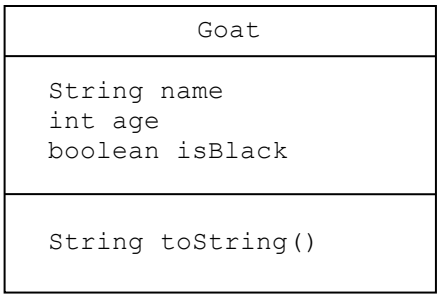
Goat



1. Opret en klasse, *Goat*, der repræsenterer en ged. Klassen er specificeret i UML-diagrammet til højre. De tre feltvariabler skal initialiseres i en konstruktør (via parametre af passende type). Metoden *toString* skal returnere en streng-repræsentation for en *Goat* på formen

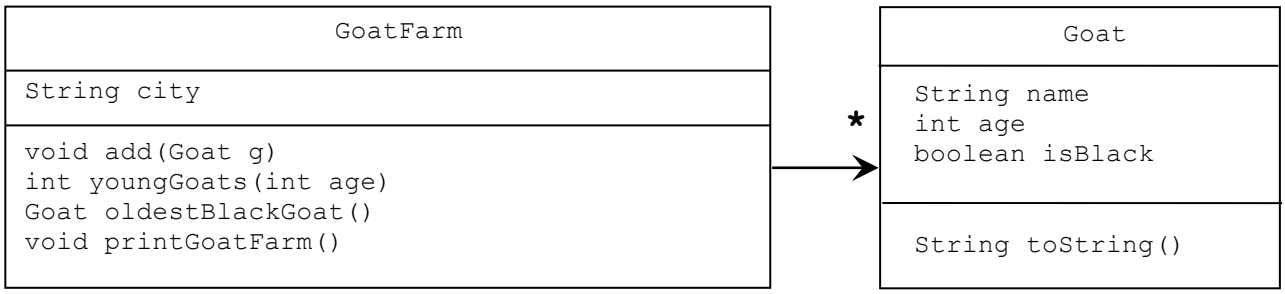
   ```
   "Mads: 4 years, color: black"
   "Anna: 6 years, color: white"
   ```

2. Lav en *Driver*-klasse med en *exam*-metode. Metoden skal være static, have returtype void og være uden parametre.

3. Opret fem velvalgte *Goat*-objekter i *exam*-metoden, via objektreferencer *g1,g2, g3, g4* og *g5,* og udskriv disse vha. *toString*-metoden.

Tilkald tilsynsførende og demonstrer det du har lavet indtil nu.

4. Opret en ny klasse, *GoatFarm*, der repræsenterer en bondegård med geder. Klassen *GoatFarm*, og dens relation til klassen *Goat*, er specificeret i følgende UML-diagram:



5. Programmér metoden *add,* der tilføjer *Goat*-objektet *g* til *GoatFarm*-objektet.

6. Opret et objekt af typen *GoatFarm* i *exam*-metoden i *Driver*-klassen og knyt de allerede oprettede *Goat*-objekter hertil.

7. Programmér metoden *youngGoats*. Metoden skal returnere antallet af geder, hvis alder er mindre end eller lig den angivne alder. Udvid *Goat*-klassen med de nødvendige get-metoder.

8. Afprøv metoden *youngGoats* i *exam*-metoden i *Driver*-klassen.

Tilkald tilsynsførende og demonstrer det du har lavet indtil nu.

9. Programmér metoden *oldestBlackGoat*. Metoden skal returnere den ældste sorte ged. Hvis der ikke findes en sådan ged returneres null. Afprøv *oldestBlackGoat* i *exam*-metoden.

10. Programmér metoden *printGoatFarm*. Metoden skal udskrive den by som bondegården ligger i efterfulgt af alle geder sorteret efter alder (lavest til højest). Hvis to har samme alder sorteres efter farve (sorte først). Afprøv *printGoatFarm* i *exam*-metoden.

Tilkald tilsynsførende og demonstrer din færdige løsning.