



What is ANTLR?

- ANTLR (ANother Tool for Language Recognition) is a powerful parser generator.
- ANTLR is a tool that translates your grammar to a parser/lexer in Java (or another target language) and the runtime needed by the generated parsers/lexers.
- From a grammar, ANTLR generates a parser that can build parse trees.

Identifiers

- Lexer rules names always start with a capital letter [UpperCase].
- Parser rules names always start with a lowercase letter. The initial character can be followed by uppercase and lowercase letters, digits, and underscores.
- Here are some sample names:
 - ID, ZERO //lexer rules
 - expr, start_rule //parser rules

Literals


- ANTLR does not distinguish between character and string literals as most languages do. All literal strings one or more characters in length are enclosed in single quotes, such as:
 - '0'
 - 'Hello'

Fragments

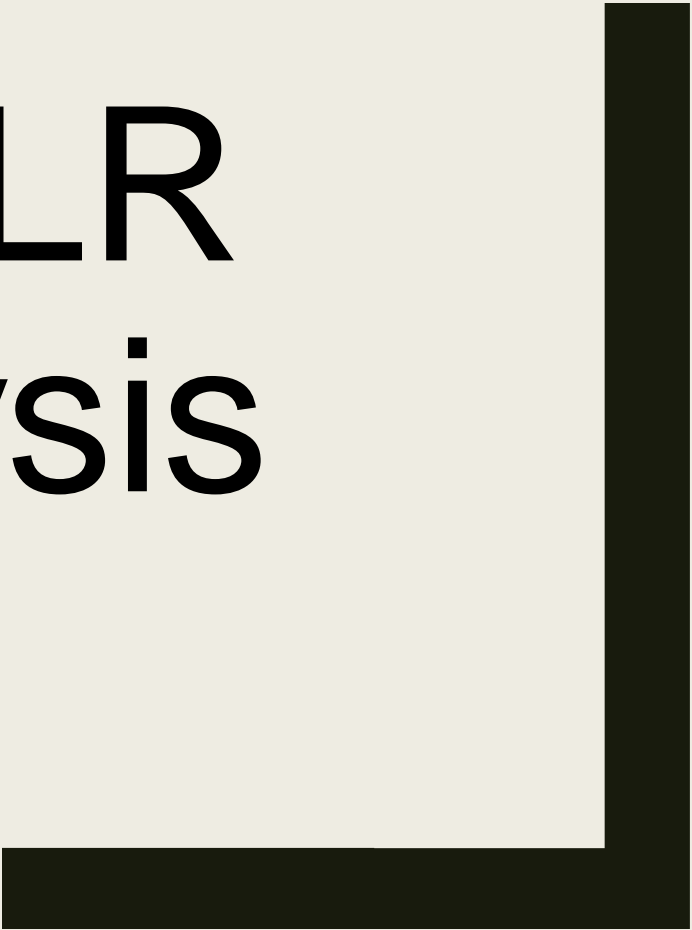
- Fragments are reusable parts of lexer rules which cannot match on their own - they need to be referenced from a lexer rule.

ANTLR

- For more information:
- <https://github.com/antlr/antlr4/blob/master/doc/index.md>



