Øving 10: Prosjekt del 2

Læringsmål

Dere skal lære hvordan å bygge videre på et program som dere selv har lagd. Dere skal erfare hvordan det er å bygge videre på eksisterende programvare heller enn å starte fra bunnen av.

Godkjenning

Denne øvingen godkjennes i de samme gruppene som øving 9, og på samme vis som øving 9.

Prosjektoppgave: Avtalebok utvidelser

Dere skal utvide avtaleboka slik at avtalene har kategorier samt at dere lagrer mer informasjon om stedet enn en tekst. En avtale kan ha flere kategorier. For eksempel en forelesning i DAT120 kan ha både kategorien «Forelesning» og kategorien «DAT120».

Deloppgaver

a) Denne oppgaven bygger direkte på øving 9. Dere velger om dere gjør den i samme mappe og repository som øving 9 eller ikke. Hvis ikke må dere kopiere over alt innholdet fra øving 9.
b) Fordel de resterende deloppgavene på personene i gruppa på samme måte som for øving 9.
c) Lag en klasse Kategori. Avtaler kan ha kategorier, og man vil ofte filtrere avtaler på kategori. En kategori har minimum en id, et navn og en prioritet. Prioritet er et tall som har 1 som default verdi. Lag en __str__ metode for Kategori-klassen som returnerer en streng som inneholder alle disse tre verdiene.
    a. **Frivillig:** I stedet for at strengen inneholder et tall for prioritet, bruk «Vanlig» for 1, «Viktig» for 2 og «Svært viktig» for 3.
d) Lag en funksjon som leser inn en ny kategori til systemet fra brukeren. Funksjonen skal bruke input-setninger til å spørre brukeren om id, navn or prioritet for kategorien. Funksjonen skal returnere et objekt av typen kategori.
e) Enten lag funksjoner som lagrer og leser inn kategorier fra en egen kategori-fil, eller utvid funksjonene fra øving 9 slik at de også lagrer og leser inn kategorilista. Avgjør om kategoriene skal lagres i samme fil som avtalene eller i sin egen fil.
f) Sjekk funksjonen for å skrive ut alle avtalene inkludert indeksene i lista fra øving 9 deloppgave g. Kan den brukes uforandret også for å skrive ut ei liste med kategorier? Hvis ja, bruk den, og kanskje gi den et nytt navn for å vise det mer generelle formålet med funksjonen nå. Ellers må dere enten lage en egen funksjon for å skrive ut ei liste med kategorier eller modifisere funksjonen for avtaler fra øving 9 slik at den også kan brukes for kategorier.
g) Lag en klasse for et sted. Et sted skal ha en id, et navn og en adresse (gateadresse, postnummer og poststed). Lag en __str__ metode for sted som returnerer en streng som inneholder denne informasjonen.
    a. **Frivillig:** Bare id og stedsnavn er obligatorisk, de andre egenskapene skal ha None som default verdi. I __str__ metoden sin returverdi, ta med bare de egenskapene som ikke er None.
h) Lag en funksjon som leser inn et nytt sted til systemet. Funksjonen skal bruke input-setninger til å spørre brukeren om id, navn og adresse. Funksjonen skal returnere et Sted-objekt for stedet.

