Limbaje Formale și Compilatoare (LFC)

- Curs -

Ș.l.dr.ing Octavian MACHIDON

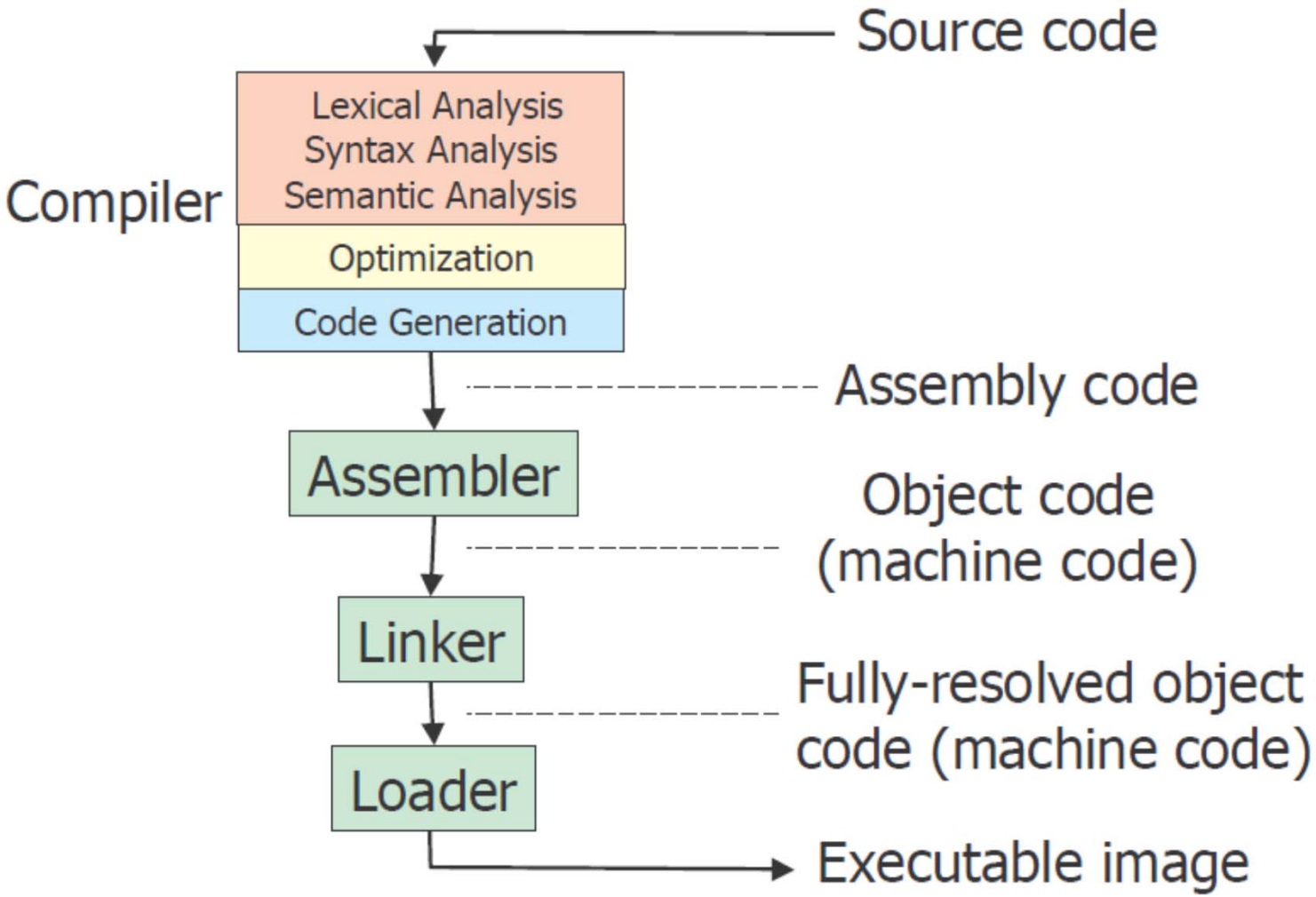
octavian.machidon@unitbv.ro

Universitatea
Transilvania
din Brașov

Astăzi

- Analiza lexicală

Reamintire: etapele compilării

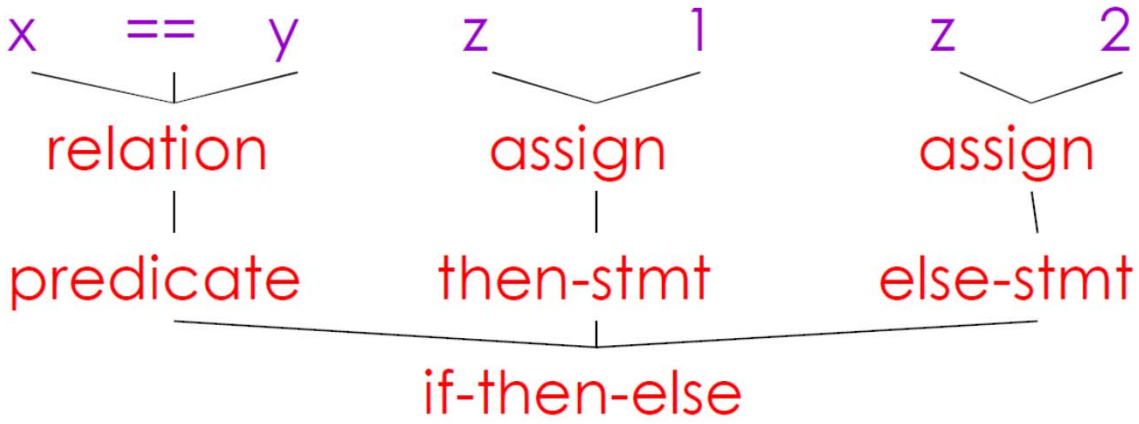


