Prøve

i fagene

Programmering & Design

Alle skriftlige materialer, pc'er, bærbare computere og internet ressourcer er tilladt til eksamen.

Mobiltelefoner og kommunikation med andre personer (inkl. chatbots), bortset fra kommunikation med "eksaminatorer", er forbudt.

Du må ikke benytte/gemme din løsning på eksterne netværksdrev/værter som GitHub, Facebook o. lign.

Ved prøvens afslutning skal du lægge din løsning op i Wiseflow.

Prøven varer 4 timer og efterfølges af 1 times evaluering.

Prøvesættet består af 7 opgaver (samt en ekstra opgave).
Ud over disse opgaver, kan du kan blive bedt at besvare nogle ekstra spørgsmål omkring dine svar og eventuelle andre valg.

Domæne beskrivelse

Politiet ønsker et nyt system ***FartKontrol*** til at registrere fartkontroller med automatiske fartmålinger.

Systemet ***FartKontrol*** skal anvendes til at registrere fartmålinger på en given lokation og skal kunne udskrive statistik over målingerne

Politiet ønsker at få udviklet et nyt system til at håndtere oplysninger i forbindelse med at politiet udfører fartkontrol på en given lokalitet.

I systemet skal der være en klasse **SpeedMeasurementCatalog**. Klassen skal indeholde:

- **Address** (hvor fartkontrollen foregår, f.eks. "Maglegårdsvej 2, 4000"
- **Zone** (fx. "By", "Motortrafikvej", "Landevej" eller "Motorvej"
- **SpeedLimit** (Fartgrænse) fx. 50, 60, 80, 90 ...

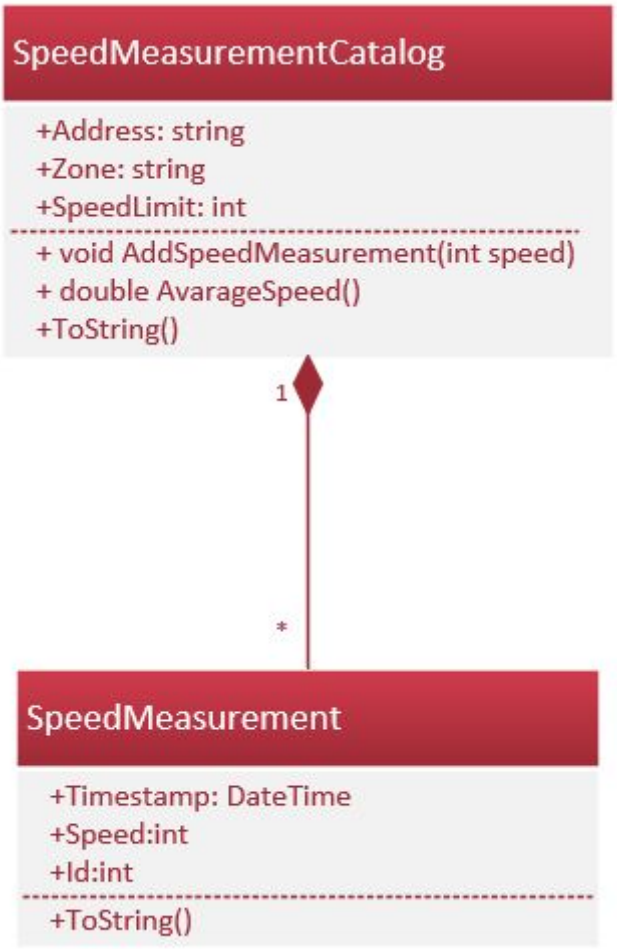
Ligeledes skal der være en klasse SpeedMeasurement, som indeholder oplysninger om den registrerede hastighed og tidspunktet.

Klassen **SpeedMeasurement skal indeholde:**

- **Timestamp** (tid) fx. '10:33:22'
- **Speed** (hastighed) fx. 87
- **Id**

Klassen **SpeedMeasurementCatalog** skal indeholde en liste af **SpeedMeasurement** (s).

Udgangspunktet for systemet er nedenstående design klassediagram.



Opgave 1 Opret klassen SpeedMeasurement

- Opret et nyt ASP.Net Core console Application projekt, **SpeedCheck** (FartKontrol).
- Implementer klassen **SpeedMeasurement** med instans felter, properties og passende konstruktør(er).
- Klassen skal ligeledes have en ToString metode.

Opgave 2 Opret klassen SpeedMeasurementCatalog

SpeedMeasurementCatalog skal indeholde en liste (af typen List) af **SpeedMeasurement** objekter. Implementer denne. Denne liste skal instantieres i konstruktøren.

- **SpeedMeasurementCatalog** klassen skal ligeledes have 3 properties (Address, Zone og SpeedLimit). Implementer disse og initialiser dem gennem klassens konstruktør.

