INF3120 Tutorial Exercise on OCL - Solutions

1. *Write an OCL constraint on Course restricting the course name to between 10 and 25 characters. Write an OCL constraint on Student stating that a student may only be registered for any courses if they have paid their fees.*

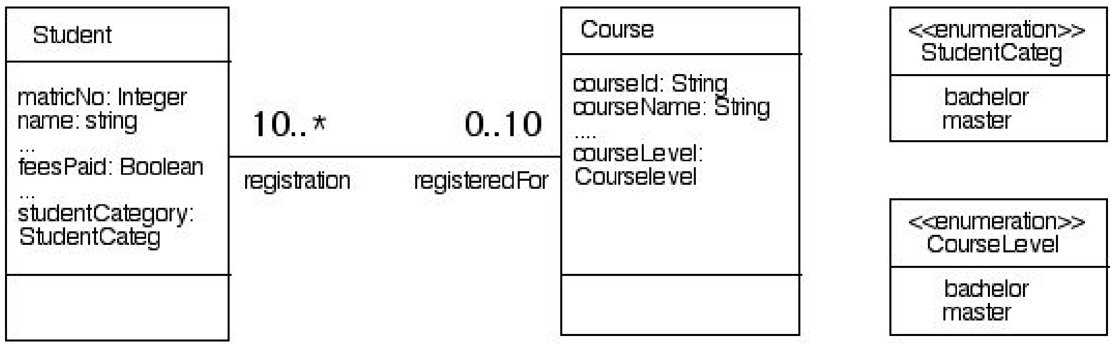
   context Course
   inv: courseName.size() >= 10 and courseName.size() <= 25

   context Student
   inv: feesPaid = false implies registeredFor -> isEmpty

   -- alternatively

   context Student
   inv: not feesPaid implies registeredFor -> isEmpty

2. *Each Bachelor student may be registered for up to 10 courses but a Master student may only register for a maximum of 8 courses (in each case, provided they have paid their fees). There is no upper limit on the number of student registrations for a course but every Bachelor course must have at least 20 registrations and every Master course at least 10 registrations. Add cardinalities to the associations on the above diagram to implement these restrictions as far as possible and write OCL constraints to add more specific restrictions.*



   -- registeredFor has minimum cardinality 0 (fees not paid) and maximum cardinality 10 (for Bachelor) To restrict further for Master students:

   context Student
   inv: studentCategory = StudentCateg::master implies registeredFor -> size() <= 8

   -- registration has minimum cardinality 10 (for Master) and no upper limit. To show higher number required for Bachelor students:

   context Course
   inv: courseLevel = CourseLevel::bachelor implies registration -> size() >= 20

3. *Suppose that there is a simple rule that Bachelor students may only attend Bachelor level courses and Master students may only attend Master level courses.*
   (a) Write OCL constraints for the two basic classes, Student and Course, to express this rule.

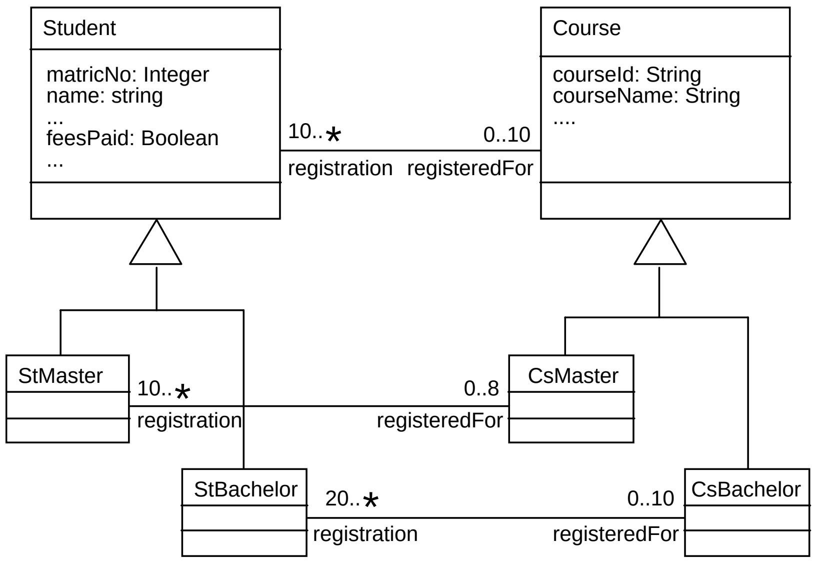
   context Student
   inv: studentCategory = StudentCateg::master implies
   registeredFor -> forall (courseLevel = CourseLevel::master)

   context Student
   inv: studentCategory = StudentCateg::bachelor implies
   registeredFor -> forall (courseLevel = CourseLevel::bachelor)

   context Course
   inv: courseLevel = CourseLevel::master implies
   registration -> forall (studentCategory = StudentCateg::master)