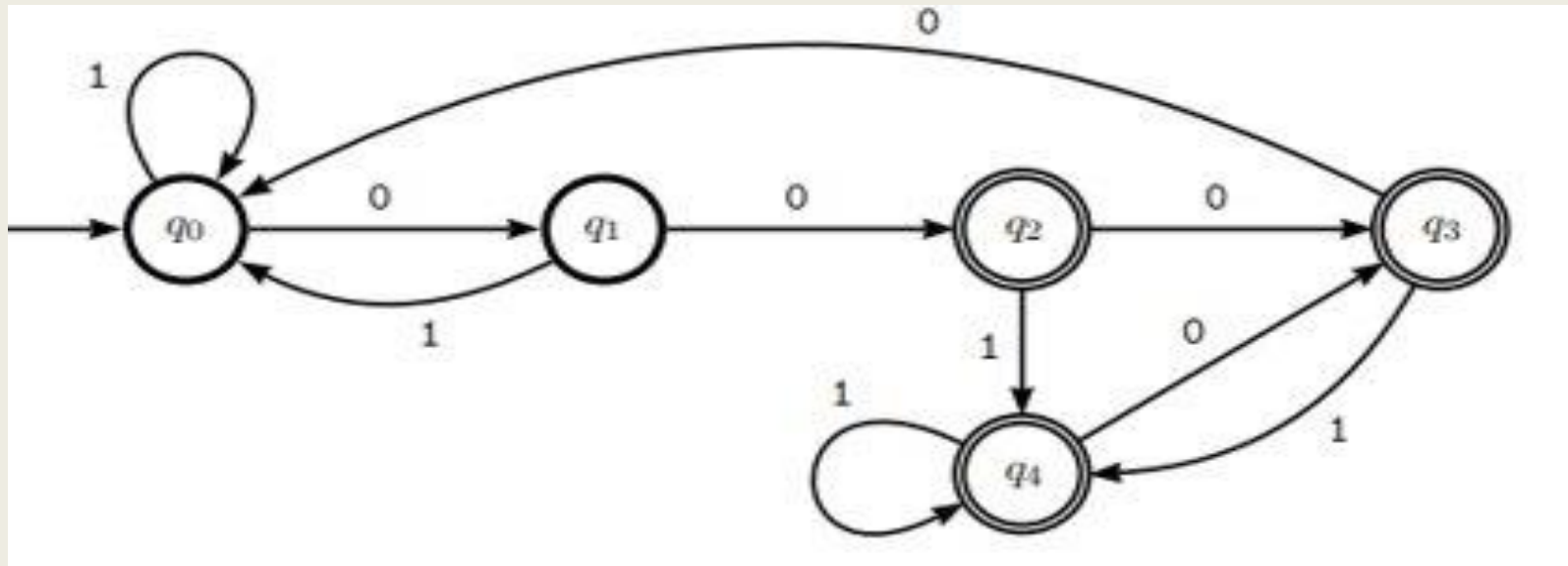
Task 8: ANTLR Lexical Analysis



Task 8: ANTLR Lexical Analysis

- You will implement an ANTLR lexical analyzer for the following fallback DFA:

0,1,0,;1,2,0,;2,3,4,Q2;3,0,4,Q3;4,3,4,Q4#2,3,4



Task 8: ANTLR Lexical Analysis

- Your task is to get the regular expression for this FDFA
- Then write its grammar using ANTLR
- For example, running the lexical analyzer implementing the FDFA on the string:
 - 00010001 produces the output 0 0 0 1 0 , Q 3 , 0 0 1 , Q 4
-



Setup

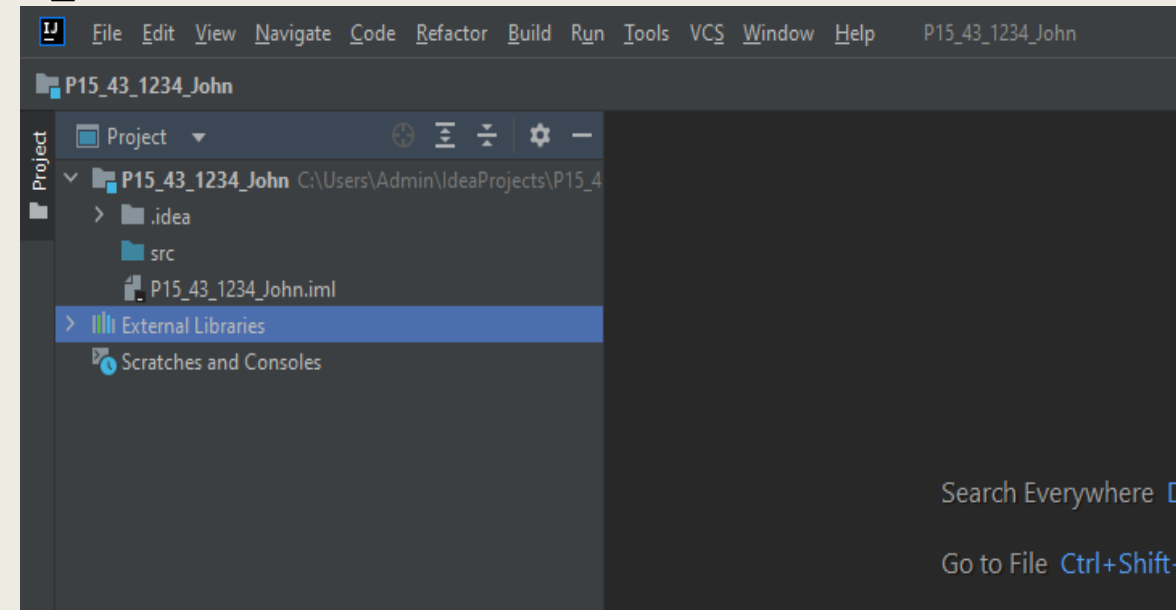
Setup

We should first:

1. Make sure you have JAVA installed and running.
2. Download IntelliJ IDEA: <https://www.jetbrains.com/idea/download/>
 - a. Please install and activate it before the session.
 - b. Either Community or Ultimate will work.
 - c. Make sure to download the latest version (2022.1).
3. Download ANTLR v4: <https://www.antlr.org/download/antlr-4.10.1-complete.jar>
4. Download ANTLR v4 plugin: <https://plugins.jetbrains.com/files/7358/169193/antlr-intellij-plugin-v4-1.18.zip>

