```
====University Exam Grading===
1.Add Course
2.Add Student
3.Create Assessment
4.Schedule Exam
5.Record Marks
6.Publish Grades
7.Generate Transcript
8.Exit
Enter your choice: 1
Enter Course ID: 107
Enter Course Name: C programming
Course added.
====University Exam Grading===
1.Add Course
2.Add Student
3.Create Assessment
4.Schedule Exam
5.Record Marks
6.Publish Grades
7.Generate Transcript
8.Exit
Enter your choice: 2
Enter Student ID: 108
Enter Student Name: anu
Student added.

====University Exam Grading===
1.Add Course
2.Add Student
3.Create Assessment
4.Schedule Exam
5.Record Marks
6.Publish Grades
7.Generate Transcript
8.Exit
Enter your choice: 3
Enter student ID: 108
Enter course code: 107
anuregistered for thisC programming
Enter assessment name: midterm
Enter weightage: 20.0
Assessment 'midterm' added for course C programming
Assessment added.
```

Assessments

-name:String
-weightage:double
-marks:double

+getName()
+setName()
+getWeightage()
+setMarks()

Grade

-getGrade()
-getGradePoints()
-Resit()
-Improve()

1

Course

-courseID:String
-courseName:String
-courseAssessments:List

+addAssessment()
+calculateTotalMarks()
+getCourseGrade()
+toString()

1

ExamSchedule

-course:Course
-examDate:String
-examLocation:String

+getCourse()
+setCourse()
+getExamDate()
+setExamDate()
+getExamLocation()
+setExamLocation()

Menu

-sc:Scanner
-List:Courses
-List:Student
-schedules:List

+main(String[])

Student

-studentId:int
-studentName:String
-registeredCourse:List
-courseAssessment:Cour
se
-courseMarks:Course

+register(Course())
+evaluateStudent()
+recordMarks()
+addAssessment()
+printTranscript()

1

Transcript

-student:Student
-courseGrades:String

+addCourseGrade()
+calculateGPA()
+applyResit()
+applyImprovement()
+printTranscript()

MarkEntry

-student:Student
-course:Course
-marks:double

+getStudent()
+setStudent()
+getCourse()
+setCourse()
+getMarks()
+setMarks()
+toString()