«Sykkelutleie»

Prosjekt2

# Oppgave:

Du skal utvikle en Python-løsning for sykkelutleie

Del 1: Oppstart med utvikling av tenkt løsning med klassediagram (UML)

* Tegn UML for alle klasser du vet du trenger og skriv opp de attributtene og metodene du tror du vil trenge (skisse)
* Tegn opp menysystemet du ønsker å lage (skisse)

Del 2: Programmere løsningen

* Dere skal ta egne avgjørelser for hvordan deres system skal se ut, men det må dekke alle kravene under krav og kriterier.

# Krav og kriterier:

**Obs: Trenger ikke arv i denne innleveringen**

**Kunde**

* Lage kunde
* Logge inn
* Velge butikk
* Se ledige sykler
* Leie sykler til 50kr timen
* Leie sykler til 200kr per dag
* Leie sykler til 600kr per uke
* Familierabatt
  + Ved leie av 3-5 sykler så får kunden 30% rabatt på totalprisen, gjelder alle kombinasjoner over.
* Se historikk
  + Aktiv sykkel-leie
  + Innleverte/leide sykler over tid

**Butikk**

* Lage butikk
* Logge inn
* Lage en faktura når syklene blir levert
* Vise lagerbeholdningen
* Endre lagerbeholdning (kjøpe inn eller kaste ødelagte sykler)
* Sjekke forespørsler på time-, daglig- og ukentlig basis ved å kryssverifisere lagerbeholdningen. (er det nok ledige sykler for forespørselen. Tenk en kunde ringer og en butikkansatt sjekker ledige sykler)

**Teknisk**

* Skal være løst med OOP-tankegang
* Du skal kunne lage en bruker (så enkelt du selv føler for, men bruk en klasse der det gir mening)
* Du skal kunne logge inn (trenger ikke sikkerhet)
* All informasjon skal lagres og hentes opp igjen underveis fra en fil. Du skal kunne se i filen hvilken kunde som har kjøpt hva. Når en kunde leverer sykkel tilbake så skal infoen bevares på en eller annen måte sånn at man kan se kundehistorikk.
* Oppgaven skal settes opp med Git-backup

**Dokumentasjon i prosjekt2.docx**

Alt under skal lagres i Word-filen **prosjekt2.docx**

* Klassediagram fra del1 (skissert UML)
* Klassediagram fra faktisk ferdig løsning og skrive om forskjellen fra skissert UML til ferdig program UML
  + Skriv litt om hva du har lært her i prosessen.
* Skriv om hva Git brukes til og hvorfor dette er viktig innen programmering. Maks 500 ord.
* Skriv kort om din løsning for fillagring. Ta skjermbilder av koden du bruker og forklar hvordan det funger. Eksempel:

Et bilde som inneholder tekst

Automatisk generert beskrivelse  
Hva gjør **f**, **open**, **demofile.txt**, **r** og andre ting du bruker i din løsning. Vis hvordan informasjonen blir lagret i textfilen og snakk om løsningen du har valgt her.

## Kompetansemål:

De i blått vil alle være innom, de andre kan dere se om dere får vist noe kompetanse innom selv.

* utforske og vurdere muligheter, utfordringer og konsekvenser ved bruk av informasjonsteknologi i ulike sammenhenger
* drøfte etiske dilemmaer som oppstår som en konsekvens av informasjonsteknologi, både for individ og samfunn
* utforske og vurdere alternative løsninger for design og implementering av et program
* anvende objektorientert modellering til å beskrive et programs struktur
* utvikle objektorienterte programmer med klasser, objekter, metoder og arv
* vurdere og bruke strategier for feilsøking og testing av programkode
* generalisere løsninger ved å utvikle og bruke gjenbrukbar programkode
* vurdere brukervennligheten i egne og andres programmer og foreslå forbedringer
* velge og bruke relevante systemutviklingsmetoder og -verktøy for samarbeid med andre
* gjøre rede for standarder for lagring, utveksling og sikring av ulike typer data
* bruke programmering til å innhente, analysere og presentere informasjon fra reelle datasett

# Vurdering / innlevering

Prosjektet blir levert på teams i form av en zip-fil med alt som trengs for at lærer skal kunne teste programmet. Eventuelle andre kommentarer du ønsker å vise skal ligge i Word-filen. Dette prosjektet har full karakter og vil telle frem mot vinter- og sommerkarakter.

**Eksempel:**

**elevnavn\_prosjekt2.zip**

fil1.py

fil2.py

fil3.py

fil4.py

tekstfil.txt

**prosjekt2.docx**

**Frist: 18 desember, men kan hende jeg endrer til etter jul. Skal se an arbeidsmengde og andre innspill.**

**Måloppnåelse / Kjennetegn**

6: Alle krav er overholdt og alt er svart på med god kontroll og oversikt over egen kode. Kan ha noen små feil i programmet, men selve OOP-en skal være laget på en god oversiktlig måte og alt som skal inn i prosjekt2.docx skal være svart på en god måte.

5: De fleste krav er overholdt. Kan ha noen større feil eller problemer med koden/kravene, men programmet skal virke etter hensikten

4: Eleven viser forståelse for OOP og andre ting som funksjoner, valgsetninger, løkker og annet, men har ikke fått til oppgaven i seg selv.

3: Programmet virker, men ikke løst med OOP. Eleven har god kontroll på alt utenom OOP

2: Programmet virker for det meste ikke og det mangler OOP i løsningen. Eleven har vist litt forståelse for elementer vi lærer i faget som for eksempel IF, For og funksjoner

Stryk: Ikke levert oppgave eller løst alt feil