Setup

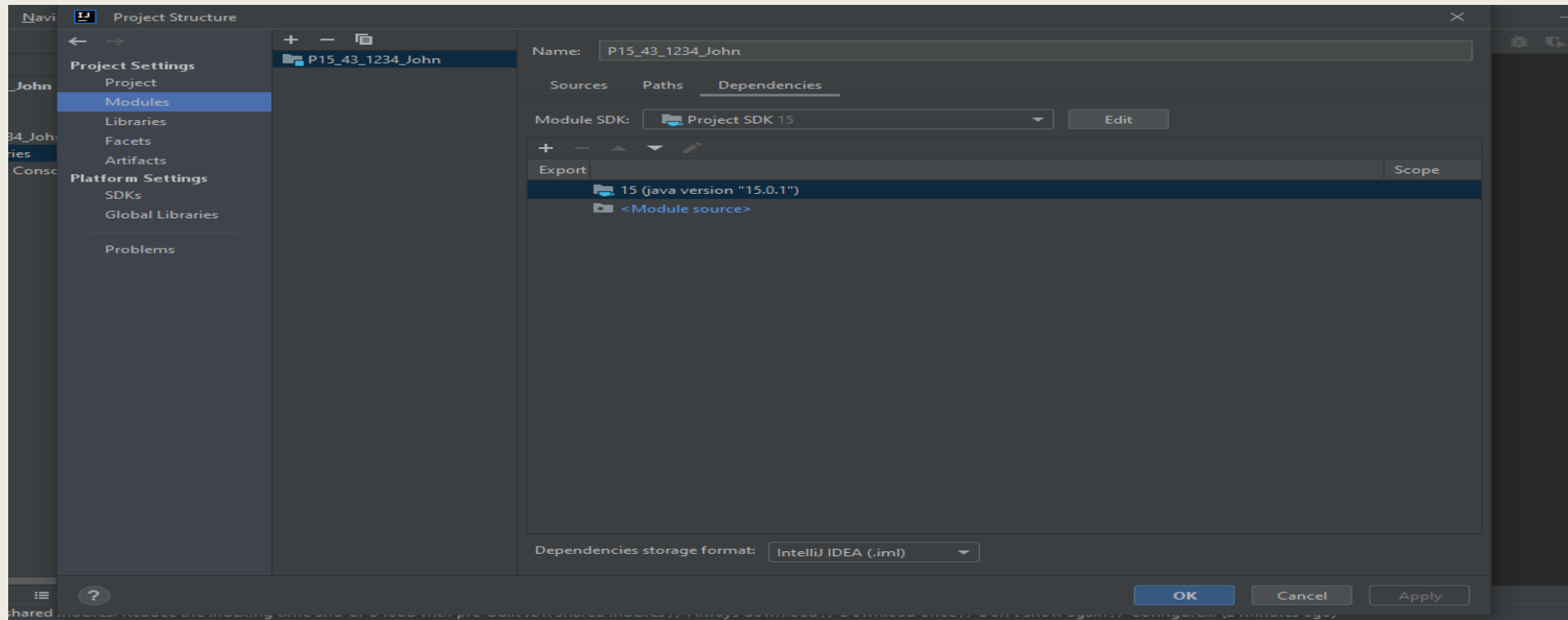
- Open intellj
- Create a new project:
 - **File > New > Project**