Etapele compilării - exemplificare

- **Analiza lexicală**: recunoașterea cuvintelor

If|x|==|y| then|z|=| 1;| else| z| =| 2;|

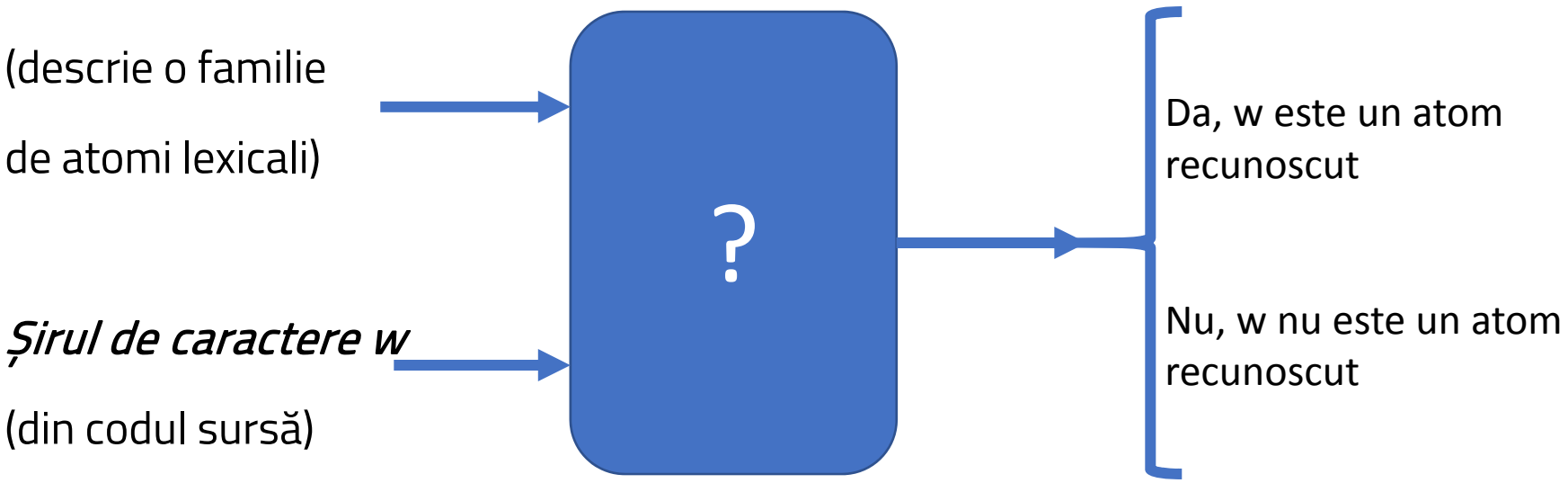
- **Parsarea (Analiza sintactică)**: recunoașterea structurii frazei/propoziției



