Opgavesæt 5

Når der afleveres løsningsforslag, så pak alle programmets filer, mapper og undermapper – eller projektet som Visual Studio kalder det – ned i én komprimeret fil, altså .suo-filen, .sln-filen plus tilhørende mapper og undermapper..

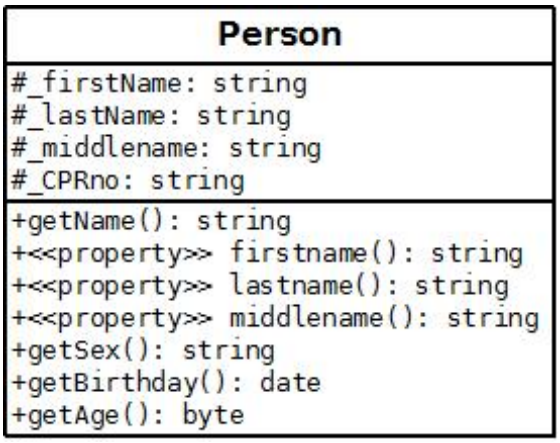
Man kan hente pakke/komprimeringsprogramet 7-zip gratis her: http://www.7-zip.org/, og det kan håndtere de mest brugte komprimerings-formater .zip, .rar og .7z.

Hvis du kan aflevere på SmartLearning Online inden 6. maj er det fint ☺

Opgave 5.1 – Person klassen

Lav en Person klasse som beskrevet nedenfor:

- 4 protected members/variabler
- *getName* metoden skal returnere personens fulde navn
- Properties til *get* og *set* af firstname, lastname og middlename
- *getSex* metoden skal returnere 'mand' eller 'kvinde'
- *getBirthday* metoden skal returnere en dato (datatypen DateTime)
- *getAge* metoden skal returnere personens alder som datatypen Byte (kan holde værdier i intervallet 0-255)



Som du måske ved, er et CPR nummers opbygget sådan:

- 10 cifre i formatet ***ddmmyynnnc***, hvor
  - ***ddmmyy*** er personens fødselsdag, f.eks. 150879
  - ***nnn*** er et løbenr (ikke helt; men det er uden betydning her)
  - ***c*** er et kontrolciffer (modulus-11)
    hvis ***c*** er ulige er personen en mand, hvis ***c*** er lige er personen en kvinde

Klassen skal have 2 konstruktører

- En der tager et CPR nummer som parameter
- En anden, som tager firstname, middlename, lastname og CPRnummer som parametre

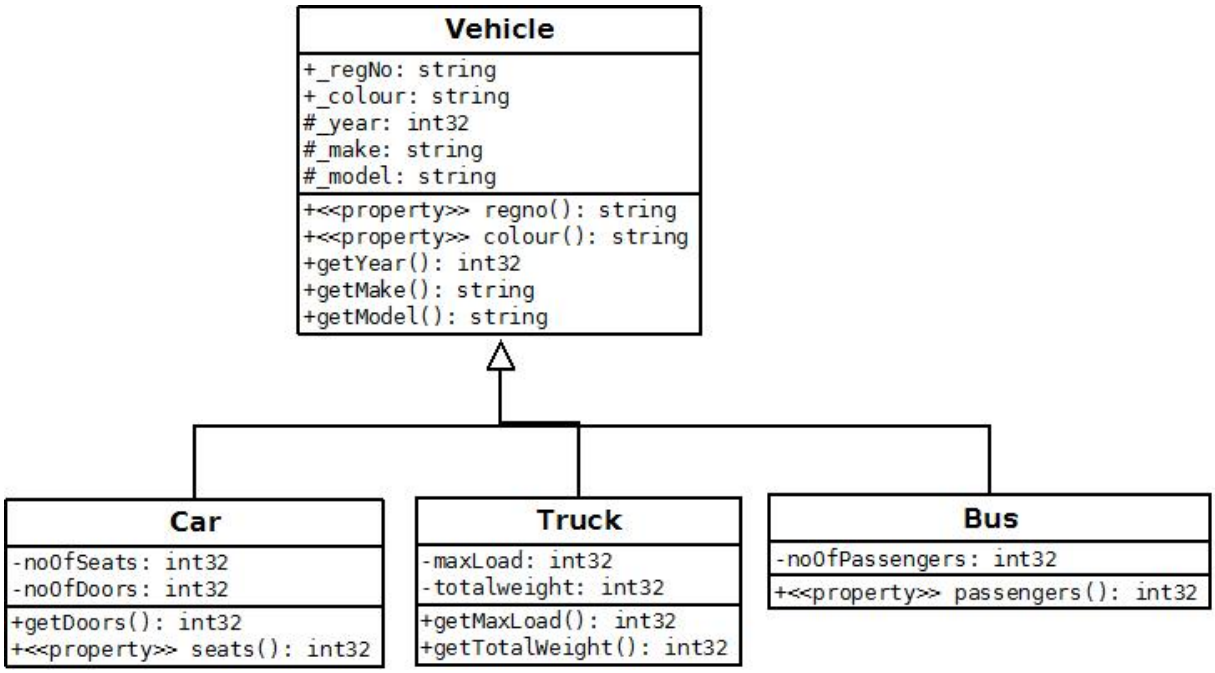
Implementer klassen i en class fil ved navn Person.cs, og lav et program med en GUI der demonstrerer Person klassens properties og metoder – der må gerne anvendes flere Person objekter.

