CS305 – Programming Languages

Spring 2024 – 2025

HOMEWORK 3

Implementing a Semantic Analyzer for MS

Due date: April ~~24~~ 29, 2025 @ 23:55

NOTE

Only SUCourse submission is allowed. No submission by e-mail. Please see the note at the end of this document for late submission policy.

1 Introduction

In this homework, you will implement a tool that includes a simple semantic analyzer and a translator for MS language. Detailed information about the context-free grammar of this programming language was given in the second homework document. You can check it for more information.

The tool that you will implement in this homework will first check if a given MS program has any syntax errors grammatically. The tool will also perform some semantic checks if there are no syntax errors. If these checks are passed, then your tool will print out some results. Read the rest of the document for more information.

2 Parser and Scanner

The scanner and the parser, which you can use to implement this homework, are provided to you. The semantic analysis will require you to implement an attribute grammar. You can start from the scanner/parser files provided, or of course, you can write your own versions of scanner and parser from scratch.

3 Semantic Rules

Your semantic analyzer should start by performing an analysis for the following semantic rules. Your semantic analyzer must print out an error message for each violation of these rules. Note that after printing out an error message, your semantic analyzer must not terminate and keep working to find further violations, if any exists.

1. For the time objects, only the following range is valid: {00.00 – 23.59}. That is, the hour part can be any two-digit number between {00 – 23}, and the minute part can be any two-digit number between {00 – 59}. If a time object does not obey this rule, an error message in the following format must be printed:

   `LINE_INVALID_TIME_(TIME)`

   Here, `LINE` must be the line on which the invalid time object appears. For example, if `25.15` is seen at line 7, then the following error message will be printed:

   `7_INVALID_TIME_(25.15)`

2. For the date objects, **we are using a date system on planet CS305, where there are again 12 months in a year, but each month has exactly 28 days.** In this homework, the valid date ranges are as follows: {01.01.2025 – 28.12.2050}. That is, the day part can be any two-digit number between {01 – 28}, the month part can be any two-digit number between {01 – 12}, and the year part can be any four-digit number between {2025 – 2050}. If a date object does not obey this rule, an error message in the following format must be printed:

   `LINE_INVALID_DATE_(DATE)`

   Here, `LINE` must be the line on which the invalid date object appears. For example, if `34.44.2923` is seen at line 11, then the following error message will be printed:

   `11_INVALID_DATE_(34.44.2923)`

3. If both the date objects and the time objects (for the start and the end of the meeting) of a meeting are valid, then the program must control the following condition. The end time of a meeting must be after the start time of the meeting. The start time and the end time cannot be the same. Note that, you need to take the day part of the meeting start and end into account as well. Therefore a meeting that starts on 24.03.2025 at 23.00, and ends on 25.03.2025 at 01.00 is valid. If there is a meeting block whose end time is not after its start time, an error message in the following format must be printed:

   `LINE_ENDTIME_ERROR_(MEETINGNUMBER)`

   Here, `LINE` must be the line on which the `Meeting` keyword of the meeting for which we have this problem, and `MEETINGNUMBER` is the meeting number set as the value of the `meetingNumber` element of the meeting block. For example, if there is such a meeting block:

```
Meeting "Erroneous Meeting"
   meetingNumber = 673
   description = "This meeting has an end time error."
   startDate = 28.07.2025
   startTime = 14.40
   endDate = 15.05.2025
   endTime = 16.30
   locations = FENSG029
   isRecurring = yes
   frequency = monthly
   repetitionCount = 12
endMeeting
```

then the following error message will be printed:

`1_ENDTIME_ERROR_(673)`

4. A sub–meeting must not exceed the time limits of its parent meeting. In other words, a sub–meeting cannot start before the start time of its parent and cannot end after the end time of its parent. This check will only be performed for a sub–meeting and its parent meeting, which have valid date/time setting, that is:
   - they both have valid time and date objects for their start and end times
   - the sub–meeting starts before it ends, and
   - the parent meeting starts before it ends.

   If these conditions are not satisfied (that is, there are already some errors for the date/time setting of the parent and/or the sub–meeting), this check will not performed.