 - Filename: [LabNo_ID_Name], ex: P15_43_1234_John



Setup

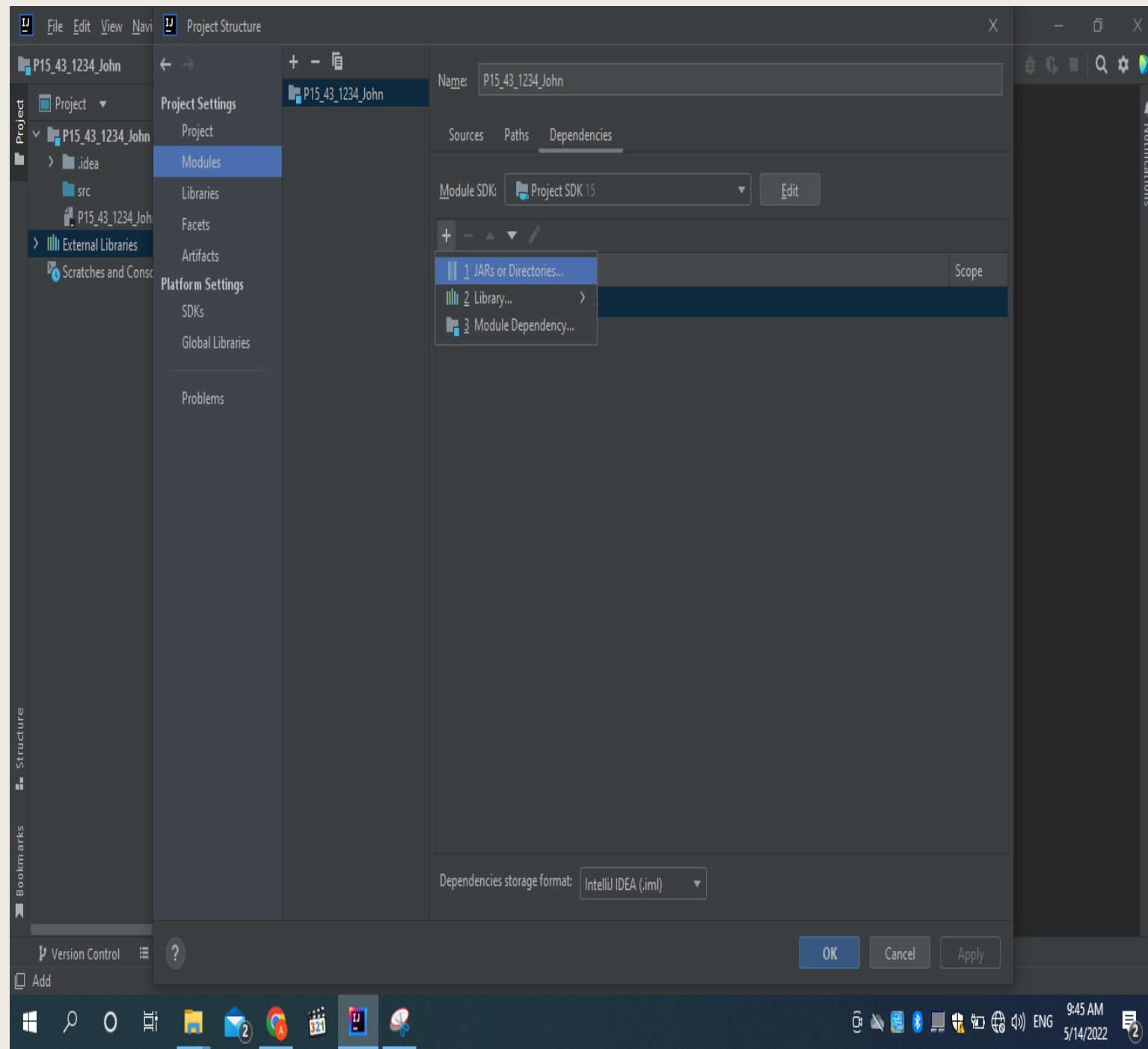
You should add the ANTLR jar in intellj:

