UniTime is an academic scheduling software that is primarily used by universities and colleges to create and manage class schedules. The software enables administrators and faculty to efficiently schedule courses, manage room assignments, and optimize class times to maximize student enrollment and minimize scheduling conflicts. Also provides a range of features and tools to support academic planning and management, including reporting and analytics tools, student enrollment management, and integration with external systems.

Using Bottom-up code comprehension in understanding UniTime since we are unfamiliar with the system. Starting with the bottom components we can deduce the following in class *InstructorAddAction*. Function *excute()* that includes many conditions that checks if that department is not null then it returns back the department of that intended session

*if (sessionContext.getAttribute(SessionAttribute.DepartmentId) != null)*

another example the condition *if (MSG.actionSaveInstructor().equals(op))* statement is checking whether the value returned by MSG.actionSaveInstructor() is equal to the value stored in the op. it will proceed to one of two cases update instructor or reporting an error one of the two cases is done according to the nested condition and its else.

Other class *MainAction.java* its basic building blocks are some setters and getters function that used with *messege, systemMessage.* Function *printInitializationError()* used to output the error message if that error is not null. Its *excute()* function assign message with it’s value in both cases if message has a value of null.

Bottom-up code comprehension is an approach to understanding software code that starts with the most basic building blocks of the code and works up to higher-level functionality.