på udleveret oplæg

Anvendt litteratur:

Pensum

Elevernes egen research

16.2 Omfang

20 klokketimer

16.3 Særlige fokuspunkter

Kompetencer:

- Bruge programmering til at undersøge et emne eller problemområde, med henblik på – via programmets funktion - at skabe ny indsigt eller til at løse et problem
- Anvende avancerede konstruktioner i et programmeringssprog
- Redegøre for arkitekturen af programmer på forskellige abstraktionsniveauer, herunder relationen mellem brug og funktion
- Redegøre for simple specifikationsmodeller og realisere disse i simple velstrukturerede programmer samt teste disse
- Demonstrere viden om fagets identitet og metoder
- Arbejde inkrementelt og systematisk i programmeringsprocessen.

16.4 Væsentligste arbejdsformer

- Gruppearbejde
- Hands-on projektarbejde