Til denne opgave kan genbruges noget fra opgave 1.2, ligesom demo programmet ClassDemo kan hjælpe lidt på vej.

Bogens side 201-210 omhandler Class members og properties
Bogens side 184-185 omhandler konstruktører

Opgave 5.2 – Vehicle klassen, igen + arv

I opgave 4.4 lavede vi *Vehicle* klassen, og vi skal nu have lavet 3 klasser *Car*, *Truck* og *Bus*, som alle skal arve fra *Vehicle* klassen (se nedenfor)



Car klassen skal have members, properties og metoder som vist, og skal implementere en konstruktør, som tager parametrene regno, colour, year, make, model, noofdoors og noofseats.

Truck klassen skal have members, properties og metoder som vist, og skal implementere en konstruktør, som tager parametrene regno, colour, year, make, model, maxload og totalweight.

Bus klassen skal have members, properties og metoder som vist, og skal implementere en konstruktør, som tager parametrene regno, colour, year, make, model og noofpassengers.

Implementer klasserne, enten i samme fil som *Vehicle* klassen, eller i separate .cs-filer, og lav et program med en GUI der demonstrerer *Car*-, *Truck*- og *Bus*-klassernes properties og metoder – der må gerne anvendes flere objekter af de enkelte klasser.

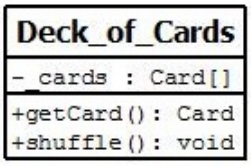
Til denne opgave kan videoen 'Arv' hjælpe lidt på vej.

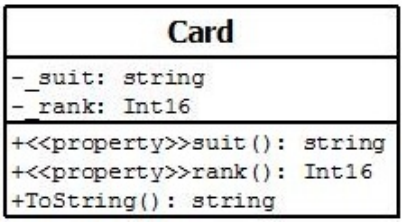
Bogens side 165-167 handler om arv

Opgave 5.3 – for spillefugle (og andet godtfolk)

Hele opgave 5.3 er en ekstra-opgave til dem der måtte have lyst – det er ikke noget I behøver lave.

Lav 2 klasser som vist nedenfor:





Card klassen skal have 2 members: *_suit* (farve) og *_rank* (værdi).

Konstruktøren skal tage to parametre, og skal sætte værdien af _suit og _rank – dog med følgende betingelser:

- *_rank* skal være et heltal mellem 2 og 14 (esset har værdien 14).
  Enhver anden værdi skal afvises ved at konstruktøren kaster en exception, f.eks. sådan

  ```
  if (rank < 2 || rank > 14)
      throw new Exception(rank + ” er en ugyldig værdi”);
  else
      _rank = rank;
  ```

- *_suit* skal være enten ’Klør’, ’Spar’, ’Hjerter’ eller ’Ruder’
  Enhver anden værdi skal afvises ved at konstruktøren kaster en exception.

Klassen skal have get properties, som kan returnere rank hhv. suit. Disse properties skal ikke have noget set-del, da et kort jo ikke kan skifte farve eller værdi.

Klassen skal også have en ToString metode, som kan returnere et kort, f.eks. ’Hjerter 7’, Spar konge’. Dvs. ToString metoden skal konvertere _rank for billedkort sådan: 11 skal være ’knægt’, 12 skal være ’dame’ og 13 skal være ’konge’.

Deck_of_Cards klassens private member *_cards* skal være en arraylist med Card objekter.

Konstruktøren skal befolke arraylisten _cards med 52 forskellige Card objekter – du bestemmer hvordan det skal implementeres; men en løkke eller to er jo nok ikke til at komme udenom ☺

getCard metoden skal simulere at man giver et kort fra ’bunken’ (Deck’et). Metoden skal returnere det næste Card-objekt i arraylisten _cards. Heri ligger implicit at metoden faktisk skal fjerne det kort, der er ’givet’ af metoden.

Og så en lille ekstra-ekstra-opgave til nørderne ☺:

shuffle metoden skal blande kortene – hvordan er op til dig; men det skulle undre mig meget om ikke der kan findes flere eksempler og beskrivelser på nettet.

Lav et program med en GUI der demonstrerer brugen af Card- og Deck_of_Cards-klassen.

Hvis nu du bliver grebet af det – eller bare kan li’ at spille kort, kunne opgaven jo udvides til at simulere f.eks. et spil 31, et spil 500, et spil whist eller…. eller ….. nå, det må nok blive efter eksamen er overstået ☺

Bogens side 168-169 handler om Relationship between objects

Eksemplet side 224-232 i bogen indeholder en stor del af hvad der skal bruges til denne opgave