- **File > Project Structure > Modules > Dependencies**



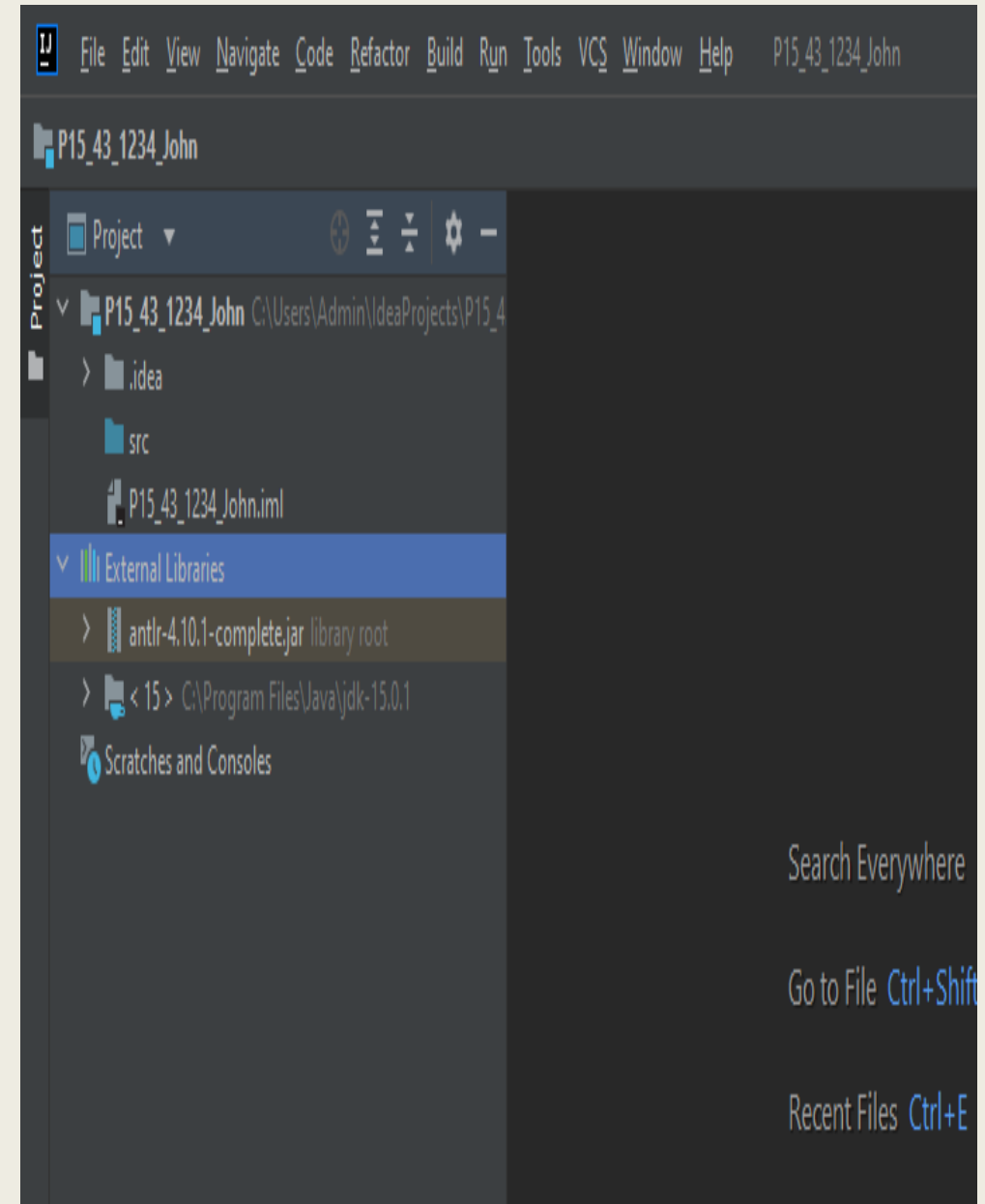
Setup

- Then Add '+' > **1 JARs or Directories**
- Choose ANTLR (.jar) > **Ok** > **OK**



