**1.** **A student can view the list of courses (along with other details) offered by any department in any semester.  
invariant** ViewCoursesByDepartment: *Department*.*allInstances*()->*forAll*(dept | *Semester*.*allInstances*()->*forAll*(semester | *dept*.*courses*->*exists*(course | *course*.*semester* = *semester*)));

**2. Semester can be either Fall, Spring or Summer for any year.**context Semester inv:   
self->Degree.year->forAll(self.title=Title::spring or self.title=Title::fall or self.title=Title::summer);

**invariant** SemesterSeason: **self**.*title* = *Title*::*Fall* **or** **self**.*title* = *Title*::*Spring* **or** **self**.*title* = *Title*::*Summer*;

**3.** **All core courses have to be taken in order to get the degree. Students can choose between elective courses.**context Student inv: self.enrolledcourses->forAll(includes(self.coreCourses));

**invariant** CoreCoursesTaken: **self**.*coreCourses*->*forAll*(course | **self**.*enrolledCourses*->*includes*(*course*));

**4. A BS student can register in maximum 5 courses during a semester whereas an MS student can register in max 3 courses (Exception will be given if a student has to take a pre-requisite course). NOT WRITTEN IN ECORE**

**invariant** MaxCoursesPerSem:   
**if** **self**.type = Type::BS **then** **self**.enrolledCourses>size() <= 5

**else** **if** **self**.type = Type::MS **then** **self**.enrolledCourses->size() <= 3

**else** **true** **endif**

**5. A student can withdraw a course before the announced date.**context Student inv: self.Info.withdrawn = ‘true’ implies self.Info.date < = self.course.announcedDate

**invariant** WithdrawBeforeAnnouncedDate: **self**.announcedDate > \_'ecore.xml.type'::Date.now()

**6. A withdrawn course appears as a ‘W’ on the transcript.**context Info inv: self.withdrawn = ‘true’ implies self.grade = ‘W’;

**invariant** WithdrawnCourseGrade: **self**.withdrawn **implies** **self**.grade = Grade::W;

**7. A student can drop a course within two weeks of registration.**context Student inv: self.Info.dropped = ‘true’ implies self.Info.date >= current.Date().subtractDays(14);

**invariant** droppedwithin14: **self**.enrolledCourses->forAll(course | **let** today : \_'ecore.xml.type'::Date = \_'ecore.xml.type'::Date.now() **in** course.announcedDate <> **null** **implies**(today - course.announcedDate).days() <= 14)

**8. A dropped course does not appear on the transcript.**Context Transcript inv: self.Student.enrolledCourses->exlcuse(c | c.Info.dropped = ‘true’)

**invariant** DroppedCourseNotOnTranscript: **self**.*data*->*forAll*(courseData | **not** *courseData*.*course*.*info*.*dropped*);

**9. CGPA is calculated by taking the aggregate of all GPAs of semesters taken so far.**Context student inv: self.cgpa = self.semesters->forAll(sgpa->sum()/self.semesters->count())

**invariant** CalculateCGPA: **self**.CGPA = **self**.semesters->collect(s | s.SGPA)->sum() / **self**.semesters->size();

**10. A student must repeat all failed courses.**Context Student inv: self->enrolledcourses->(includeforAll(self.failedcourses));

**invariant** RepeatFailedCourses: **self**.failedCourses->forAll(course | **self**.repeatCourses->includes(course))

**11. Students under warning must repeat such passed courses, in which their GPA is less than the minimum CGPA required for their degree. For example, MS students under warning (CGPA less than 2.50) must repeat courses in which their GPA is less than 2.50 (i.e. C or C- grade).**context Student inv: self.repeatcourses->include(c|exists(c.Info.gpa<self.Degree.minCGPA) && (c.Info.passed=’true’)

**invariant** RepeatCoursesUnderWarning: **self**.repeatCourses->includesAll(**self**.passedCourses->select(course | course.info.gpa < **self**.has->first().minCGPA **and** course.info.passed = **true**);

**12. A student can repeat a course if that course has not been passed or the student desires to improve his/her grade.**context Student inv: self.repeatcourses->include(c|c.Info.passsed=’false’ or (self.desiredRepeatcourses->include(c)))

**invariant** RepeatCourseCondition: **self**.*repeatCourses*->*forAll*(course | *course*.*info*.*passed* = **false** **or** **self**.*desiredRepeatCourses*->*includes*(*course*))

**13. Repeat courses will be indicated on the transcript together with the repeat count.**context course inv: self.Transcript.data->include(forAll(self.Student.repeatCourses)&&(self.repeatCount))

