Tværfaglig Auktionshus opgave beskrivelse

Dette er en større tværfaglig opgave, som binder Database Programmering, GUI og OOP sammen. Det betyder at ud over de normale timer i de fag, så bliver i også evalueret i ud fra jeres løsning af denne opgave.

Opgaven er delt op i del opgaver for at gøre det nemmer at overskue, dette dokument anbefales at bruges som opslagsværk til læse detaljerne af opgaverne. For overblik over de individuelle opgave refereres der til kravspecifikationen for opgaven.

Se indholdsfortegnelsen for en oversigt over opgave delene.

Eventuelle uklarheder i opgaveformuleringen forventes afklaret som en del af opgaveløsningen.

Indholdsfortegnelse

OOP evalueringskrav

Lav et passende klassedesign der opfylder ovenstående krav, inkl. et passende klassehierarki der reducerer koderedundans. Brug passende Access modifiers og datatyper. Navngivning af klasser, egenskaber og metoder skal være korte og præcise. Det står det Jer frit for om I vil foretage data validering ud over de beskrevne krav. Endelig står det Jer frit for at tilføje yderligere funktionalitet (klasser, metoder og properties) ud over de eksplicit angivne krav, men sørg for at opfylde kravene først.

Der vil dog være fokus på at I laver et implementations klassedesign(detaljeret), samt at I sørger for at jeres kode er læsbar. Det står jer frit for om i vil skrive på Dansk eller Engelsk.

Krav til program og dokumentation

- **Krav til afleveret opgave**
    1. Nedarvning
    2. Polymorfi
    3. Function Overload
    4. Gør brug af statiske variable
    5. Én eller flere af jeres klasser, skal implementere én eller flere statiske metoder
    6. Function Override
    7. Interfaces (brug af disse)
    8. Delegates (function pointer/callback)
    9. Løs kobling mellem klasser/objekter (Dependency Injection)
    10. Threads (brug af disse)
    11. Lave UML klassediagrammer (godt program til dette på hjemmesiden : draw.io)
- Kildeteksten samt UML diagram af det udviklede program skal afleveres i som zip fil i moodle.
- Hver offentlig/internal type skal placeres i en .cs fil med samme navn som typen (eks.: Køretøj
    - klasse => Køretøj.cs, LastBil klasse => LastBil.cs, osv).

Del 1 Køretøj model

I denne opgave skal der laves et auktions-system der administrerer salg af forskellige køretøjer (inspireret af lignende systemer som f.eks. autocom.dk). Systemet skal administrere følgende typer køretøjer: Personbiler, lastbiler og busser. I det følgende beskrives køretøjernes fællestræk samt deres forskelle.

Køretøj

Følgende egenskaber er fælles for køretøjer

- **Navn** (f.eks., "WV Polo" eller "Skoda Octavia")
  - Navn må ikke være null. Forsøg på at tildele en null værdi skal udløse en passende exception.
- **Km** (angiver hvor mange kilometer køretøjet har kørt i dets levetid)
  - Km må ikke være et negativt tal. Forsøg på at tildele en negativ værdi skal udløse en passende exception.
- **Registreringsnummer** (nummerplade)
  - Et registreringsnummer skal bestå af to bogstaver (i denne opgave accepteres alle bogstaver, ikke kun bogstaverne A-Z) efterfulgt af fem cifre. Forsøg på at tildele et ulovligt registreringsnummer skal udløse en brugerdefineret exception.
  - Når et registreringsnummer aflæses, skal de to første og de to sidste tegn skjules. Registreringsnummer XY12345 skal derfor vises som **123**.
- **Årgang** (angiver hvilket år køretøjet er indregistreret)
  - Årgang skal være read-only, dvs. efter instantiering må værdien ikke kunne ændres.
- **NyPris** (angiver køretøjets oprindelige købspris)
  - NyPris må ikke være et negativt tal. Forsøg på at tildele en negativ værdi skal håndteres ved at NyPris tildeles værdien 0.
- **Trækkrog** (ja/nej)
  - I denne opgave skal personbiler til erhverv være udstyret med en trækkrog. For alle andre køretøjer er det frivilligt at have trækkrog.
- **KørekortType** (angiver hvilket kørekort der kræves for at føre køretøjet)
  - De mulige kategorier er A, B, C, D, BE, CE, DE. Se subklasserne for detaljer.
- **Motorstørrelse** (som angivet ved slagvolumen i liter)2
  - I denne opgave vil forskellige typer køretøjer have forskellige tilladte motorstørrelser:
    - Den tilladte motorstørrelse for personbiler er 0.7 - 10 l.
    - Den tilladte motorstørrelse for lastbiler og busser er 4.2 - 15 l.
  - Forsøg på at tildele en værdi uden for den tilladte rækkevidde skal udløse en passende exception.
- **Km/l** (Angiver hvor mange kilometer køretøjet kan køre på en liter brændstof)
- **Brændstof** (Angiver den brændstoftype som køretøjet anvender)
  - Mulige værdier er Diesel, Benzin, Elektrisk eller Hydrogen.
  - Lastbiler og busser kan kun køre på diesel. Alle andre køretøjer kan fås med begge brændstoftyper.
- **Energiklasse** (angiver hvor brændstoføkonomisk køretøjet er)
  - Energiklassen er en udregnet egenskab der afhænger af køretøjets brændstof, km/l eller kWh og årgang.
  - For køretøjer før 2010 gælder følgende energimærker: (x <= km/l < y betyder at km/l skal være imellem x og y.)
    - Hvis motorens brændstof-type er diesel, udregnes energiklassen som:
      - A klasse: km/l >= 23
      - B klasse: 18<= km/l < 23
      - C klasse: 13 <= km/l < 18
      - D klasse: km/l < 13
    - Hvis brændstof-typen er benzin, udregnes energiklassen som:
      - A klasse: km/l >= 18
      - B klasse: 14 <= km/l < 18
      - C klasse: 10 <= km/l < 14