- **Analiza semantică**: recunoașterea sensului
  - În cazul compilatorului, există reguli care se aplică pentru a detecta și semnala ambiguități

Cum se folosesc expresiile regulate?

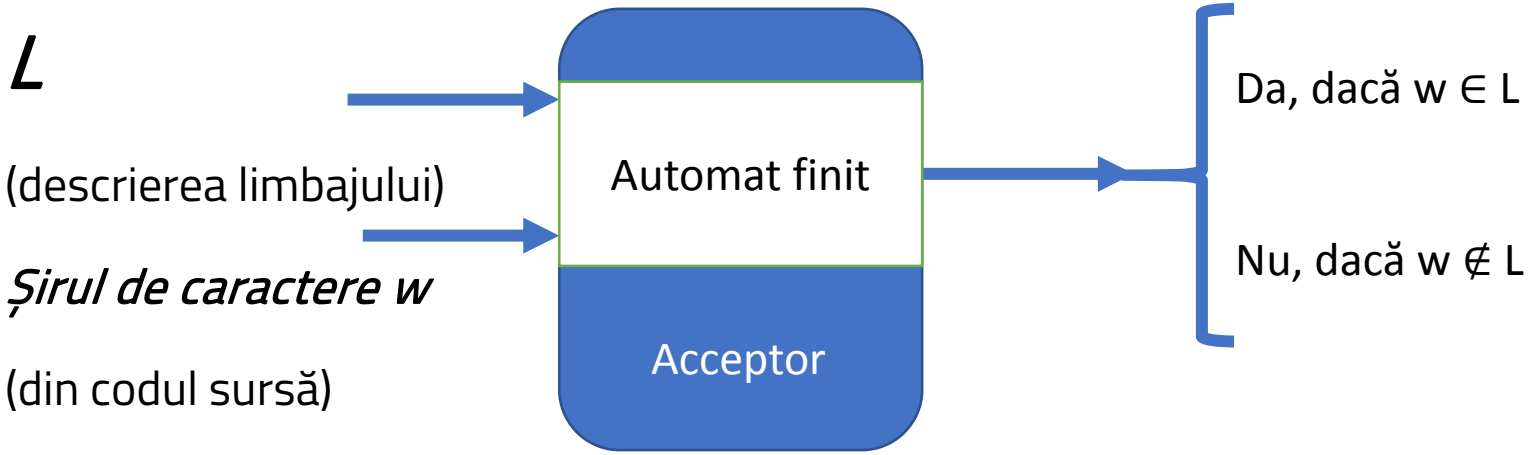
- Având expresia regulată R (R ∈ RE) și șirul de caractere w, e necesar un mecanism care să determine dacă w∈L(R)