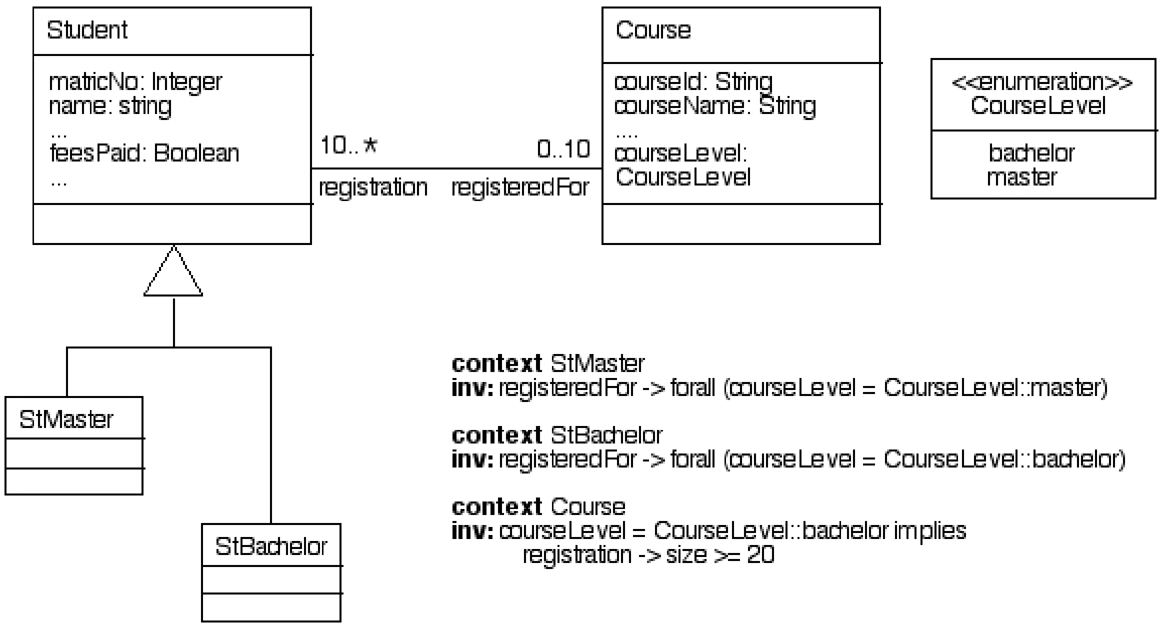
   context Course
   inv: courseLevel = CourseLevel::bachelor implies
   registration -> forall (studentCategory = StudentCateg::bachelor)

   (b) Define subclasses of Student and Course and express these constraints diagrammatically, together with those described in part 2, above.



   There might be a modelling argument for defining subclasses of Student or Course if there are significant differences in the attributes (or operations) required for bachelor and master students or courses. (In our system there is little difference between student or course details at bachelor and master level.)

(c) Consider using subclasses for either Student or Course together with some OCL constraints.



```
context StMaster
inv: registeredFor -> forall (courseLevel = CourseLevel::master)

context StBachelor
inv: registeredFor -> forall (courseLevel = CourseLevel::bachelor)

context Course
inv: courseLevel = CourseLevel::bachelor implies
        registration -> size >= 20
```

4. *The rules are changed so that Master students may register for one or two Bachelor courses as part of their curriculum. Bachelor students are still restricted to registering for only Bachelor level courses.*
   Write an OCL constraint to express these rules, modifying the rules expressed in 3(a) above.

Replace the first constraint in 3(a) with the following:

context Student
inv: studentCategory = StudentCateg::master implies
   registeredFor -> select (courseLevel = CourseLevel::bachelor) -> size() <= 2

Note that this revised constraint is only valid because there are only two values in the enumeration CourseLevel; if the value is not 'bachelor' then it must be 'master'. It would be safer to rewrite the first constraint of 3(a) as:

context Student
inv: studentCategory = StudentCateg::master implies
          registeredFor -> forall (courseLevel = CourseLevel::master
                                        or courseLevel = CourseLevel::bachelor)

and then add the restriction on the number of bachelor level courses specified above.

We also need to remove or revise the fourth constraint in 3(a), the most precise approach would be to write:

context Course
inv: courseLevel = CourseLevel::bachelor implies
          registration -> forall (studentCategory = StudentCateg::bachelor
                                        or studentCategory = StudentCateg::master)