Rybners®
TEKNISKE GYMNASIUM

MINISTERIET FOR BØRN OG UNDERVISNING
KVALITETS- OG TILSYNSSTYRELSEN

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	8HX118E

Oversigt over gennemførte undervisningsforløb

1. år:

Titel 1 - Introduktion til struktureret programmering m. JavaScript

Titel 2 - Introduktion til funktionel programmering m. JavaScript

Titel 3 - Tværfagligt projekt - Informatik C/Programmering B

Titel 4 - Grundlæggende struktureret programmering m. Python

Titel 5 - Grundlæggende funktionel programmering m. Python

Titel 6 - Grundlæggende objekt-orienteret programmering m. Python

Titel 7 - Database-programmering m. Python

2. år:

Titel 8 - Spilprogrammering m. Python

Titel 9 - Struktureret og funktionel programmering m. C#

Titel 10 - Objekt-orienteret programmering m. C#

Titel 11 - Designmønstre m. C#

Titel 12 - Studieretningscase

3. år:

Titel 13 - Videregående struktureret programmering m. Java

Titel 14 - Videregående funktionel programmering m. Java

Titel 15 - Videregående objekt-orienteret programmering m. Java

Titel 16 - Grafiske brugergrænseflader m. Java

Titel 17 - Algoritmer m. Java

Titel 18 - Analyse af selvvalgt sprog

Titel 19 - Eksamensprojekt

1 Introduktion til struktureret programmering m. JavaScript

1.1 Indhold

Kernestof:

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

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 programmers 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 Tværfagligt projekt - Informatik C/Programmering B

3.1 Indhold

Kernestof:

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

IDE: https://repl.it/languages/html
Emner:
HTML, CSS, webudvikling
Anvendt litteratur:
Diverse materialer fra: https://www.w3schools.com/html/ og https://www.w3schools.com/css/ gennemgået i Informatik.

3.2 Omfang

4 lektioner

3.3 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
- Behandle problemstillinger i samspil med andre fag
- 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

3.4 Væsentligste arbejdsformer

- Gruppearbejde
- Hands-on projektarbejde

4 Grundlæggende struktureret programmering m. Python

4.1 Indhold

Kernestof:

- Programmeringssprog og elementer i programmers opbygning, herunder variabler, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmemønstre
- 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:
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

4.2 Omfang

8 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
- Opgaveløsning

5 Grundlæggende funktionel programmering m. Python

5.1 Indhold

Kernestof:

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

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
- Rette, tilpasse og udvide avancerede programmer
- Arbejde inkrementelt og systematisk i programmeringsprocessen
- Demonstrere viden om fagets identitet og metoder

5.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Hands-on projektarbejde
- Opgaveløsning

6 Grundlæggende objekt-orienteret programmering m. Python

6.1 Indhold

Kernestof:

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

6.2 Omfang

12 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
- 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.

6.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Hands-on projektarbejde
- Opgaveløsning

7 Database-programmering m. Python

7.1 Indhold

Kernestof:

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

IDE: PyCharm m. Python 3.6.8 & SQLite m. SQLite Studio
Emner:
Databaser (SQLite), databehandling
Anvendt litteratur:
https://sqlite.org/docs.html
https://www.pythoncentral.io/introduction-to-sqlite-in-python/
https://docs.python.org/2/library/sqlite3.html
Egenfremstillet introduktion til SQLite
Egenfremstillede kodeeksempler

7.2 Omfang

6 lektioner

7.3 Særlige fokuspunkter

Kompetencer:

- 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
- Behandle problemstillinger i samspil med andre fag

7.4 Væsentligste arbejdsformer

- Gruppearbejde
- Hands-on projektarbejde

8 Struktureret og funktionel programmering m. C#

8.1 Indhold

Kernestof:

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

8.2 Omfang

22 lektioner

8.3 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

8.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Hands-on projektarbejde
- Opgaveløsning

9 Objekt-orienteret programmering m. C#

9.1 Indhold

Kernestof:

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

9.2 Omfang

24 lektioner

9.3 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.

9.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Hands-on projektarbejde
- Opgaveløsning

10 Designmønstre m. C#

10.1 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: 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

10.2 Omfang

18 lektioner

10.3 Særlige fokuspunkter

Kompetencer:

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

10.4 Væsentligste arbejdsformer

- Tavleundervisning
- Opgaveløsning
- Gruppearbejde
- Projektarbejde

11 Spilprogrammering m. Python

11.1 Indhold

Kernestof:

- 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

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"

11.2 Omfang

30 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
- 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

11.4 Væsentligste arbejdsformer

- Gruppearbejde
- Hands-on projektarbejde

12 Studieretningscase

12.1 Indhold

Kernestof:

- Programmeringssprog og elementer i programmers opbygning, herunder variabler, typer, udtryk, kontrolstrukturer, parametrisering/abstraktionsmekanismer, rekursion, polymorfi og algoritmemønstre
- Behandle problemstillinger i samspil med andre fag
- 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
Løsning af tværfaglig opgave med Matematik A.
Anvendt litteratur:
Pensum
Elevernes egen research

12.2 Omfang

1 uge

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
- Demonstrere viden om fagets identitet og metoder
- Arbejde inkrementelt og systematisk i programmeringsprocessen.

12.4 Væsentligste arbejdsformer

- Hands-on projektarbejde

13 Videregående struktureret programmering m. Java

13.1 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

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

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
- Redegøre for simple specifikationsmodeller og realisere disse i simple velstrukturerede programmer samt teste disse
- Rette, tilpasse og udvide avancerede programmer

13.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Opgaveløsning

14 Videregående funktionel programmering 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:
metoder, polymorfi, rekursion, paradigmer, array, arraylist, linkedlist, kø, stak, Manãna Princippet
Anvendt litteratur:
David Lindholm - Bogen om Java v0.16, Kapitel 15-17
Egenfremstillede kodeeksempler

14.2 Omfang

12 lektioner

14.3 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

14.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Opgaveløsning

15 Videregående objekt-orienteret programmering m. Java

15.1 Indhold

Kernestof:

- Programmeringssprog og elementer i programmers opbygning, herunder variabler, 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, klassediagram, 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

15.2 Omfang

20 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
- 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

15.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Opgaveløsning

16 Grafiske brugergrænseflader m. Java

16.1 Indhold

Kernestof:

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

16.2 Omfang

10 lektioner

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
- Arbejde inkrementelt og systematisk i programmeringsprocessen.

16.4 Væsentligste arbejdsformer

- Tavleundervisning
- Gruppearbejde
- Opgaveløsning

17 Algoritmer m. Java

17.1 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

17.2 Omfang

10 lektioner

17.3 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

17.4 Væsentligste arbejdsformer

- Tavleundervisning
- Opgaveløsning

18 Analyse af selvvalgt sprog

18.1 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

18.2 Omfang

15 lektioner

18.3 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

18.4 Væsentligste arbejdsformer

- Gruppearbejde
- Hands-on projektarbejde

19 Eksamensprojekt

19.1 Indhold

Kernestof:

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

19.2 Omfang

20 klokketimer

19.3 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.

19.4 Væsentligste arbejdsformer

- Gruppearbejde
- Hands-on projektarbejde