- Un astfel de mecanism se numește acceptor

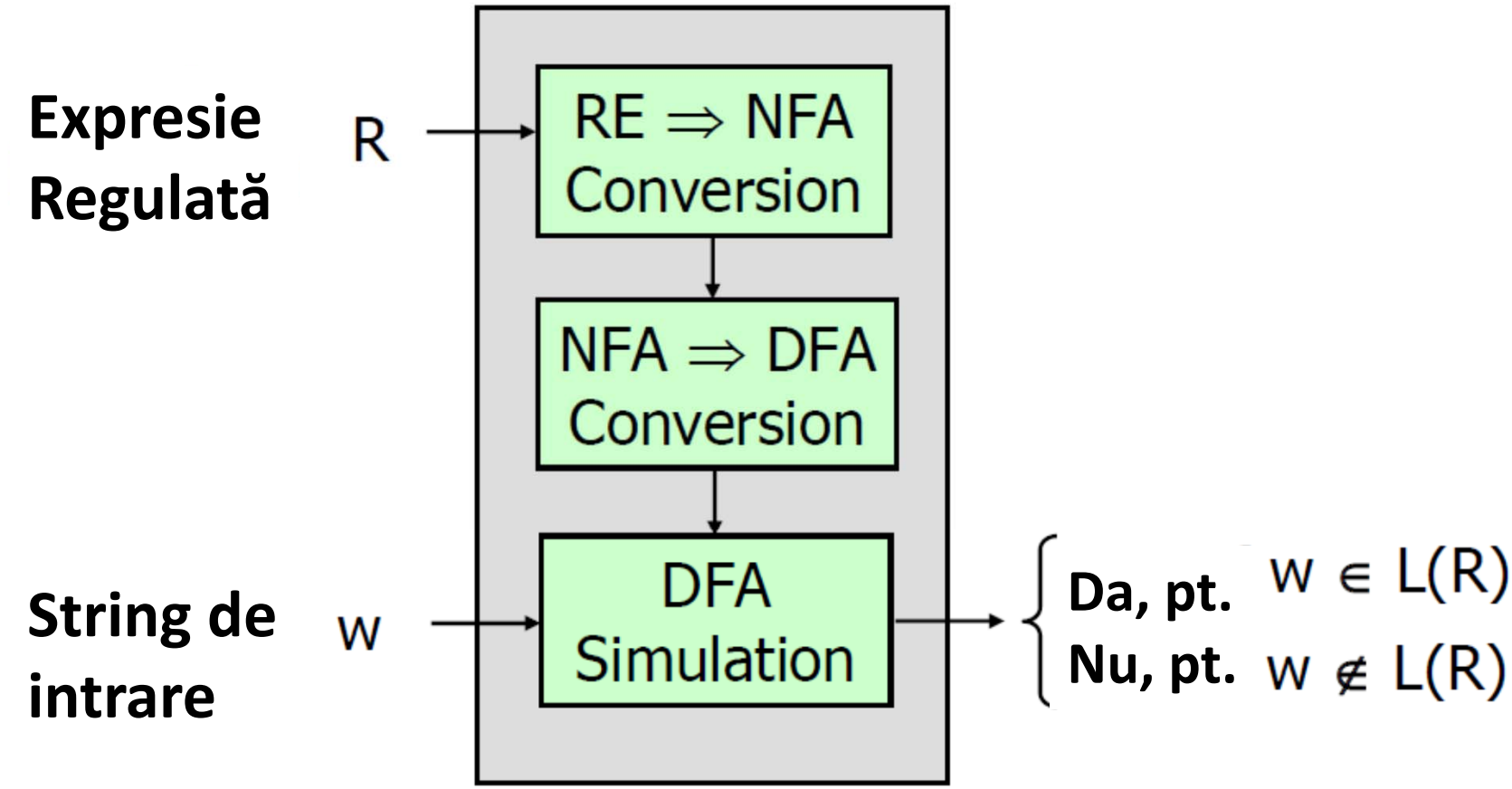
Acceptorii

- Acceptorul determină dacă un șir de caractere aparține