**invariant** IndicateRepeatCoursesOnTranscript: **self**.*Transcript*.*data*->*forAll*(courseData | **self**.*students*->*exists*(student | *student*.*repeatCourses*->*includes*(**self**) **and** **self**.*repeatCount* > 0))

**14. A student can view the status of their warnings if any.  
invariant** hasWarning: **Boolean** = **self**.*warningCount* > 0;

**15. At the end of every semester (including the first semester and summer semesters), warning is issued provided the CGPA of a student is less than the minimum required for the degree program.**context Student inv: (self.CGPA < self.Degree.minCGPA) implies self.Degree.warning = ‘true’;

**invariant** IssueWarningForLowCGPA: **self**.*degree*->*exists*(degree | **self**.*CGPA* < *degree*.*minCGPA* **implies** *degree*.*warning* = **true**);

**16. A student with a warning cannot register in a subsequent semester without approval of the relevant Head of Department or any other senior faculty nominated by him.**context Student inv: self.Degreee.warning=’true’ implies self.Approval.approvalReq=’true’

**invariant** WarningRequiresApproval: **self**.*degree*.*warning* = **true** **implies** **self**.*approval*.*approvalReq* = **true**

**17.** **After each semester (including the summer or the first semester), if the CGPA of a student is less than the required minimum, the Warning Count increases by one.**Context Semester inv: if(self.universiry.Student.CGPA < self.university.minCGPA) [self.university.warningCount@pre+1](mailto:self.university.warningCount@pre+1) else endif

**invariant** IncreaseWarningCount: **if** **self**.*university*.*students*->*exists*(student | *student*.*CGPA* < **self**.*university*.*minCGPA*) **then** **self**.*university*.*warningCount*@**pre** + 1 **else** **true** **endif**;

**18. If Warning Count becomes three, the admission of student at the University is closed. If the CGPA equals or exceeds the required minimum, the Warning Count becomes zero.**Context Student inv: if(self.Degree.warningCount >=3)   
 self.Degree.admissionCancel = ‘true’  
 elseif(self.CGPA >= self.Degree.minCGPA)  
 self.Degree.warningCount=0 endif

**invariant** ManageWarningAndAdmission: **if** **self**.degree.warningCount >= 3 **then** **self**.degree.admissionCancel = **true**

**elseif self**.CGPA >= **self**.degree.minCGPA **then** **self**.degree.warningCount = 0 **endif**;

**19. The admission will be cancelled automatically after completion of maximum duration allowed to earn a degree.**Context Degree inv: if (self.currenttime >= self.maxTime) self.admissionCancel = ‘true’ else endif

**invariant** CancelAdmissionAfterMaxDuration: **if** **self**.*currentTime* >= **self**.*maxTime* **then** **self**.*admissionCancel* = **true** **else** **true** **endif**;

**20. The students can view fee details.**

**invariant** viewFeeDetails: **Boolean** = **not** **self**.*has*->*isEmpty*() **and** **self**.*has*->*forAll*(fee | *fee*.*paid* = **true**);

**21. The fee is charged on Semester basis and it becomes due two weeks earlier than the start of a semester at the time of course registration.  
invariant** FeeDueDate: **self**.*feeDueDate* = **self**.*startDate*.*subtractDays*(14);

**22. Fees are paid in the bank through bank challan available in Accounts Office/Flex.**

**invariant** PaymentMethodIsBankChallan: **self**.*paymentMethod* = 'BankChallan';

**23. All fees are non-Refundable (except security deposit).  
invariant** NonRefundableFees: **self**.*refund* = **false** **or** **self**.*refund* = **true** **and** **self**.*name* = 'Security Deposit';

**24. A student cannot appear in any exam until all the dues are paid unless approved by the HOD.  
invariant** ExamEligibility: (**self**.FeesPaid = **true** **or** **self**.Approval.approvalAcquired = **true**) **implies** (**self**.Exam.examEligible = **true**)

**25. Indigent Students (decided by BASR) can apply for the loan.**Context Student inv: self.Loan.eligible = ‘true’ implies self.Finance.financialType = ‘indigent’

**invariant** LoanEligibilityAndFinancialStatus: (**self**.*canTakeLoan* = **true**) **implies** (**self**.*financialStatus* = *Finances*::*indigent*);

**26. This assistance is subject to renewal every semester in the light of the student’s academic performance and financial need.  
invariant** LoanRenewal: **self**.*renewedEverySemester* **and** **self**.*renewalCriteriaMet*;

**27. The financial assistance is limited to tuition fees only  
invariant** LoanAssistanceType: **self**.*assistanceType* = 'Tuition Fees';