   If there is an error with respect to this rule, then an error message in the following format must be printed:

   `LINE_RANGE_ERROR_(SUBMEETINGNUMBER_PARENTMEETINGNUMBER)`

   Here, the `LINE` must be the line on which the `Meeting` keyword of the sub–meeting for which we have this problem, the `SUBMEETINGNUMBER` must be the `meetingNumber` of the sub–meeting, and the `PARENTMEETINGNUMBER` must be the `meetingNumber` of the parent meeting. For example, if there is such a meeting block:

```
Meeting "A meeting that starts at a normal time"
    meetingNumber = 888
    description = "An important one"
    startDate = 24.03.2025
    startTime = 19.40
    endDate = 25.03.2025
    endTime = 21.30
    locations = FENSG035
    isRecurring = yes
```

```
        frequency = monthly
        repetitionCount = 12
        subMeetings
            Meeting "A sub-meeting that starts lately"
                meetingNumber = 333
                description = "Another important one"
                startDate = 24.03.2025
                startTime = 20.40
                endDate = 25.03.2025
                endTime = 20.30
                locations = FENSG035
                isRecurring = no
                subMeetings
                    Meeting "A sub-meeting that starts earlier than its parent"
                        meetingNumber = 243
                        description = "Another important one"
                        startDate = 24.03.2025
                        startTime = 14.40
                        endDate = 25.03.2025
                        endTime = 21.30
                        locations = FENSG035
                        isRecurring = no
                    endMeeting
                endSubMeetings
            endMeeting
        endSubMeetings
    endMeeting
```

then the following error message will be printed:

```
23_RANGE_ERROR_(243_333)
```

5. For the locations element of a meeting (or a sub–meeting), a comma–separated locations (rooms) are provided. The same room cannot be given multiple times within the locations of a meeting. Note that the location names are case–sensitive; hence, FENSG032 and FensG032 are considered to be different locations. If there exists such a redundant locations issue, then an error message in the following format must be printed:

   ```
   LINE_REPEATED_ROOM_ERROR_(LOCATIONS)
   ```

   Here, `LINE` must be the line on which the `locations` keyword of the meeting for which we have this problem. `LOCATIONS` is the comma–separated list of the identifiers given for the locations of this meeting.

   For example, for the following example:

```
    Meeting "Programming Languages"
```

```
    meetingNumber = 123
    description = "Basically the best course"
    startDate = 07.01.2025
    startTime = 08.40
    endDate = 23.05.2025
    endTime = 10.30
    locations = FENSG077, FENSG035, FENSG035, FENSG032, FENSG077
    isRecurring = no
endMeeting
```

the following single error message will be printed:

```
8_REPEATED_ROOM_ERROR_(FENSG077, FENSG035, FENSG035, FENSG032, FENSG077)
```

6. A sub–meeting cannot use a location not given in the locations of its parent meeting. Hence, any room name given in the location list of a sub–meeting must exist also in the location list of its parent meeting. If at least one of the locations of a sub–meeting is not in the location list of the parent meeting, then an error message in the following format must be printed:

   ```
   LINE_LOCATION_ERROR_(SUBMEETINGNUMBER_PARENTMEETINGNUMBER)
   ```

   Here, the `LINE` must be the line on which the `locations` keyword of the sub–meeting for which we have this problem, the `SUBMEETINGNUMBER` must be the `meetingNumber` of the sub–meeting, and the `PARENTMEETINGNUMBER` must be the `meetingNumber` of the parent meeting.

   Note that, if there are multiple locations in the sub–meeting which do not appear in the locations of the parent meeting, only one error will be produced.

   For example, if there is such a meeting block:

```
Meeting "Programming Languages"
    meetingNumber = 123
    description = "Basically the best course"
    startDate = 07.01.2025
    startTime = 08.40
    endDate = 23.05.2025
    endTime = 10.30
    locations = FENSG077, FENSG035
    isRecurring = yes
    frequency = weekly
    repetitionCount = 14
    subMeetings
        Meeting "Monday Class"
            meetingNumber = 124
            description = "Basically the best course"
            startDate = 10.02.2025
            startTime = 13.40
```

```
                endDate = 10.02.2025
                endTime = 14.30
                locations = UCG030, FENSG032, FENSG077
                isRecurring = yes
                frequency = weekly
                repetitionCount = 14
            endMeeting
        endSubMeetings
    endMeeting
```

then the following error message will be printed:

```
20_LOCATION_ERROR_(124_123)
```

7. A meeting given as a sub–meeeting has the same basic grammar rules as a top level meeting. However, a sub–meeting cannot repeat. Therefore, `isRecurring` element of a sub–meeting must always be set to no. If any sub–meeting has its isRecurring element set to yes, an error message in the following format must be printed:

   ```
   LINE_REPEATING_SUBMEETING_(SUBMEETINGNUMBER)
   ```

   Here, the `LINE` must be the line on which the `isRecurring` keyword of the sub–meeting for which we have this problem, `SUBMEETINGNUMBER` is the meeting number of the sub–meeting. For example, if there is such a meeting block:

```
Meeting "Weekly Meeting"
    meetingNumber = 1
    description = "Repeat weekly"
    startDate = 11.08.2029
    startTime = 10.40
    endDate = 11.08.2029
    endTime = 11.30
    locations = zoom
    isRecurring = yes
    frequency = weekly
    repetitionCount = 52
    subMeetings
        Meeting "Daily Meeting
            meetingNumber = 2
            description = "Repeat daily"
            startDate = 11.08.2029
            startTime = 10.40
            endDate = 11.08.2029
            endTime = 11.30
            locations = zoom
            isRecurring = yes
```

```
            frequency = daily
            repetitionCount = 7
        endMeeting
    endSubMeetings
endMeeting
```

then the following error message will be printed:

`21_REPEATING_SUBMEETING_(2)`

8. If in a meeting block the `isRecurring` element assigned to `no`, then `frequency` element must not exist. If there is such an unexpected `frequency` element, then an error message in the following format must be printed:

   `LINE_UNEXPECTED_FREQUENCY_(MEETINGNUMBER)`

   Here, `LINE` must be the line on which the `frequency` keyword of the meeting for which we have this problem, `MEETINGNUMBER` is the meeting number of the erroneous meeting. For example, if there is such a meeting block:

```
Meeting "Weekly Meeting"
    meetingNumber = 1
    description = "Repeat weekly"
    startDate = 11.08.2029
    startTime = 10.40
    endDate = 11.08.2029
    endTime = 11.30
    locations = zoom
    isRecurring = yes
    frequency = weekly
    repetitionCount = 52
    subMeetings
        Meeting "Daily Meeting"
            meetingNumber = 2
            description = "Repeat daily"
            startDate = 11.08.2029
            startTime = 10.40
            endDate = 11.08.2029
            endTime = 11.30
            locations = zoom
            isRecurring = no
            frequency = daily
        endMeeting
    endSubMeetings
endMeeting
```

then the following error message will be printed:

`22_UNEXPECTED_FREQUENCY_(2)`

9. If in a meeting block the `isRecurring` element assigned to `no`, then `repetitionCount` element must not exist. If there is such an unexpected `repetitionCount` element, then an error message in the following format must be printed:

   `LINE_UNEXPECTED_REPETITIONCOUNT_(MEETINGNUMBER)`

   Here, `LINE` must be the line on which the `repetitionCount` keyword of the meeting for which we have this problem, `MEETINGNUMBER` is the meeting number of the erroneous meeting. For example, if there is such a meeting block:

```
1  Meeting "Weekly Meeting"
2      meetingNumber = 1
3      description = "Repeat weekly"
4      startDate = 11.08.2029
5      startTime = 10.40
6      endDate = 11.08.2029
7      endTime = 11.30
8      locations = zoom
9      isRecurring = yes
10     frequency = weekly
11     repetitionCount = 52
12     subMeetings
13         Meeting "Daily Meeting"
14             meetingNumber = 2
15             description = "Repeat daily"
16             startDate = 11.08.2029
17             startTime = 10.40
18             endDate = 11.08.2029
19             endTime = 11.30
20             locations = zoom
21             isRecurring = no
22             repetitionCount = 7
23         endMeeting
24     endSubMeetings
25 endMeeting
```

   then the following error message will be printed:

   `22_UNEXPECTED_REPETITIONCOUNT_(2)`

10. If in a **top–level** meeting block (Note that, this rule does not apply to sub–meetings) the `isRecurring` element assigned to `yes`, then `repetitionCount` and `frequency` elements must exist. If there is an absence of either the `frequency` or `repetitionCount` element, then an error message in the following format must be printed:

    `LINE_MISSING_ELEMENT_(MEETINGNUMBER)`

Note that, if both `frequency` and `repetitionCount` elements are missing, there will be only one error printed.

Here, `LINE` must be the line on which the `isRecurring` keyword of the sub–meeting for which we have this problem, `MEETINGNUMBER` is the meeting number of the erroneous sub–meeting. For example, if there are such meeting blocks:

```
1  Meeting "Daily Meeting 1"
2      meetingNumber = 256
3      description = "Repeat daily"
4      startDate = 11.08.2029
5      startTime = 10.40
6      endDate = 11.08.2029
7      endTime = 11.30
8      locations = zoom
9      isRecurring = yes
10     frequency = daily
11 endMeeting
12 Meeting "Daily Meeting 2"
13     meetingNumber = 349
14     description = "Repeat daily"
15     startDate = 11.08.2029
16     startTime = 11.40
17     endDate = 11.08.2029
18     endTime = 12.30
19     locations = zoom
20     isRecurring = yes
21     repetitionCount = 7
22 endMeeting
23 Meeting "Daily Meeting 3"
24     meetingNumber = 17
25     description = "Repeat daily"
26     startDate = 11.08.2029
27     startTime = 10.40
28     endDate = 11.08.2029
29     endTime = 12.30
30     locations = zoom
31     isRecurring = yes
32 endMeeting
```

then the following error message will be printed:

```
9_MISSING_ELEMENT_(256)
20_MISSING_ELEMENT_(349)
31_MISSING_ELEMENT_(17)
```

11. Every meeting block must have a unique `meetingNumber` value. If there are meeting blocks that share the same `meetingNumber` value, an error message in the following format must be printed:

    `LINE_REPEATED_MEETINGNUMBER_(MEETINGNUMBER)`

    Here, `LINE` must be the line on which the `meetingNumber` keyword of the meeting for which we have this problem, `MEETINGNUMBER` is the meeting number of the erroneous meeting.