sau nu unui limbaj L

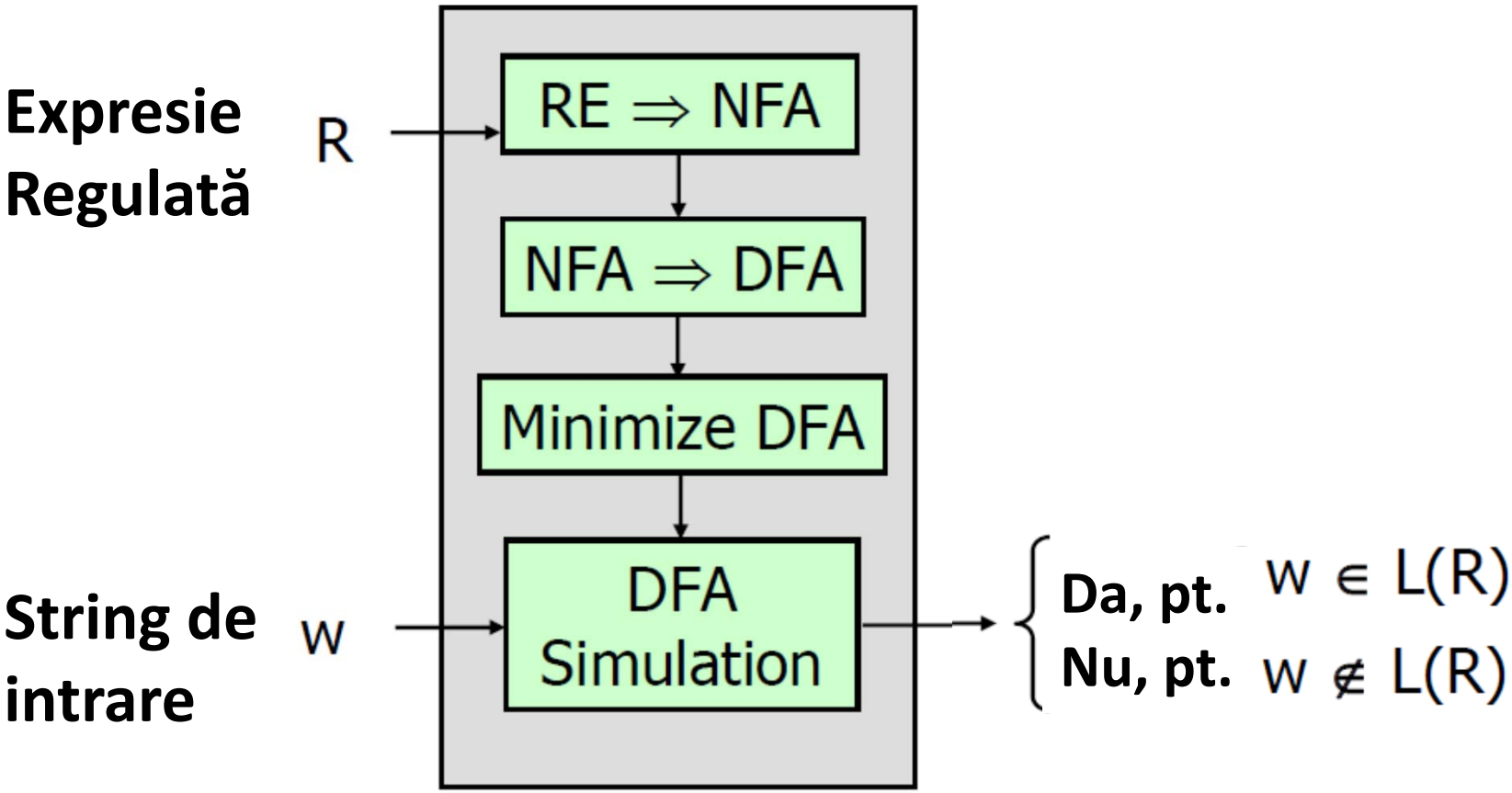


- Automatele finite sunt acceptori pentru limbajele descrise de expresii regulate

