

# Laboratorul 2.

## Analiză lexicală

### Configurare proiect

Descărcați secheletul de laborator de pe Moodle. Dacă folosiți IntelliJ sau Eclipse, asigurați-vă că JDK-ul și calea către jar-ul de ANTLR sunt configurate corect în cadrul proiectului. În mașina virtuală, jar-ul se află la `/usr/local/bin/antlr4.jar`.

### Exercițiul 0

În cadrul laboratorului, vom utiliza limbajul CPLANG. Începeți prin a parcurge manualul acestuia de pe Moodle.

De asemenea, familiarizați-vă cu scheletul de laborator, implementat la cursul anterior, și rulați clasa `Test` pe fișierul `program.txt`. Parcurgeți și comentariile din fișierul `CPLangLexer.g4`.

Obiectivul laboratorului este ca analizorul lexical implementat de voi să recunoască în final toți *token*-ii din fișierul `manual.txt`, care conține secvența de cod din manualul CPLANG. Un exemplu de ieșire posibilă a analizorului pentru acest fișier se găsește în `reference.txt`.

### Exercițiul 1

Implementați analizorul lexical astfel încât să poată clasifica identificatori (nume de variabile și de funcții), numere întregi, operatori (aritmetici, de comparație, de atribuire), simbolul „;” și cuvintele cheie (`if`, `then`, `else`, `fi`, `true`, `false`).

Observații:

- Analizorul lexical va considera toți literalii întregi ca fiind numere pozitive. Semnul „minus” va fi tratat doar ca operator aritmetic.
- Identificatorii încep cu litera mică, și pot conține apoi litere mici, litere mari și *underscore*.

## Exercițiul 2

Extindeți analizorul astfel încât să trateze corect numele de tipuri de date, parantezele, acoladele și comentariile pe un singur rând.

Observații:

- Numele de tipuri de date încep mereu cu o literă mare. Verificarea validității tipurilor se va face în pasul de analiză semantică.
- Comentariile vor fi ignorate (acțiunea `skip`).

## Exercitiul 3

Implementați tratarea literalilor cu virgula mobilă și a comentariilor bloc imbricate împerecheate.

Observații:

- Comentariile care nu au simbol-pereche (ex: `"/ * / * */`) sunt considerate erori lexicale. Prin urmare veți afișa un mesaj la `stderr` dacă întâlniți o astfel de situație. Utilizați pentru aceasta acțiuni introduse direct în specificația lexicală (vezi ca exemplu regula `STRING`) și `token`-ul predefinit `EOF`.
- Valoarea `Float` `.25` este echivalentă la nivel semantic cu `0.25`. Valoarea `3.` este echivalentă cu `3.0`. Valoarea `".` nu reprezintă un literal `Float` valid. Pentru recunoașterea literalilor de forma `.25` va trebui să extindeți regula existentă din schelet.