**Documentație Laborator 1**

**Muntea Andrei – Marius (235)**

***I Cerință***

Se consideră un mini-limbaj de programare. Scrieți un analizor lexical pentru acest mini-limbaj. Datele de intrare pentru analizorul lexical vor fi reprezentate de un fișier conținând cod sursă pentru un program scris în mini-limbajul respectiv. Rezultatele vor fi Forma Internă a Programului (FIP) și Tabela de Simboluri (TS). Programul trebuie să identifice erorile lexicale și să specifice locul acestora.

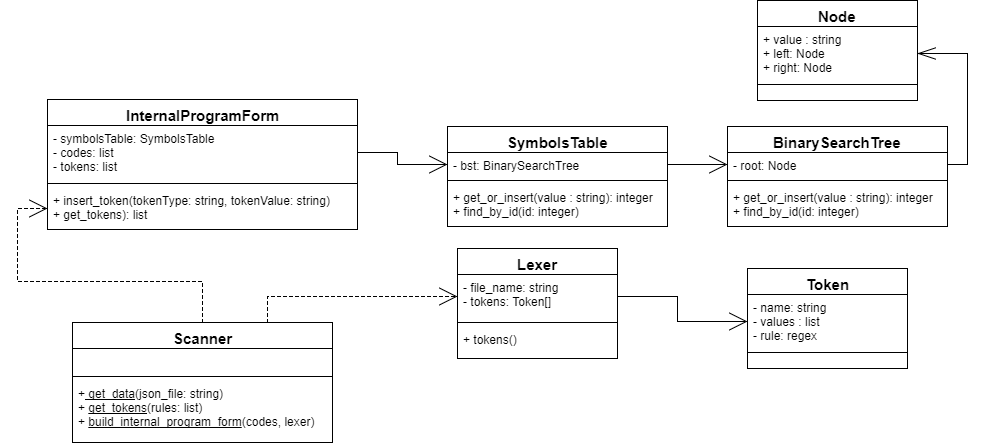
* Identificatorii trebuie să aibă lungime arbitrară, mai mică decât 250 de caractere
* Tabela de simboluri va fi unică pentru identificatori și constante
* Tabela de simboluri va conține un arbore binar ordonat lexicografic

***II Idee de rezolvare***

Vom defini într-un fișier în format *json* ( ”codes.json” ) codurile prin care se identifică fiecare dintre simboluri, constante, identificatori respectiv cuvinte cheie precum și regulile prin care se formează acestea. Tot în acest fișier specificăm dacă după un anumit atom urmează o clasă specifică de atomi (ex. după cuvintele cheie urmează obligatoriu un atom separator). Programul va fi scris într-un fișier separat ( ”program.txt” ). Considerăm că regulile descrise în fișier sunt consistente.

Într-o primă fază se va încărca jsonul cu reguli și coduri apoi va încerca să obțină un șir de atomi folosind regulile respective. Construcția formei interne a programului se face într-o manieră leneșă. Pe măsură ce avem un atom, îl înserăm. Ca identificator în arborele binar de căutare, vom returna *id-ul* obiectului. Folosim metoda *id(object)* care returnează un număr unic pe toată durata de viață a obiectului.

***III Diagramă***

******