Structura unui acceptor



Optimizarea acceptorului

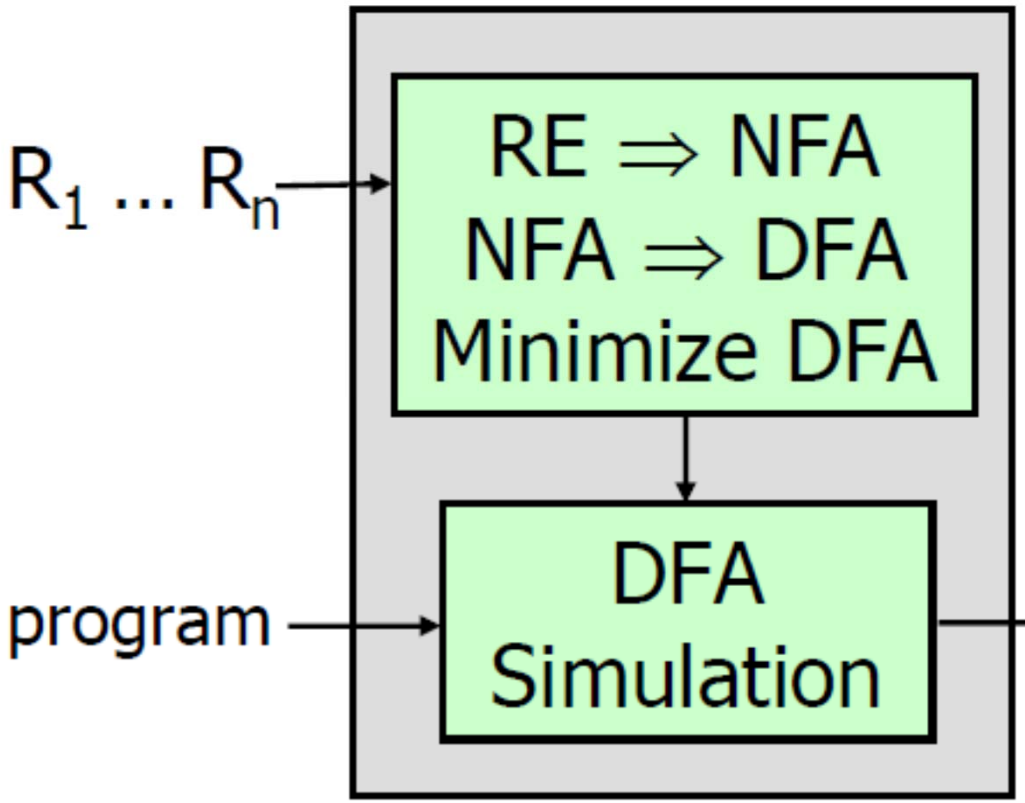


Analizoare lexicale vs. Acceptori

- Analizoarele lexicale folosesc același mecanism, dar:
  - Conțin descrierea mai multor RE corespunzătoare mai multor atomi lexicali
  - La ieșire produc o secvență de atomi care se potrivesc (sau o eroare în caz de nepotrivire)
  - Întotdeauna returnează atomul lexical corespunzător celei mai lungi potriviri
  - Pentru mai mulți atomi lexicali de aceeași lungime care se potrivesc, se folosesc reguli de priorități

Analizor lexical

Expresii Regulate pt. Atomii lexicali



String de intrare

Flux de atomi (și erori)