- Vis i Main metoden, hvordan der kan oprettes et objekt af typen **SpeedMeasurementCatalog**.

- Implementer metoden ***AddSpeedMeasurement(int speed)***.
    - *AddSpeedMeasurement* metoden skal tage en hastighed (speed) som argument og benytte speed til at oprette et nyt **SpeedMeasurement** objekt. Objektet skal tilføjes til listen i **SpeedMeasurementCatalog**.
      Bemærk: Du kan evt. benytte *DateTime.Now* til initialisering af propertien *TimeStamp* i konstrutøren.
    - Afprøv metoden med 3 metodekald fra Main. Vis at der oprettes 3 objekter af typen **SpeedMeasurement** og at de er tilføjet listen i **SpeedMeasurementCatalog**.

- Implementer metoden ***AverageSpeed()***
    - *AverageSpeed* metoden skal beregne gennemsnitshastigheden af samtlige **SpeedMeasurement** objekter i listen.
    - Afprøve metoden fra main.

- Implementer ToString metoden
    - Metoden skal udskrive oplysninger om properties i **SpeedMeasurementCatalog** og desuden en formateret præsentation af samtlige fartregistreringer (SpeedMeasurement) i listen.
    - Afprøve metoden fra Main.

Opgave 3

Tegn et sekvensdiagram for et kald af metoden *AddSpeedMeasurement*.
Udgangspunktet for sekvensdiagrammet er Main metoden og sekvensdiagrammet skal kun vise et enkelt kald af **AddSpeedMeasurement** og vise, hvorledes der oprettes et objekt af typen **SpeedMeasurement** og at det tilføjes til listen i **SpeedMeasurementCatalog**.

Opgave 4. Udvid klassen SpeedMeasurementCatalog

Der skal implementeres yderligere metoder i klassen **SpeedMeasurementCatalog**.

- Der skal implementeres en metode **NoOfOverSpeedLimit**, som finder antal målinger over fartgrænsen (angivet ved propertien SpeedLimit).

  int NoOfOverSpeedLimit() //Returnerer antal målinger over fartgrænsen

    - Metoden skal laves vha. et foreach loop
    - Afprøv metoden ved at kalde den fra Main

- Der skal implementeres en metode **NoOfCutInLicense**, som finder antal målinger der resulterer i klip i kortet, dvs at fartregistreringen er 30% over det tilladte.

  int NoOfCutInLicense() //Returnere antal målinger der giver klip i kortet

  - Metoden skal laves vha. et for loop
  - Afprøv metoden ved at kalde den fra Main
- Beskriv forskellen mellem et for- og et foreach-loop.

Opgave 5 Exceptions

Hvis argumentet (speed) til **AddSpeedMeasurement(int speed)** er 0, negativ, eller større end 300 - skal der kastes en ArgumentException() med en passende message.

Udvid **AddSpeedMeasurement**, således at der kastes en exception ud fra ovenstående regler.

Afprøv metoden AddSpeedMeasurement i Main. Tilføj sætninger i Main, som samler exceptionen op og udskriver information om fejlen. Prøv med 3 forskellige speed værdier (både lovlige og ulovlige) som argument til AddSpeedMeasurement.

Opgave 6 Udvid design klasse diagrammet

For hver fartregistrering skal der tilknyttes et **Køretøj**. Vis hvorledes dette kan modelleres i designklassediagrammet.

Ved at anvende nedarvning skal det vises, hvorledes der kan forekomme forskellige typer af køretøjer, **Lastvogn** og **Personbil**.

For en lastvogn skal der registreres tilladt last, MaxLoad, og for en personbil skal der registreres antal personer, NoOfPersons.

Der skal ligeledes registreres et registreringsnummer. I hvilken klasse skal det registreres?

Udvid design klassediagrammet med de ekstra klasser.

Opgave 7 Implementation af arv

På baggrund af dit design klassediagram i opgave 6, skal du implementere klasserne **Køretøj**, **Personbil** og **Lastvogn**.

Ekstraopgaver (vil normalt ikke være med hvis opgaven blev givet til prøven):

Opgave 8 Udvid klassen SpeedMeasurementCatalog

Implementer nedenstående metoder:

int NoOfConditionalRevocation()
Metoden skal bestemme og returnere antal målinger, der som minimum resultere i betinget frakendelse af kortet.

Bemærk:
Man får en betinget frakendelse af kørekortet (alm. bil) ved:

- En overskridelse på 30% eller derover på en motorvej, hvor det er tilladt at køre 130 km/t
- En overskridelse på 60 % eller derover på øvrige strækninger.

int NoOfUnconditionalRevocation()
Metoden skal bestemme og returnere antal målinger, der resultere i ubetinget frakendelse af kortet.

Bemærk:
Man får en ubetinget frakendelse af kørekortet (alm. bil) ved:

- En overskridelse på 100% (dog minimum 100 km/