    Note that, if there are multiple meetings with the same meeting number, the first such meeting is not an error. Only the occurrences after the first meeting will be reported. For example, for the following meeting blocks:

```
1   Meeting "Daily Meeting 1"
2       meetingNumber = 256
3       description = "Repeat daily"
4       startDate = 11.08.2029
5       startTime = 10.40
6       endDate = 11.08.2029
7       endTime = 11.30
8       locations = zoom
9       isRecurring = yes
10      frequency = daily
11      repetitionCount = 7
12  endMeeting
13  Meeting "Daily Meeting 2"
14      meetingNumber = 256
15      description = "Repeat daily"
16      startDate = 11.08.2029
17      startTime = 11.40
18      endDate = 11.08.2029
19      endTime = 12.30
20      locations = zoom
21      isRecurring = yes
22      frequency = daily
23      repetitionCount = 7
24  endMeeting
25  Meeting "Daily Meeting 3"
26      meetingNumber = 256
27      description = "Repeat daily"
28      startDate = 11.08.2029
29      startTime = 10.40
30      endDate = 11.08.2029
31      endTime = 12.30
32      locations = zoom
33      isRecurring = yes
```

```
34          frequency = daily
35          repetitionCount = 7
36      endMeeting
```

the following error messages will be printed:

```
14_REPEATED_MEETINGNUMBER_(256)
26_REPEATED_MEETINGNUMBER_(256)
```

4 Understanding Recurring Meetings

In this section, we will explain how the recurring meetings need to be understood. In a recurring meeting (when `isRecurring` is set to yes, which is only possible for top–level meeetins), we have both the `frequency` and the `repetitionCount` information provided (as guaranteed by Rule 10).

Let us assume that there is a top–level recurring meeting M with `repetitionCount` set to N. This is equivalent to having N copies of M (let us call these copies as $M_0, M_1, \ldots, M_{N-1}$). Each copy M_i has exactly the same information as M (including the submeetings if it has any), with the following exceptions. For M_i, `isRecurring` element is no, and there are no `repetitionCount` and `frequency` elements. Furthermore, the date elements of $M_0, M_1, \ldots, M_{N-1}$ (and if exists, sub–meetings at any level) are set as follows:

> If the meeting is recurring daily (respectively, weekly–monthly–yearly), the `startDate` and the `endDate` of the copy M_i is i days (respectively, weeks–months–years) later than that of M.

Note that, if M has sub–meetings (sub–sub–meetings, sub–sub–sub–meetings, etc.), then each copy M_i will also have the same sub–meetings (sub–sub–meetings, sub–sub–sub–meetings, etc.), where the date elements of all these sub–meetings (sub–sub–meetings, sub–sub–sub–meetings, etc.) are modified in the same way, as explained above.

Please recall that, in this homework we are using the date system on planet CS305, where there are again 12 months in a year, but each month has exactly 28 days. For example, 6 days after 26.01.2025 is 04.02.2025 (not 01.02.2025).

To give an example of copies of a recurring meeting, consider the following meeting block:

```
1  Meeting "A recurring meeting"
2      meetingNumber   = 123
3      description     = "Top-level recurring meeting"
```

```
    startDate       = 24.03.2025
    startTime       = 19.00
    endDate         = 24.03.2025
    endTime         = 21.00
    locations       = FENSG032, FENSG035
    isRecurring     = yes
    frequency       = weekly
    repetitionCount = 3
    subMeetings
        Meeting "A sub-meeting of a recurring meeting"
            meetingNumber = 124
            description   = "It also has sub-meetings"
            startDate     = 24.03.2025
            startTime     = 19.00
            endDate       = 24.03.2025
            endTime       = 20.00
            locations     = FENSG032, FENSG035
            isRecurring   = no
            subMeetings
                Meeting "A sub-sub-meeting"
                    meetingNumber = 125
                    description   = "A leaf level meeting, i.e. no submeetings"
                    startDate     = 24.03.2025
                    startTime     = 19.00
                    endDate       = 24.03.2025
                    endTime       = 20.00
                    locations     = FENSG032
                    isRecurring   = no
                endMeeting
                Meeting "A sub-meeting that starts earlier than its parent"
                    meetingNumber = 126
                    description   = "Another leaf level meeting"
                    startDate     = 24.03.2025
                    startTime     = 19.00
                    endDate       = 24.03.2025
                    endTime       = 20.00
                    locations     = FENSG035
                    isRecurring   = no
                endMeeting
            endSubMeetings
        endMeeting
        Meeting "Another submeeting of top-level meeting."
            meetingNumber = 127
            description   = "This one is also a leaf-level meeting"
            startDate     = 24.03.2025
```

```
            startTime     = 20.00
            endDate       = 24.03.2025
            endTime       = 21.00
            locations     = FENSG035
            isRecurring   = no
        endMeeting
    endSubMeetings
endMeeting
```