Automatizarea analizei lexicale

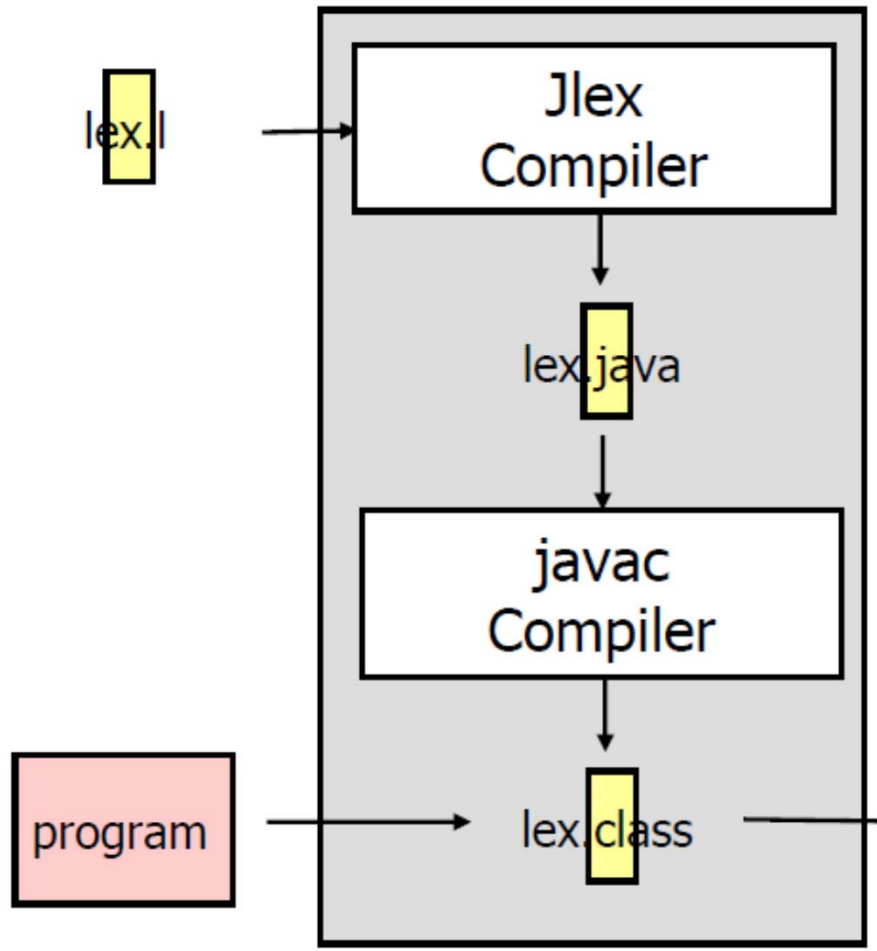
- Procesul de analiză lexicală poate fi automatizat:
  - RE → AFN → AFD → AFD minimizat
  - AFD minimizat → Analizor lexical (Program de simulare al AFD)
- Trebuie să specificăm:
  - Expresiile regulate pentru atomi
  - Regulile de priorități în cazul potrivirii mai multor atomi de aceeași lungime

Generatori automați de lexeri

- Input: specificațiile atomilor lexicali
  - Listă de expresii regulate în ordinea priorității
  - Acțiuni asociate pentru fiecare expresie regulate
- Output: program lexer
  - Aplicație care citește un șir de caractere și îl separă în atomi lexicali pe baza expresiilor regulate (sau raportează erori lexicale dacă există caractere nerecunoscute)

Generator de analizori lexicali

Expresii Regulate pt. Atomii lexicali



String de intrare

Flux de atomi (și erori)

Fișierul de specificații Jlex

- Jlex = Generator de analizori lexical
  - Scris în Java
  - Generează un analizor lexical Java
- Are trei părți:
  - Preambul, care conține declarații de tip package/import
  - Definiții, care conțin abrevierea expresiilor regulate
  - Expresii regulate și acțiuni corespunzătoare, care conțin:
    - Lista cu expresiile regulate pentru toți atomii
    - Acțiunile corespunzătoare pentru fiecare atom (codul Java care se va executa pentru fiecare atom recunoscut)