i) Enten lag funksjoner som lagrer og leser inn steder fra en egen sted-fil, eller utvid funksjonene fra øving 9 slik at de også lagrer og leser inn stedlista. Avgjør om stedene skal lagres i samme fil som avtalene eller i sin egen fil.
j) Enten sjekk funksjonen fra øving 9 oppgave g for å skrive ut alle avtaler om den også kan brukes for steder, eller skriv en ny funksjon som skriver ut alle stedene i ei liste.
k) Utvid klassen Avtale fra øving 9 slik at den også inneholder en liste med kategorier. En avtale skal kunne være med i flere kategorier. Lag en metode legg_til_kategori som legger til en ny kategori til avtalen. Metoden skal ta et Kategori-objekt for kategorien som parameter.
l) Endre funksjonen som lagrer avtaler til fil slik at den også lagrer kategori-lista samt lagrer sted korrekt som et objekt heller som en streng. Endre funksjonen som leser inn avtaler fra fil slik at den først leser inn kategoriene og stedene og deretter leser inn avtalene.
    a. Hint: Når dere lagrer kategori-lista til en avtale til fil, kan dere lagre kategori-lista som ei komma-separert liste med id-ene til kategoriene. Når dere laster inn dataene, last inn kategoriene før avtalene. Når dere deretter leser inn kategori-lista til avtalene, ta hver ID i lista over kategorier og finn kategori-objektet med den ID-en i hele lista over kategorier. Slik kan dere rekonstruere kategori-lista fra ei tekstfil. Gjør tilsvarende for sted-objektene.
m) Legg til menyvalg i menyen fra øving 9 oppgave l for å legge til kategorier og steder til i systemet. Systemet må ha egne lister eller dictionaries for kategorier og steder som er registrert. Et dictionary kan bruke id som nøkkel.
n) Endre funksjonen for å legge inn avtaler fra øving 9 oppgave f slik at du i stedet for å skrive inn sted velger fra lista over registrerte steder. Alternativt kan du skrive inn et sted som ikke er i lista. Da vil dette stedet bli registrert i systemet med id og stedsnavn men uten adresse.
o) Lag et nytt menyvalg i menysystemet for å legge en kategori til en avtale. Funksjonen skal skrive ut kategori-lista og la brukeren velge hvilken kategori ved å oppgi indeksen til kategorien i lista.
    a. **Frivillig:** La brukeren oppgi en liste med indekser med enten mellomrom eller komma mellom for å legge til flere kategorier på en gang.
p) Lag et menyvalg for å finne alle avtaler som foregår på et bestemt sted. Skriv ut lista over steder. Gå gjennom alle avtalene og sjekk stedet den foregår på. Skriv ut avtalen hvis den foregår på det angitte stedet.
q) Endre __str__ metoden til avtalene slik at den inkluderer navnene til alle kategoriene som avtalen har samt at den inkluderer navnet som ligger i sted-objektet som ligger i avtalen.
r) **Frivillig:** Legg til en variabel «over-kategori» i klassen kategori. Kategorier kan være medlemmer i andre kategorier. For eksempel kan man ha en avtale-kategori «Forelesning» som er med i kategorien «Studier». Lag en metode i klassen kategori som skriver ut alle over-kategoriene. Så hvis «forelesning» har over-kategorien «studier» som har over-kategorien «UiS», så skal denne metoden for kategorien forelesning skrive ut «forelesning – studier – UiS».
s) **Frivillig:** Prioriteten til en avtale. En avtale har prioritet lik den høyeste prioriteten til alle kategoriene den er medlem av. Lag en metode som regner ut prioriteten til en avtale.
t) **Frivillig:** Det kan være enklere å lagre mange objekter med referanser til hverandre med binærfiler heller enn med tekstfiler. For å lagre en mengde objekter til fil bruker man vanligvis pickle-modulen. Denne er ikke pensum i faget. Derfor er det frivillig å teste ut denne modulen for å lagre kategorier, steder og avtaler til ei binærfil i stedet for å bruker id-er i ei tekstfil.
u) **Frivillig:** Lag et system i kategori-klassen og sted-klassen som automatisk lager unike id-er til kategorier og steder. En måte å gjøre dette på er å ha en klasse-variabel neste_id som starter

på 0. Når dere lager et nytt objekt setter dere id-en til objektet lik neste_id og øker neste_id med 1. Når dere leser inn fra fil og dermed setter id-ene fra fila så sjekk neste_id variabelen. Er neste_id lavere enn id-en som er brukt nå så sett neste_id lik en høyere enn nåværende id. Denne oppgaven er frivillig siden klasse-variabler ikke er pensum i DAT120.