The meeting given above is equivalent to the following 3 meetings (note how dates are modified and how the sub–meetings are also copied):

```
Meeting "A recurring meeting"
    meetingNumber   = 123
    description     = "Top-level recurring meeting"
    startDate       = 24.03.2025
    startTime       = 19.00
    endDate         = 24.03.2025
    endTime         = 21.00
    locations       = FENSG032, FENSG035
    isRecurring     = no
    subMeetings
        Meeting "A sub-meeting of a recurring meeting"
            meetingNumber = 124
            description   = "It also has sub-meetings"
            startDate     = 24.03.2025
            startTime     = 19.00
            endDate       = 24.03.2025
            endTime       = 20.00
            locations     = FENSG032, FENSG035
            isRecurring   = no
            subMeetings
                Meeting "A sub-sub-meeting"
                    meetingNumber = 125
                    description   = "A leaf level meeting, i.e. no submeetings"
                    startDate     = 24.03.2025
                    startTime     = 19.00
                    endDate       = 24.03.2025
                    endTime       = 20.00
                    locations     = FENSG032
                    isRecurring   = no
                endMeeting
                Meeting "A sub-meeting that starts earlier than its parent"
                    meetingNumber = 126
                    description   = "Another leaf level meeting"
```

```
                    startDate     = 24.03.2025
                    startTime     = 19.00
                    endDate       = 24.03.2025
                    endTime       = 20.00
                    locations     = FENSG035
                    isRecurring   = no
                endMeeting
            endSubMeetings
        endMeeting
        Meeting "Another submeeting of top-level meeting."
            meetingNumber = 127
            description   = "This one is also a leaf-level meeting"
            startDate     = 24.03.2025
            startTime     = 20.00
            endDate       = 24.03.2025
            endTime       = 21.00
            locations     = FENSG035
            isRecurring   = no
        endMeeting
    endSubMeetings
endMeeting
Meeting "A recurring meeting"
    meetingNumber   = 123
    description     = "Top-level recurring meeting"
    startDate       = 03.04.2025
    startTime       = 19.00
    endDate         = 03.04.2025
    endTime         = 21.00
    locations       = FENSG032, FENSG035
    isRecurring     = no
    subMeetings
        Meeting "A sub-meeting of a recurring meeting"
            meetingNumber = 124
            description   = "It also has sub-meetings"
            startDate     = 03.04.2025
            startTime     = 19.00
            endDate       = 03.04.2025
            endTime       = 20.00
            locations     = FENSG032, FENSG035
            isRecurring   = no
            subMeetings
                Meeting "A sub-sub-meeting"
                    meetingNumber = 125
                    description   = "A leaf level meeting, i.e. no submeetings"
                    startDate     = 03.04.2025
```

```
                    startTime     = 19.00
                    endDate       = 03.04.2025
                    endTime       = 20.00
                    locations     = FENSG032
                    isRecurring   = no
                endMeeting
                Meeting "A sub-meeting that starts earlier than its parent"
                    meetingNumber = 126
                    description   = "Another leaf level meeting"
                    startDate     = 03.04.2025
                    startTime     = 19.00
                    endDate       = 03.04.2025
                    endTime       = 20.00
                    locations     = FENSG035
                    isRecurring   = no
                endMeeting
            endSubMeetings
        endMeeting
        Meeting "Another submeeting of top-level meeting."
            meetingNumber = 127
            description   = "This one is also a leaf-level meeting"
            startDate     = 03.04.2025
            startTime     = 20.00
            endDate       = 03.04.2025
            endTime       = 21.00
            locations     = FENSG035
            isRecurring   = no
        endMeeting
    endSubMeetings
endMeeting
Meeting "A recurring meeting"
    meetingNumber   = 123
    description     = "Top-level recurring meeting"
    startDate       = 10.04.2025
    startTime       = 19.00
    endDate         = 10.04.2025
    endTime         = 21.00
    locations       = FENSG032, FENSG035
    isRecurring     = no
    subMeetings
        Meeting "A sub-meeting of a recurring meeting"
            meetingNumber = 124
            description   = "It also has sub-meetings"
            startDate     = 10.04.2025
            startTime     = 19.00
```

```
124             endDate       = 10.04.2025
125             endTime       = 20.00
126             locations     = FENSG032, FENSG035
127             isRecurring   = no
128             subMeetings
129                 Meeting "A sub-sub-meeting"
130                     meetingNumber = 125
131                     description   = "A leaf level meeting, i.e. no submeetings"
132                     startDate     = 10.04.2025
133                     startTime     = 19.00
134                     endDate       = 10.04.2025
135                     endTime       = 20.00
136                     locations     = FENSG032
137                     isRecurring   = no
138                 endMeeting
139                 Meeting "A sub-meeting that starts earlier than its parent"
140                     meetingNumber = 126
141                     description   = "Another leaf level meeting"
142                     startDate     = 10.04.2025
143                     startTime     = 19.00
144                     endDate       = 10.04.2025
145                     endTime       = 20.00
146                     locations     = FENSG035
147                     isRecurring   = no
148                 endMeeting
149             endSubMeetings
150         endMeeting
151         Meeting "Another submeeting of top-level meeting."
152             meetingNumber = 127
153             description   = "This one is also a leaf-level meeting"
154             startDate     = 10.04.2025
155             startTime     = 20.00
156             endDate       = 10.04.2025
157             endTime       = 21.00
158             locations     = FENSG035
159             isRecurring   = no
160         endMeeting
161     endSubMeetings
162 endMeeting
```

