Courses UML Class Diagram

<<Abstract>> Session
+ Time { get; } : TimeCell
+ Room { get; } : string
+ ITeacher { get; } : Teacher

+ Session(TimeCell time, string room, ITeacher teacher)
+ EnsureRole(ITeacher teacher) : void

<<Abstract>> Coursework
+ Title { get; } : string
+ TypeOfDefenseSubmission { get; } : ISubmissionCoursework
+ Supervisor { get; } : ISuperviseCapable

+ Coursework(string title, ISubmissionCoursework typeOfDefenseSubmission, ISuperviseCapable supervisor)
+ Submit(List<string> load) : void

StudentGroup
+ Name { get; } : string
+ Schedule { get; set; } : List<Session>
+ Courses { get; set; } : List<CourseEnrollment>
+ CourseWorks { get; } : List<CourseWork>

+ StudentGroup(string name) : void
+ AddSession(Session session) : void
+ RemoveSession(Session session) : void
+ Enroll(ICourseFactory factory, Lecturer lecturer, Assistant assistant, ISuperviseCapable supervisor, TimeCell lectureTime, string lectureRoom, TimeCell practicalTime, string practicalRoom) : CourseEnrollment
+ CheckConflicts(Session session) : bool

CourseEnrollment
+ CourseName { get; } : string
+ Lecture { get; } : Session
+ Practical { get; } : Session
+ Coursework { get; } : Coursework
+ LectureRoom { get; } : string
+ PracticalRoom { get; } : string

+ CourseEnrollment(string courseName, Session lecture, Session practical, Coursework coursework, string lectureRoom, string practicalRoom)

ProgrammingCoursework
+ ProgrammingCoursework(string title, ISubmissionCoursework typeOfDefenseSubmission, ISuperviseCapable supervisor) : base(title, typeOfDefenseSubmission, supervisor)
+ Submit(List<string> load) : void

DatabasesCoursework
+ DatabasesCoursework(string title, ISubmissionCoursework typeOfDefenseSubmission, ISuperviseCapable supervisor) : base(title, typeOfDefenseSubmission, supervisor)
+ Submit(List<string> load) : void

Practice
+ Practice(TimeCell time, string room, ITeacher teacher) : base(time, room, teacher) {
+ EnsureRole(ITeacher teacher) : void
}

Lecture
+ Lecture(TimeCell time, string room, ITeacher teacher) : base(time, room, teacher) {
+ EnsureRole(ITeacher teacher) : void
}

DatabasesLecture
+ DatabasesLecture(TimeCell time, string room, ITeacher teacher) : base(time, room, teacher)

DatabasesPractice
+ DatabasesPractice(TimeCell time, string room, ITeacher teacher) : base(time, room, teacher)

ProgrammingLecture
+ ProgrammingLecture(TimeCell time, string room, ITeacher teacher) : base(time, room, teacher)

ProgrammingPractice
+ ProgrammingPractice(TimeCell time, string room, ITeacher teacher) : base(time, room, teacher)

TimeCell
+ Day { get; } : DayOfWeek
+ Start { get; } TimeOnly
+ End { get; } TimeOnly

+ TimeCell(DayOfWeek Day, TimeOnly Start, TimeOnly End)
+ CheckOverlay(TimeCell other) : bool
+ Parse(DayOfWeek day, string start, string end) : TimeCell

GitHubRepoDefense
+ Submit(string load) : void

<<Interface>> ISubmissionCoursework
+ Submit(string load) : void

OnlineUploadDefense
+ Submit(string load) : void

OfflineDefense
+ Submit(string load) : void

<<Interface>> IPracticalCapable

<<Interface>> ILectureCapable

<<Interface>> ISuperviseCapable

<<Interface>> ICourseFactory
+ CourseName { get; } : string

+ CreateLecture(TimeCell time, string room, Lecturer lecturer) : Session
+ CreatePractical(TimeCell time, string room, Assistant assistant) : Session
+ CreateCoursework(ISuperviseCapable supervisor) : Coursework

<<Abstract>> SessionFactory
+ CreateSession(TimeCell time, string room, ITeacher teacher) : Session
+ Make(TimeCell time, string room, ITeacher teacher) : Session

<<Interface>> ITeacher
+ Name { get; } : string

Lecturer
+ Lecturer(string name)

Assistant
+ Assistant(string name)

ExternalMentor
+ ExternalMentor(string name)

ProgrammingCourseFactory
+ CourseName { get; } : string

+ CreateLecture(TimeCell time, string room, Lecturer lecturer) : Session
+ CreatePractical(TimeCell time, string room, Assistant assistant) : Session
+ CreateCoursework(ISuperviseCapable supervisor) : Coursework

DatabasesCourseFactory
+ CourseName { get; } : string

+ CreateLecture(TimeCell time, string room, Lecturer lecturer) : Session
+ CreatePractical(TimeCell time, string room, Assistant assistant) : Session
+ CreateCoursework(ISuperviseCapable supervisor) : Coursework

LectureFactory
+ Make(TimeCell time, string room, ITeacher teacher) : Session

PracticeFactory
+ Make(TimeCell time, string room, ITeacher teacher) : Session