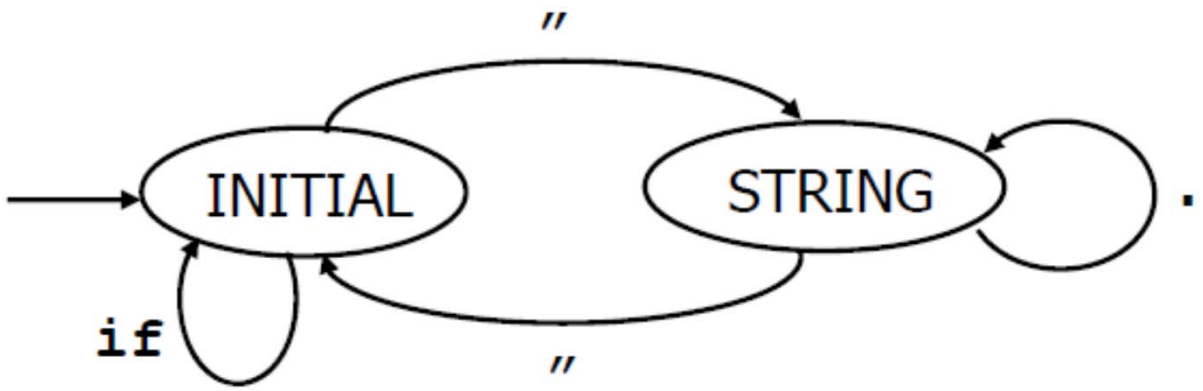
Exemplu fișier specificatori

```
Package Parse;

Import Error.LexicalError;

%%

digits = 0|[1-9][0-9]*

letter = [A-Za-z]

identifier = {letter}({letter}|[0-9_])*

whitespace = [\\t\n\r]+

%%

{whitespace}        {/* discard */}

{digits}            { return new
                      Token(INT, Integer.valueOf(yytext()); }

“if”                { return new Token(IF, null); }

“while”             { return new Token(WHILE, null); }

{identifier}        { return new Token(ID, yytext()); }

.                   { ErrorMsg.error(“illegal character”); }
```

Stări de pornire

- Mecanism care specifică starea în care se începe execuția AFD-ului
- Stările se declară în a doua secțiune a fișierului
  - %state STATE
- Stările sunt introduse ca prefixe ale expresiilor regulate în a treia secțiune:
  - <STATE> regex {action}
- Se specifică starea curentă în acțiuni:
  - yybegin(STATE)
- Există o stare inițială predefinită: YYINITIAL

Exemplu



- %%
- %state STRING
- %%
- <YYINITIAL> “if”{ return new Token(IF, null); }
- <YYINITIAL> “\””{ yybegin(STRING); …}
- <STRING> “\””{ yybegin(YYINITIAL); …}
- <STRING> .{ …}

Stări de pornire și expresii regulate

- Folosirea stărilor de pornire permite lexer-ului să recunoască mai multe decât doar expresii regulate (sau AFD-uri)
- Motivul: lexer-ul poate „sări” între diferite stări în cadrul acțiunilor semantice folosind yybegin(STATE)

- Exemplu: comentarii în comentarii
  - Incrementarea unei variabile globale la găsirea unei paranteze deschise și decrementarea la o paranteză închisă
  - Când variabila ajunge la zero, se sare la YYINITIAL
  - Variabila globală practic modelează un număr infint de stări!

Concluzii

- Analiza lexicală, realizată folosind expresii regulate și automate finite
- Expresiile regulate: modalitate concisă de specificarea a atomilor lexicali
- Conversia RE în AFN, apoi în AFD care esti în final minimizat
- AFD-ul minimizat este folosit pentru a recunoaște atomii din fluxul de intrare
- Procesul este automatizat folosind generatori de analizori lexicali
  - Se scriu descrierile atomilor ca expresii regulate
  - Se obține automat un program – analizor lexical – care identifică atomii dintr-un flux de caractere primit ca intrare

Cursul viitor:

- Analiza sintactică

Întrebări?