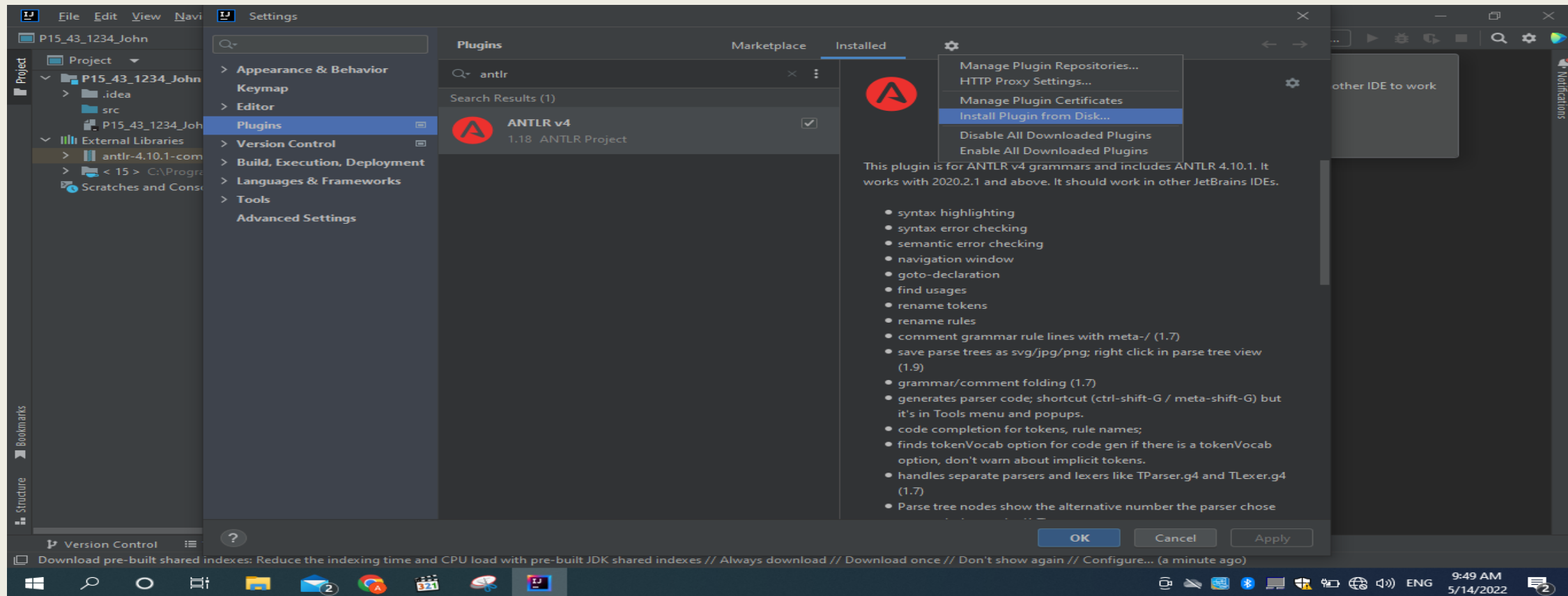
Setup

- You will find the jar appears in the **External Libraries**



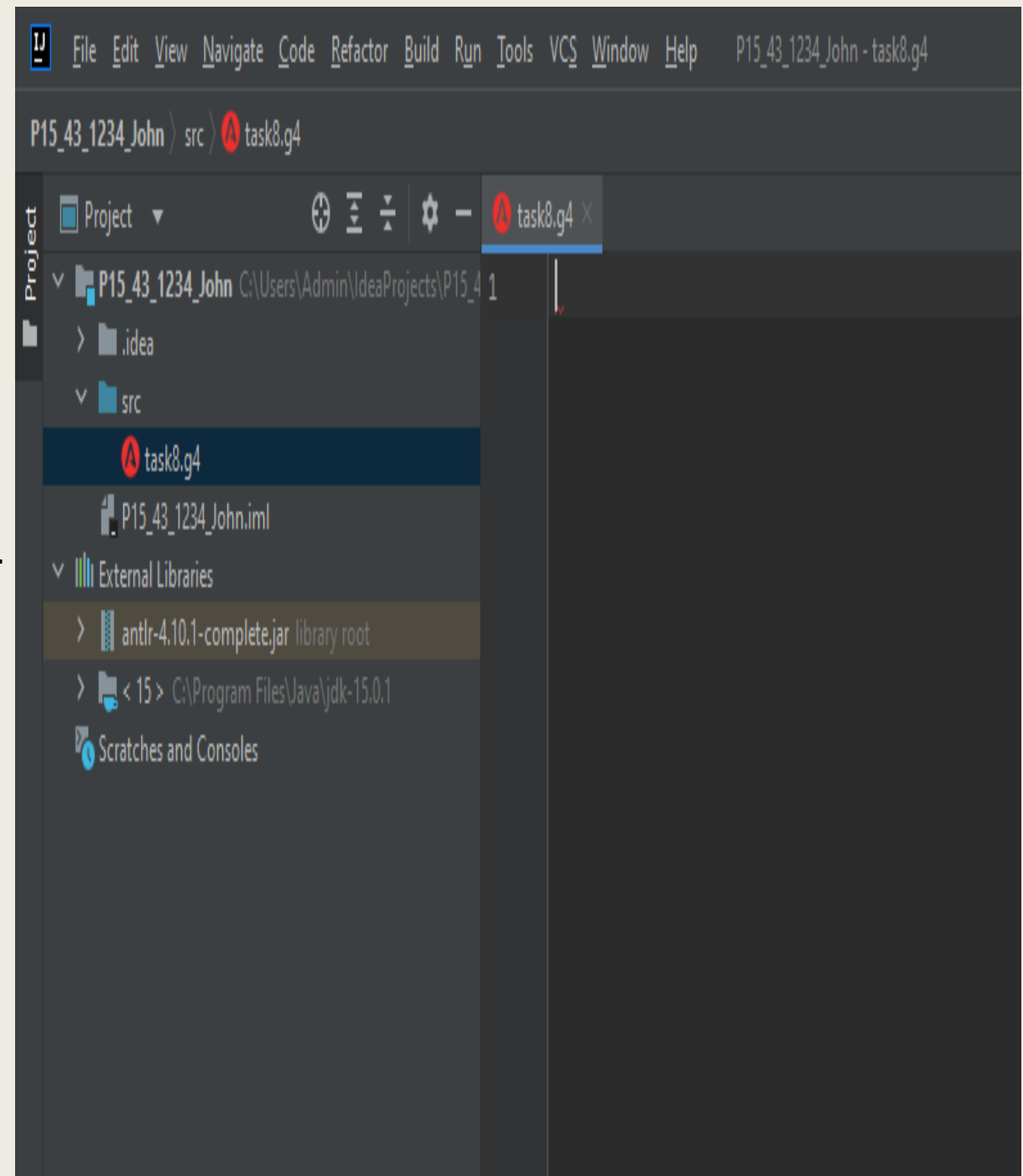
Setup

- File > Settings > Plugins > Add the plugin either from:
 - Marketplace > Write ANTLR OR Install Plugin from disk (.zip)



