**UNIVERSIDAD ICESI**

*Faculty of Engineering*

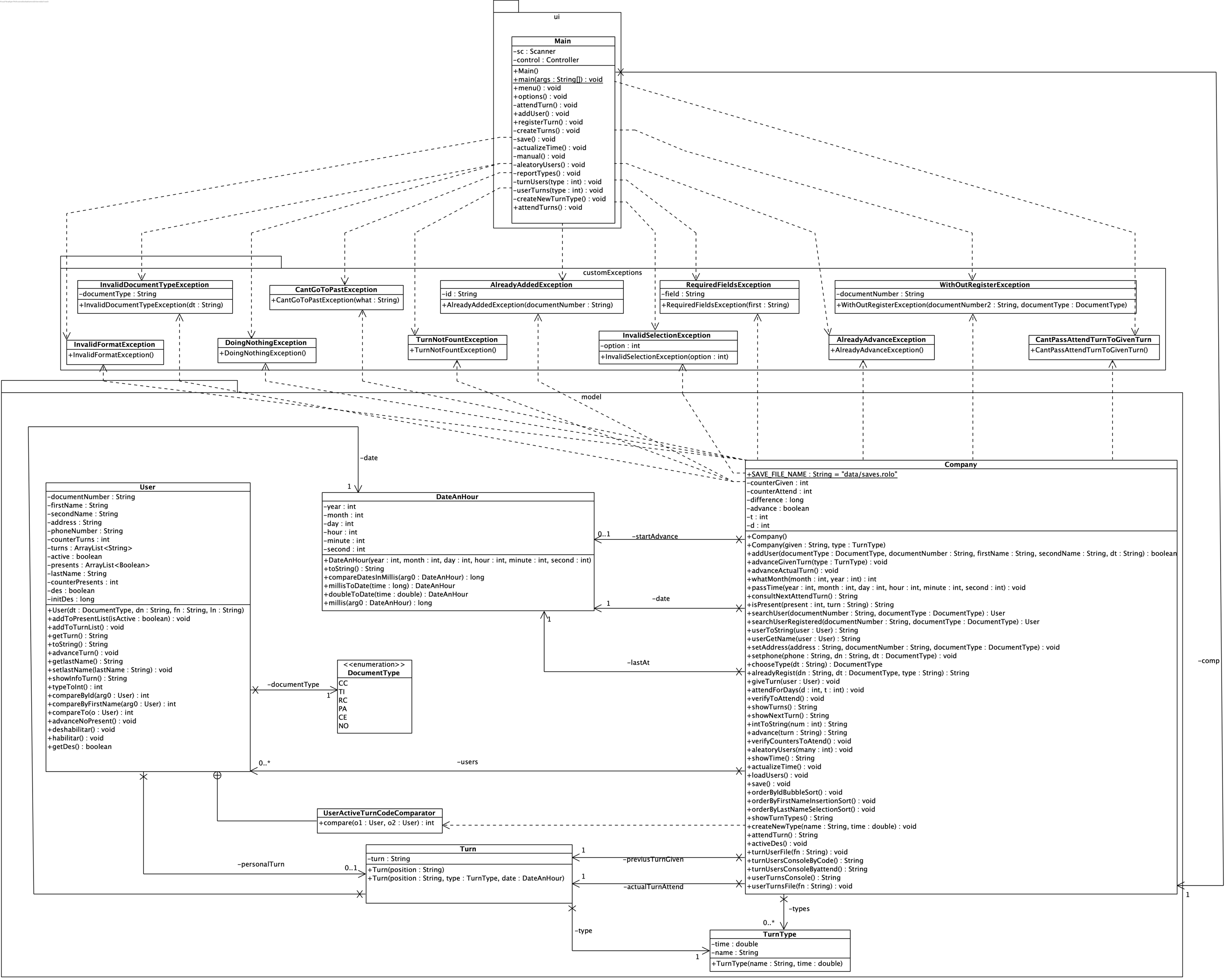
*Algorithms and Programming II Term 2020-1*

**Laboratoy #2 Turns Management**

Sebastián Barrera1.

*1Software Engineering, Faculty of Engineering, Icesi University*

Class Diagram:



Funcional Requirements:

The system must be able to:

**Req1. Add** a new user with document type, document number, first name, last name, phone, and address. Users with the same number and document type will not be allowed. Neither users without document type, document number, first names or last names.

**Req2. Assign** a turn to an already created user, asking for their type and document number. The turn will consist of a letter and a number between 00 and 99, the first turn is A00, the next A01 and so on and so forth. When you get to the A99 turn you will switch to B00 and so on. The moment a user is assigned a turn, he will be marked as active, indicating that he will not be able to be assigned more turns until he is taken care of. In case the user asks, the turn he already has will be shown.

**Req3. Allow** you to assign endless turns regardless of whether the previous ones were already served or not.

**Req4. Attend** turns in ascending order until you reach the last given turn. For each turn you will have the possibility to indicate whether the user was served, or not because it was not there. This information will be saved and displayed when the user re-orders a new turn.

**Req5. Manage** turn types with preset durations in minutes. It'll always be given in minutes. since the turn is created, you will have your kind of shift. The program will be attended by one person, also The turns will be attended in the turn type time.

**Req6. Show** the current system date and time, indicating: year, month, day, hour, minute, and second..

**Req7. Update** the system date and time, it have to be with the local time of the computer, and with values indicates by the user, it only can be advanced. The time advanced have to be always advance.

**Req8. Generate** reports with the information of all the turns that a user have been take, and all the information of all the users that take the turn X##.

**Req9. Suspending** a person has not been present in the last two shifts, so that they cannot request new shifts for two days.

**Req10. Generate** randomly users reading different files and turns to the users register previously in the system. The user can say hoy many days and turns per day generate.

**Req11.** Show always the time that take the operation to be do its

No functionals Requirements

The system must be able to:

**Req1. Use** “serializable” to save.

**Req2. Sort** by bubble, insertion and selection, also use collections, reverse order, comparator and comparable.

**Req3. Use** a binary search and a sequential search.

**Req4. Read** files and write over files.

Location of some algoritms in the code.

Comparable was implemented in User Class.

Comparator was implemented in UserActiveTurnCodeComparator Class.

The binary search was implemented searchUserRegistered() in Company Class.

The sequential search was implemented in searchUser() in Company Class.

Bubble sort was used in orderByIdBubbleSort() in Company Class.

Insertion sort was implemented in orderByFirstNameInsertionSort in Company Class.

Selection sort was implemented in orderByLastNameSelectionSort() in Company Class.

The anonymous class was implemented in turnUsersConsoleByCode() in Company Class