Note how the original top–level meeting now has three copies (lines 1–54, lines 55–108, and lines 109–162), together with all the submeetings of the original meeting.

Also note how the dates of the meetings are modified. On planet CS305, one week after 24.03.2025 is 03.04.2005, and two weeks after 24.03.2025 is 10.04.2005.

5 Meeting Scheduler Report

If the input given in MS language is correct both grammatically (with respect to the grammar) and semantically (with respect to the semantic rules given above), then your translator will process the input to print the meetings in all the rooms mentioned in the leaf–level meetings. A top–level meeting without any sub–meetings is also leaf–level meeting.

If a room is mentioned in the program at some place, however if no leaf–level meeting is using that room, the room will have no meetings in it, and no information will be reported for such a room.

The report to be produced will have the following format:

```
ROOM_NAME_1:
STARTDATE_STARTTIME_ENDDATE_ENDTIME_MEETINGNUMBER
STARTDATE_STARTTIME_ENDDATE_ENDTIME_MEETINGNUMBER
STARTDATE_STARTTIME_ENDDATE_ENDTIME_MEETINGNUMBER
ROOM_NAME_2:
STARTDATE_STARTTIME_ENDDATE_ENDTIME_MEETINGNUMBER
ROOM_NAME_3:
STARTDATE_STARTTIME_ENDDATE_ENDTIME_MEETINGNUMBER
STARTDATE_STARTTIME_ENDDATE_ENDTIME_MEETINGNUMBER
...
```

Here `ROOM_NAME_i` is the name of the room, `STARTDATE`, `STARTTIME`, `ENDDATE`, and `ENDTIME` are the start date, the start time, the end date, and the end time of the leaf–level meeting, and `MEETINGNUMBER` is the meeting number of the leaf–level meeting. A concrete example of a report would be something like the following:

```
FENSG032:
24.03.2025_19.40_24.03.2025_20.30_331
24.04.2025_19.40_24.04.2025_20.30_332
FENSG035:
25.03.2025_20.40_25.03.2025_21.30_243
FENSL045:
24.03.2025_19.40_24.03.2025_20.30_313
24.03.2025_19.45_24.03.2025_20.00_221
```

A room can have multiple overlapping meetings in it (e.g. see the meeting of `FENSL045` above). Although that could have been another semantic check to detect and report such collisions, in this homework we excluded this check. Therefore you do not need to check such collisions, but simply give the list of the leaf–level meetings in that room.

Let us conclude this section by giving a complete example of a report to be produced for a given program in MS language. Consider the following program:

```
Meeting "A nice meeting"
    meetingNumber = 888
    description = "An important one"
    startDate = 24.03.2025
    startTime = 19.40
    endDate = 25.03.2025
    endTime = 21.30
    locations = FENSG035, FENSG032, FENSL045
    isRecurring = yes
    frequency = monthly
    repetitionCount = 3
    subMeetings
        Meeting "A nice sub-meeting"
            meetingNumber = 243
            description = "Another very important one"
            startDate = 25.03.2025
            startTime = 20.40
            endDate = 25.03.2025
            endTime = 21.30
            locations = FENSG035
            isRecurring = no
        endMeeting
        Meeting "Another nice sub-meeting"
            meetingNumber = 333
            description = "Another important one"
            startDate = 24.03.2025
            startTime = 19.40
            endDate = 24.03.2025
            endTime = 20.30
            locations = FENSG032
            isRecurring = no
        endMeeting
    endSubMeetings
endMeeting
```

