Fag / Navn: Objektorienteret programmering (16472-2)	**Varighed:** 10 dage OOP og SOLID 4 dage Trådeprogrammering 6 dage
Fagtype: Bundet	
Uddannelse: Datateknikker med programmering	
Præsentationsstandard: Avanceret	

Beskrivelse af faget

I faget objektorienteret programmering arbejdes der med objektbegreberne og de grundlæggende principper for OOP. Faget er del op i to underfag OOP & SOLID samt trådprogrammering.

OOP og SOLID

I OOP og SOLID lærer eleverne om de fundamentale principper indenfor OOP, herunder arv, polymorfi og indkapsling. Endvidere arbejdes der med S.O.L.I.D som designprincip, så eleverne får forståelsen af vigtigheden af lagdeling og korrekt design. Eleverne introduceres også til simple designmønstre. Undervisningen vil bære præg af meget tavleundervisning, med mindre opgaveløsninger undervejs.

Trådprogrammering (6 dage)

I trådprogrammering på H2 kommer eleverne til at arbejde med nedenstående principper inden for både synkron og asynkron programmering

- Hvad er en proces vs. en tråd
- Tråde og threadspools
- Tråd synkronisering
- Asynkron programmering

Der lægges vægt på selvstændigt arbejde, så eleven får at solidt programmeringsfundament inden de efterfølgende hovedforløb.
Faget afsluttes med eksempelvis en selvstændig programmeringsopgave, som gør brug af tråd koncepterne.

Uddannelsesmål

H1

1. Eleven kan anvende et objektorienteret programmeringssprog til at udarbejde konsolprogrammer, der indeholder flere klasser og er i overensstemmelse med OOP konceptet.
2. Eleven har en grundlæggende viden om det valgte programmeringssprog/framework.
3. Eleven kan definere og designe egne klasser.
4. Eleven kan erklære og instantiere objekter.
5. Eleven kan redegøre for typer af collections og kan udpege hensigtsmæssigt i forhold til et behov.
6. Eleven kan anvende en given kodestandard for det pågældende sprog.
7. Eleven kan håndtere "exception handling".
8. Eleven kan redegøre for OOP konceptet såsom indkapsling, polymorfi og arv.
9. Eleven kan udarbejde en applikation som gør brug af OOP konceptet.

H2

10. Eleven kan implementere abstrakte klasser og metoder.

11. Eleven kan skelne mellem override og overload af metoder.
12. Eleven kan begrunde valget af "access modifiers"/virkefelter.
13. Eleven kan oprette og implementere et selvudviklet interface.
14. Eleven kan benytte funktion pointer/callback.
15. Eleven kan udarbejde UML klassediagrammer.
16. Eleven kan designe en simpel domænemodel baseret på best practice.
17. Eleven kan redegøre for betydningen af løs kobling og afhængigheder mellem moduler.
18. Eleven kan udføre asynkron programmering med threads, herunder anvende forskellige thread klasser.
19. Eleven kan redegøre for grundlæggende problemstilling med Thread Safety og Atomic State.
20. Eleven kan benytte frameworkets klasser til asynkron programmering, der håndterer problemerne med Thread Safety og synkronisering.
21. Eleven kan oprette en multitrådet applikation samt redegøre for potentielle udfordringer i forhold til tråde, herunder dead locks, live locks og data race.
22. Eleven kan redegøre for mulighederne ved at anvende anonyme metoder og Lambda metoder.

Læringsmål

Synkron (19, 21)
Der arbejdes med grundlæggende tråde (multitrådet) og synkronisering imellem disse. Problemstillinger som eksempelvis dead locks, data race og starvation introduceres blandt andet via to af de klassiske problemer The Dining Philosophers og Producer-Consumer.

Læringsmål

Asynkron (18, 20)
Der arbejdes med grundlæggende asynkronprogrammering via frametworkets metoder Async/Await samt hvordan synkrone metoder implementeres som asynkrone metoder. Eleverne analyserer callback-handlere, som kaldes, når hændelsen indtræffe samt Task-objekt. Herudover lægges der vægt på at eleverne forstår forskellen imellem asynkron- & parallel programmering

Læringsmål

OOP (10,11,12,13,14,15,22)
Der arbejdes med de grundlæggende principper for OOP, såsom arv, polymorfi og interfaces. Undervejs for eleven introduktion til UML klassediagrammet. Der arbejdes løbende med løsning af mindre opgaver. Eleven får som afslutning på H2 en mulighed for at benytte og arbejde ud fra principperne i afsluttende H2 projekt.

Læringsmål

SOLID (16, 17)
Eleven skal udover at kunne benytte principperne for OOP også introduceres til selve opbygningen og det fysiske design af en applikation, som er skalerbar og nem at vedligeholde. Vi benytter derfor SOLID principperne og best practice. Endvidere introduceres eleverne for en række GoF designmønstre, hvis der er tid

Bedømmelse

Faget bedømmes efter 7-trinsskalaen og karakteren afgives den sidste dag på hovedforløbet. Eleven evalueres løbende på afleveringer, fremlæggelser, deltagelse i undervisningen, opnået resultater i quizzes, samt afsluttende modul projekter. Endvidere bedømmes eleven også på hans/hendes kode, herunder brug af kodedokumentation, overholdelse af kodestandard og "best practice", samt versionsstyring af koden.

***Faget bidrager til følgende kompetencemål**: 23, 27, 28, 30*