**28.** **The loan is discontinued if the student’s CGPA falls below 2.00 for undergraduate degree and 2.5 for graduate degree.**Context Student inv: if (self.type=’BS’ and self.cgpa < 2.00)  
 self.Loan.eligible = ‘false’  
 elseif(self.type=’MS’ and self.cgpa < 2.50)  
 self.Loan.eligible = ‘false’  
 endif

**invariant** LoanEligibilityBasedOnCGPA:   
**if** **self**.*type* = *Type*::*BS* **then** **self**.*CGPA* >= 2.00 **or** **not** **self**.*Loan*.*eligible*

**elseif** **self**.*type* = *Type*::*MS* **then** **self**.*CGPA* >= 2.50 **or** **not** **self**.*Loan*.*eligible*

**else** **true** **endif**;

**29. The repayment of loan starts three months after graduation or getting a job, whichever is earlier.**Context Student inv: self.Loan.repaymentStartDate = self.Graduate.date.earlierOf(self.Job.startDate)

**invariant** LoanRepaymentStart: **let** earliestDate : Date = **self**.graduates.date.earlierOf(**self**.getsJob.startDate) **in** **self**.loan.repaymentStartDate = earliestDate.addMonths(3)

**30. The total amount has to be repaid within a period of four years after graduation. Students are required to sign to this effect.  
invariant** LoanPeriod: **self**.*repaymentEndDate* = **self**.*repaymentStartDate*.*addYears*(4);

**31. All those who are admitted and are in real need of financial assistance should apply on the prescribed Financial Aid form. A notice to this effect will be posted on the notice board.**Context Student inv: self.realNeedofAssistance = ‘true’ imples self.Aid.applicable = ‘true’

**invariant** AidApplicability: **self**.*realNeedofAssistance* **implies** **self**.*Aid*.*applicable*;

**32. Merit scholarship is awarded to the Top Three position holders of each Examination Board.**Context Student inv: (self.position <=3 and self.Exame.type = ‘board’) implies self.Scholarship.awarded = ‘true’

**invariant** ScholarshipAward: (**self**.*position* <= 3 **and** **self**.*Exam*.*type* = *Paper*::*board*) **implies** **self**.*isAwarded*;

**33. Scholarship is also offered to top three position holders in the NU admission merit list of each campus.**Context Student inv: University.Campus -> forAll( if (self.position <= 3 and self.Exam.type = ‘NU test’) self.Scholarship.offered = ‘true’ else endif

**invariant** NUAdmissionMeritScholarship: **self**.*university*.*Campus*->*forAll*(campus | **if** (**self**.*position* <= 3 **and** **self**.*Exam*.*type* = *Paper*::*\_'NU test'*)   
**then** **self**.*isOfferedScholarship* **else** **true** **endif**)

**34.** **Students awarded merit Scholarship, 100% tuition fee for 8 regular semesters must maintain a CGPA of ≥ 3.00 in each semester.**Context Student inv: if (self.Scholarship.offered = ‘true’)   
 if(self.Scholarship.numOfSems = 8 and self.Scholarship.coverage = ‘100%’)

self.cgpa >= 3.00 else endif  
**invariant** MeritScholarshipCGPARequirement: **if** **self**.*Scholarship*.*offered* **then** **if** **self**.*Scholarship*.*numOfSems* = 8 **and** **self**.*Scholarship*.*coverage* = 100.0 **then** **self**.*cgpa* >= 3.00 **else** **true** **endif** **else** **true** **endif**;

**35. A student can view the attendance of the any course in which he/she has registered.**

**invariant** ViewAttendance: **self**.*enrolledCourses*->*forAll*(course | *course*.*Attendance*);

**36. In order for student to appear in the Final exam of a course he/she should have attendance more than 80%.**

**invariant** FinalExamEligibility: **self**.*enrolledCourses*->*forAll*(course |   
**if** *course*.*AttendancePercentage* >= 80 **then** *course*.*FinalExamScheduled* = **true**

**else** *course*.*FinalExamScheduled* = **false** **endif**);

**37. A student can view the marks of each course (detailed view).**

**invariant** ViewMarks: **self**.*enrolledCourses*->*forAll*(course | *course*.*Marks*);

**38. A student can apply for Change in Grade Request after the results are announced.**

**invariant** ChangeInGradeRequest: **self**.*enrolledCourses*->*forAll*(course | *course*.*ResultsAnnounced* **implies** *course*.*CanApplyForChangeInGrade*);

**39. A student’s request to retake any exam is approved by the respective HOD.**

**invariant** ExamRetakeApproval: **self**.*enrolledCourses*->*forAll*(course | **if** *course*.*RequestedExamRetake* **then** *course*.*Department*.*HOD*.*ApprovalRequired* **else** **true** **endif**)