For the above program, the report to be produced is given below:

```
FENSG032:
24.03.2025_19.40_24.03.2025_20.30_333
24.04.2025_19.40_24.04.2025_20.30_333
24.05.2025_19.40_24.05.2025_20.30_333
FENSG035:
25.03.2025_20.40_25.03.2025_21.30_243
25.04.2025_20.40_25.04.2025_21.30_243
25.05.2025_20.40_25.05.2025_21.30_243
```

A couple of remarks about the report produced are:

- Although the leaf–level meeting for `FENSG032` is given later in the program, its schedule is reported before the schedule of `FENSG035`. This is because, we need to produce the report based on the lexicographical orderings of the room names.
- The meetings in a room are ordered with respect to the start date and time. The earliest starting meeting is given first.
- Since the leaf–level meetings are inside a recurring meeting, the leaf–level meetings are repeated, by taking into consideration of the explanations given in Section 4.

6 Example Programs and Outputs

1. If the program is not grammatically correct then like the second homework you have to print ***ERROR***. For example, if we have the program below:

```
Meeting = 12
repetitionCount "Daily"
startDate yearly
```

Then the output should be:

```
ERROR
```

In such a case, this will be the only output that will be produced by your program. There is no need to check the semantic rules, and no need to produce any report for the schedule of the rooms.

2. If a program is grammatically correct, then your program will check the semantic rules given in Section 3. If the input contains violations of the semantic rules then the output should display all the semantic errors. For example, if we have the following program:

```
Meeting "Example Meeting"
   meetingNumber = 125
   description = "Non-recurring time meeting with frequency/repetition"
   startDate = 24.03.2025
   startTime = 14.40
   endDate = 25.03.2025
   endTime = 16.30
   locations = FENSG035
   isRecurring = no
   frequency = monthly
   repetitionCount = 12
endMeeting
```

Then the output of the above program must be as follows:

```
10_UNEXPECTED_FREQUENCY_(125)
11_UNEXPECTED_REPETITIONCOUNT_(125)
```

The way and the order these errors are reported is important. The exact syntax for error messages is explained for each type of semantic check in Section 3.

Since your program will check all the semantic rules given in Section 3, it may find many errors. The errors must be produced in the following order:
(i) The errors are reported as ordered with respect to their line numbers,
(ii) The errors on the same line are ordered lexicographically.

3. If a program is grammatically correct and does not contain any violations of the semantic rules then the output should display the report which is explained in Section 5.

Please recall the rules for ordering the output here as well. In this report, we produce the schedule for the rooms sorted with respect to the names of the rooms. The meetings in a room are ordered with respect to their start date/time.

For example, if we have the following program:

```
Meeting "Meeting with the employees"
    meetingNumber = 1257
    description = "Month-end Reports"
    startDate = 24.03.2025
    startTime = 14.40
    endDate = 25.03.2025
    endTime = 16.30
    locations = FENSG032, FENSG035
    isRecurring = yes
    frequency = monthly
    repetitionCount = 12
    subMeetings
        Meeting "Meeting with Engineers"
            meetingNumber = 222
            description = "Month-end Reports of Engineers"
            startDate = 24.03.2025
            startTime = 14.40
            endDate = 24.03.2025
            endTime = 16.30
            locations = FENSG032, FENSG035
            isRecurring = no
            subMeetings
                Meeting "Meeting with Alice"
```

```
                    meetingNumber = 2568
                    description = "Month-end Reports of Alice"
                    startDate = 24.03.2025
                    startTime = 14.40
                    endDate = 24.03.2025
                    endTime = 15.30
                    locations = FENSG032
                    isRecurring = no
                endMeeting
            endSubMeetings
        endMeeting

        Meeting "Meeting with Accountants"
            meetingNumber = 432
            description = "Month-end Reports of Accountants"
            startDate = 25.03.2025
            startTime = 14.40
            endDate = 25.03.2025
            endTime = 16.30
            locations = FENSG035
            isRecurring = no
            subMeetings
                Meeting "Meeting with George"
                    meetingNumber = 781
                    description = "Month-end Reports of George"
                    startDate = 25.03.2025
                    startTime = 14.40
                    endDate = 25.03.2025
                    endTime = 15.30
                    locations = FENSG035
                    isRecurring = no
                endMeeting

                Meeting "Meeting with Hagi"
                    meetingNumber = 4
                    description = "Month-end Reports of Hagi"
                    startDate = 25.03.2025
                    startTime = 15.40
                    endDate = 25.03.2025
                    endTime = 16.30
                    locations = FENSG035
                    isRecurring = no
                endMeeting
            endSubMeetings
        endMeeting
```

```
69          endSubMeetings
70      endMeeting
```