            - D klasse: km/l < 10
    - For køretøjer efter 2010 gælder følgende energimærker:
        - Hvis motorens brændstof-type er diesel, udregnes energiklassen som:
            - A klasse: km/l >= 25
            - B klasse: 20 <= km/l < 25
            - C klasse: 15 <= km/l < 20
            - D klasse: km/l < 15
        - Hvis brændstof-typen er benzin, er energiklassen:
            - A klasse: km/l >= 20
            - B klasse: 16 <= km/l < 20
            - C klasse: 12 <= km/l < 16
            - D klasse: km/l < 12
    - Hvis Køretøjet brændstof er elektrisk eller hydrogen skal det have Energy klasse A.
- **ToString()**
        - Alle køretøjer skal redefinere objekt klassens ToString() metode for at give en sigende beskrivelse af køretøjet.

Bus

- **Højde** (Angiver køretøjets højde)
- **Vægt** (Angiver køretøjets vægt)
- **Længde** (Angiver køretøjets længde)
- **Antal siddepladser**
- **Antal sovepladser**
- **Toilet** (Indikerer om bussen indeholder toilet-faciliteter)
- **Kørekorttype**
    - Er som udgangspunkt D. Hvis bussen har trækkrog kræver det imidlertid et DE kørekort.

Lastbil

- **Højde** (Angiver køretøjets højde)
- **Vægt** (Angiver køretøjets vægt)
- **Længde** (Angiver køretøjets længde)
- **Lasteevne** (Angiver maksimal lasteevne i kg)
- **Kørekorttype**
    - Er som udgangspunkt C. Hvis lastbilen har trækkrog kræver det imidlertid et CE kørekort.

Personbil

- **Antal sæder**
    - Der findes to slags personbiler: Personbiler til privat brug og personbiler til erhverv. Personbiler til erhverv er kendetegnet ved at de har akkurat to sæder, mens personbiler til privat brug kan have mellem 2 og 7 sæder.
- **Bagagerums dimensioner** (dette skal være en type med længde, bredde og højde)
- **Kørekorttype**
    - Kørekorttypen for personbiler er som udgangspunkt B.
    - Personbiler til erhverv kan dog også kræve et BE kørekort.

Personbil til erhverv

- **Sikkerhedsbøjle** (ja/nej)
    - Angiver hvorvidt der findes en sikkerhedsbøjle bag førersædet.
- **Lasteevne**
    - Angiver hvor meget påhængsvægt bilen kan køre med.
    - Hvis vægten overstiger 750 kg. kræver det et BE kørekort

Personbil til privat brug

- **Isofix beslag** (ja/nej)
    - Isofix beslag giver en forbedret forankring af autostole ift. selemontering.

Del 2 Database

Database opsætnings script

Database styring for user og auction

Id'er opdateres fra databasen

Del 3 Kunder med købere og sælgere

I skal lave et auktionshus hvor køretøjer kan sælges igennem. Auktionshuset formidler kontakten

mellem en sælger og potentielle købere.

Alle Kunder

Alle kunder skal have

- **Brugernavn**
  - Brugernavne skal være unikke og skal derfor valideres i forhold til alle andre tidligere brugernavne.
- **Password**
  - Password skal valideres i forhold til sikkerheden. Passwords skal gemmes som et Hash og må ikke gemmes i læselig tekst.

Sælger

En sælger er karakteriseret ved følgende:

- **Saldo**
  - Når en sælgers køretøj bliver solgt, så tilføjes køretøjets salgspris til sælgerens saldo.
- **PostNummer** (udtrykt som et heltal)
- **ModtagNotifikationOmBud**(...)5
  - Dette er en metode der kan bruges til at notificere sælgeren om at der er afgivet et interessant bud på et køretøj, som sælgeren har til salg. Metoden kan nøjes med at registrere (f.eks. udskrive) information om det pågældende køretøj samt buddets størrelse.

Køber

- **Saldo**
  - Når et køretøj købes, trækkes køretøjets købspris fra saldoens eksisterende beløb.

Firmaer og privat personer

Sælgere og købere fordeler sig på to typer: privatpersoner og firmaer. Begge typer kan optræde i begge roller; dvs. en privatperson/ et firma kan være både sælger og køber.