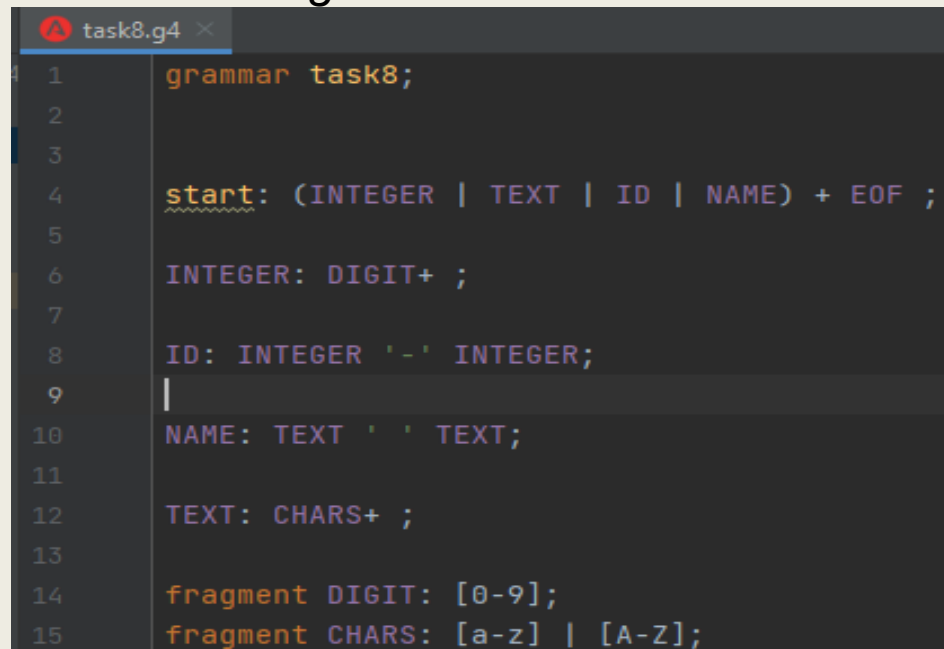
Setup

- From **src**> Right Click > **New** > **File** > **task8.g4**



Setup

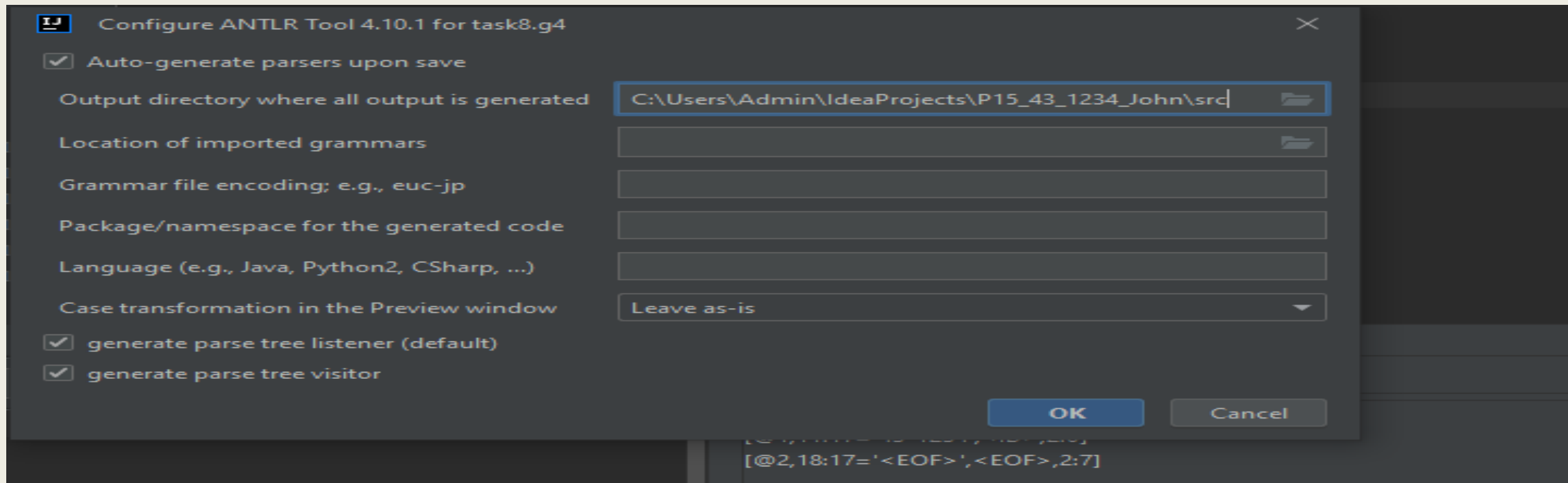
- Start write your Grammar
 - *The filename is the same name of the grammar*



```
task8.g4 x
1 grammar task8;
2
3
4 start: (INTEGER | TEXT | ID | NAME) + EOF ;
5
6 INTEGER: DIGIT+ ;
7
8 ID: INTEGER '-' INTEGER;
9 |
10 NAME: TEXT ' ' TEXT;
11
12 TEXT: CHARS+ ;
13
14 fragment DIGIT: [0-9];
15 fragment CHARS: [a-z] | [A-Z];
```