Then the output for the above program must be as follows:

```
FENSG032:
24.03.2025_14.40_24.03.2025_15.30_2568
24.04.2025_14.40_24.04.2025_15.30_2568
24.05.2025_14.40_24.05.2025_15.30_2568
24.06.2025_14.40_24.06.2025_15.30_2568
24.07.2025_14.40_24.07.2025_15.30_2568
24.08.2025_14.40_24.08.2025_15.30_2568
24.09.2025_14.40_24.09.2025_15.30_2568
24.10.2025_14.40_24.10.2025_15.30_2568
24.11.2025_14.40_24.11.2025_15.30_2568
24.12.2025_14.40_24.12.2025_15.30_2568
24.01.2026_14.40_24.01.2026_15.30_2568
24.02.2026_14.40_24.02.2026_15.30_2568
FENSG035:
25.03.2025_14.40_25.03.2025_15.30_781
25.03.2025_15.40_25.03.2025_16.30_4
25.04.2025_14.40_25.04.2025_15.30_781
25.04.2025_15.40_25.04.2025_16.30_4
25.05.2025_14.40_25.05.2025_15.30_781
25.05.2025_15.40_25.05.2025_16.30_4
25.06.2025_14.40_25.06.2025_15.30_781
25.06.2025_15.40_25.06.2025_16.30_4
25.07.2025_14.40_25.07.2025_15.30_781
25.07.2025_15.40_25.07.2025_16.30_4
25.08.2025_14.40_25.08.2025_15.30_781
25.08.2025_15.40_25.08.2025_16.30_4
25.09.2025_14.40_25.09.2025_15.30_781
25.09.2025_15.40_25.09.2025_16.30_4
25.10.2025_14.40_25.10.2025_15.30_781
25.10.2025_15.40_25.10.2025_16.30_4
25.11.2025_14.40_25.11.2025_15.30_781
25.11.2025_15.40_25.11.2025_16.30_4
25.12.2025_14.40_25.12.2025_15.30_781
25.12.2025_15.40_25.12.2025_16.30_4
25.01.2026_14.40_25.01.2026_15.30_781
25.01.2026_15.40_25.01.2026_16.30_4
25.02.2026_14.40_25.02.2026_15.30_781
25.02.2026_15.40_25.02.2026_16.30_4
```

7 How to Submit

Submit your flex file and bison file named as `username-hw3.flx` and `username-hw3.y` respectively, where `username` is your SU-Net username. You may use additional files, such as a header file or additional C files. Please also upload such files as well. We will compile your files by using the following commands:

```
flex username-hw3.flx
bison -d username-hw3.y
gcc -o username-hw3 lex.yy.c username-hw3.tab.c -lfl
```

So, make sure that these three commands are enough to produce the executable. If we assume that there is an MS file named `example1.ms`, we will try out your parser by using the following command line:

```
username-hw3 < example1.ms
```

8 Notes

- **Important**: Name your files as you are told and **don't zip them.** [-20 points otherwise]
- **Important**: Make sure you include the right file in your scanner and make sure you can compile your parser using the commands given in Section 7. If we are not able to compile your code with those commands your **grade will be zero for this homework.**
- **Important**: Some test cases are shared on the `cs305.sabanciuniv.edu` server. We are hoping to share a golden later, but no promise currently.
- **Important: Since this homework is evaluated automatically make sure your output is exactly as it is supposed to be. Some of the points that we can think of are:**
  - **There should be no extra space at the beginning or at the end of a line.**
  - **Make sure that the spelling is as it is given in the homework document.**
  - **We check in a case-sensitive manner (e.g. "ERROR" $\neq$ "error")**
- You may get help from our TA or from your friends. However, **you must implement the homework by yourself**.
- Start working on the homework immediately.

- If you develop your code or create your test files on your own computer (not on `cs305.sabanciuniv.edu`), there can be incompatibilities once you transfer them to the `cs305` server. Since the grading will be done automatically on the `cs305` server, we strongly encourage you to do your development on the `cs305` server, or at least test your code on the `cs305` server before submitting it. If you prefer not to test your implementation on the `cs305` server, this means you accept to take the risks of incompatibilities. Even if you may have spent hours on the homework, you can easily get 0 due to such incompatibilities.

LATE SUBMISSION POLICY

Late submission is allowed subject to the following conditions:

- Your homework grade will be decided by multiplying what you get from the test cases by a "submission time factor (STF)".
- If you submit on time (i.e. before the deadline), your STF is 1. So, you don't lose anything.
- If you submit late, you will lose 0.01 of your STF for every 5 mins of delay.
- We will not accept any homework later than 500 mins after the deadline.
- SUCourse's timestamp will be used for STF computation.
- If you submit multiple times, the last submission time will be used.