En privatperson er kendetegnet ved at have et unikt CPR nummer, mens et firma er identificeret ved at have et unikt CVR nummer. Et firma har desuden kredit, der tillader firmaets saldo at gå i minus. Som eksempel, så vil et firma med en saldo på 40.000 kr., og en kredit på 30.000 kr. være i stand til at købe køretøjer der koster op til 70.000 kr. En privatpersons saldo må derimod ikke gå i minus, og en tilsvarende person kan derfor højst købe køretøjer til en værdi af 40.000 kr.

Del 4 Administration af køretøjer i Auktionshus

Auktion

En samling af nødvendig auktions data, som kan samle alt data omkring en auktion. Det skal indeholde hvad der nødvendigt for Auktionshuset.

Auktionshus

I skal lave et auktionshus hvor køretøjer kan sælges igennem. Auktionshuset formidler kontakten mellem en sælger og potentielle købere.

Auktionshus- klassen skal have:

- **Auktioner**
  - Auktionhus skal have en samling af data der binder Køretøjer sammen med Sælger, størrelse på et bud og beskrevet med et auctionsNummer, der kan være andet information i denne samling, hvis der er brug for det.

Auktionshus-klassen skal tilbyde følgende metoder til at administrere salg af køretøjer:

public int SætTilSalg(Køretøj k, Sælger s, decimal minPris)

- Et køretøj sættes til salg via ovenstående metode, hvor k er køretøjet der sættes til salg, s er sælgeren, og minPris angiver mindstebeløbet før sælger vil overveje at sælge. Når der kommer et bud der overstiger den angivne minimumpris, skal sælgeren notificeres herom via sælgers ModtagNotifikationOmBud(..) metode. Endeligt skal der returneres et unikt auktionsnummer der kan bruges til senere at referere til det givne køretøj der er sat til salg.

public int SætTilSalg(Køretøj k, Sælger s, decimal minPris, notifikationsMetode)

- Som ovenfor, men her skal notifikationer i stedet sendes til den angivne notifikationsMetode.

public bool ModtagBud(Køber køber, int auktionsNummer, decimal bud)

- Potentielle købere skal kunne afgive bud på et bestemt køretøj som identificeret vha. det angivne auktionsnummer. Metoden returnerer en bool der angiver om buddet accepteres. Et bud accepteres kun hvis køberen har penge nok til at kunne købe køretøjet, og hvis buddet er større end det hidtil største bud. I tilfælde af, at buddet accepteres, og samtidig overstiger sælgers angivne minimumpris, så skal sælger notificeres om at der er afgivet et interessant bud.

public bool AccepterBud(Sælger sælger, int auktionsNummer)

- Ovenstående metode skal implementeres for at afslutte en handel med et bestemt auktionsnummer. Der skal først laves et tjek på at den angivne sælger er identisk med den sælger der har sat køretøjet til salg. Såfremt sælgerens identitet er i orden, så afsluttes salget ved at købssummen trækkes fra købers saldo (den køber der har afgivet det største accepterede bud). På dette tidspunkt skal køretøjet ikke længere figurere som værende til salg i auktionshuset, men skal i stedet overgå til en liste af solgte køretøjer. Listen af solgte køretøjer skal kunne gennemløbes uden for AuktionsHus klassen. Metodens returværdi angiver om handlen blev succesfuldt gennemført.

public Auktion FindAuktionMedID(uint AuktionID)

- Denne metode skal finde et bestemt auktion objekt fra listen over auktioner i AuktionsHuset. Det er valgfrit om man selv laver en søge algoritme eller om man bruger en fra .NET. Hvis i finder en på internettet, så husk at inkludere en henvisning til siden hvor den blev fundet.
- Denne metode skal refactores til at være trådet, det er vigtigt at tænke det ind hvis egen algoritme implementeres.

Del 5 GUI

Del 6 Ekstra opgave: Søgning i Auktionshus

Udover ovenstående funktionalitet til at administrere salg af køretøjer, så skal auktionshuset også understøtte følgende søgefunktioner blandt køretøjerne der er til salg:

1. Find køretøjer hvis navn indeholder en angivet søgestreng.
2. Find køretøjer der har et minimum angivet antal siddepladser samt toiletfaciliteter.
3. Find køretøjer der kræver stort kørekort (kategori C, D, CE eller DE) og vejer under en angivet maksimalvægt.
4. Find alle personbiler til privatbrug som har kørt under et angivet antal km, og hvor minimum salgsprisen samtidig ligger under et angivet beløb. Køretøjerne skal returneres i sorteret rækkefølge efter antal kørte km.
5. Find alle køretøjer hvor køretøjets sælger er bosiddende inden for en bestemt radius af et angivet postnummer. I denne forbindelse kan radius blot anskues som et tal der skal lægges til/trækkes fra postnummeret. F.eks. vil en søgning efter køretøjer inden for en radius af 1500 fra postnummer 8000, inkludere alle køretøjer hvor sælgers postnummer ligger mellem 6500 og 9500.