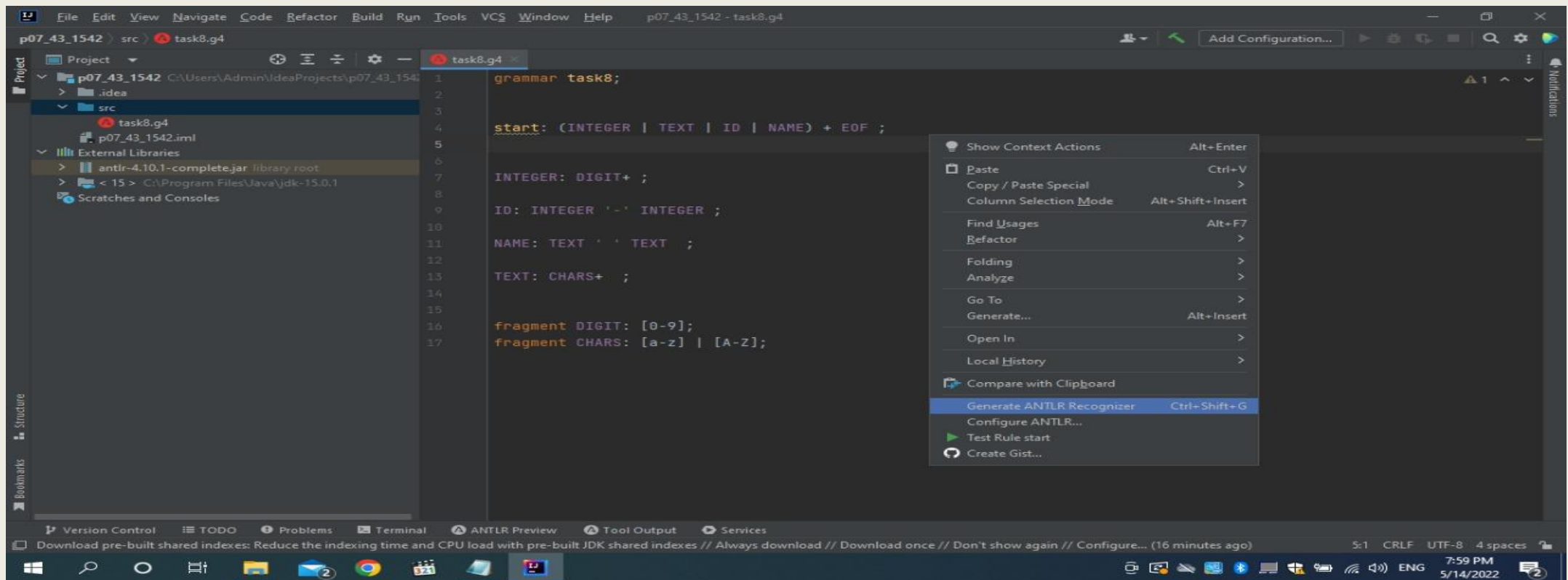
Setup

- Right click in the g4 file > Configure ANTLR > Output directory > Change to the src directory



Setup

- Right click in the g4 file > Generate ANTLR Recognizer



Setup

- We can also run the grammar and see the parse tree:
 - Right click on the start of your regular expression in the grammar, then choose **Test Rule start**
- Now **Antlr Preview** is opened at the bottom, you can write any string, and see its parse tree.
- Also if there is an error in tokens, it will appear at the bottom.

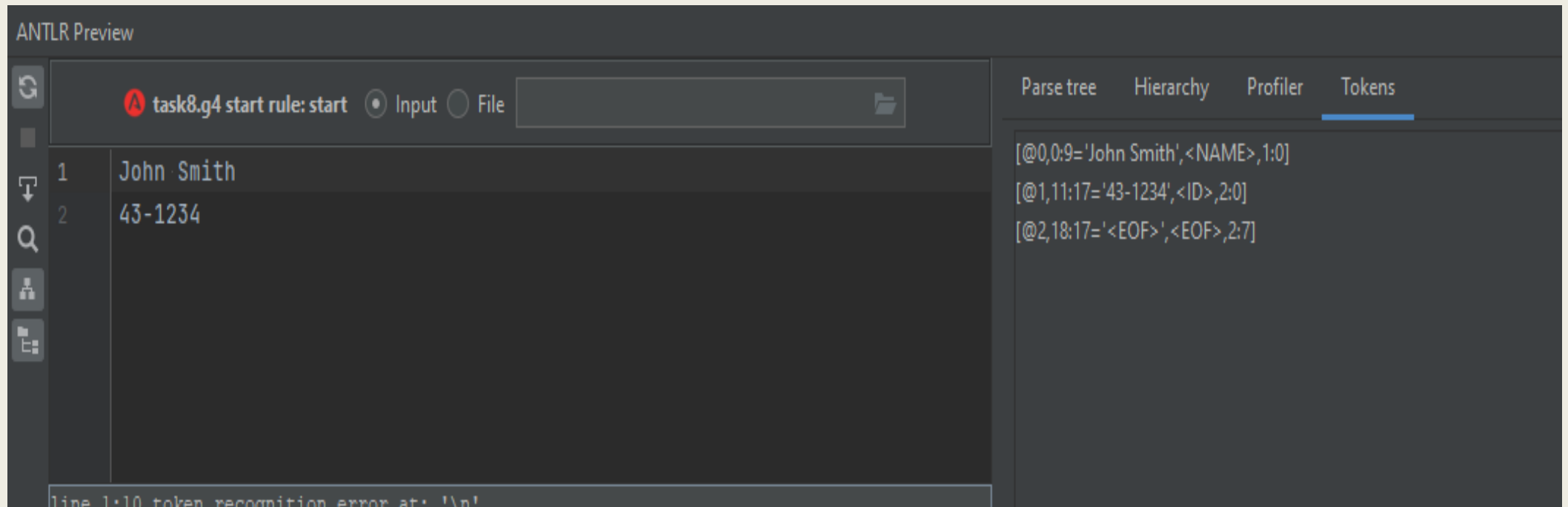
The screenshot shows the ANTLR Preview window with the following components:

- Header:** "ANTLR Preview"
- Toolbar:** Includes a refresh icon, a red error icon, and a text field containing "task8.g4 start rule: start". Below this are radio buttons for "Input" (selected) and "File", followed by a folder icon.
- Input Area:** A text area with two lines of input:

```
1 John Smith
2 43-1234
```
- Parse Tree View:** A tree diagram showing the root node "start" branching into three children: "NAME: 'John Smith'", "ID: '43-1234'", and "<EOF>".
- Navigation Tabs:** "Parse tree" (active), "Hierarchy", "Profiler", and "Tokens".

Setup

- We can also run the grammar and see the parse tree:
 - Right click on the start of your regular expression in the grammar, then choose **Test Rule start**
- Now **Antlr Preview** is opened at the bottom, you can write any string, and see its parse tree.
- Also if there is an error in tokens, it will appear at the bottom.



Setup

- Create a new java class
- New > Java Class > Test
- You will find this class on cms website

