**Gramatici generative**

**Enunț**

Să se implementeze o gramatică generativă astfel: se citesc din fișier elementele componente VN, VT, simbolul de start și producțiile. Se citește de la tastatură un număr n, iar apoi se generează n cuvinte în gramatică. Generarea are loc în modul următor: se pornește de fiecare dată de la simbolul de start. Cât timp există producții aplicabile se aplică random una dintre producțiile aplicabile.

**Barem**

1. Se cere creearea unei clase Gramatica (alta decât clasa principală - aviz celor care programează Java sau C#!). În funcția principală main se declară un obiect de tip Gramatica.

Membrii clasei vor fi: VN, VT, S, producțiie

Metodele clasei:

(1) citire() - cititrea elementelor gramaticii

(2) verificare() - verificările descrise la punctul 3

(3) afisare( ) - afișarea frumoasă a elementelor gramaticii

(3) generare(optiune) - generarea de cuvinte pornind de la simbolul de start. Dacă opțiunea este 0 se afișază doar cuvântul obținut, daca opțiunea este 1 se afișază și pașii intermediari.

Construcția clasei - **1p (**metoda de generare este punctată separat la punctul 4)

2. Se cere citire din fișier a elementelor gramaticii: VN, VT, S, P (obligatoriu)

3. Se cere după citire verificarea corectitudinii gramaticii, conform definiției unei gramatici generative, adică

(1) VN ∩VT = ∅

(2) S ∈ VN

(3) pentru fiecare regulă membrul stâng conține cel puțin un neterminal

(4) există cel puțin o producție care are în stânga doar S

(5) fiecare producție conține doar elemente din VN și VT

- **2p**.

4. La generarea unui cuvânt în gramatică producția care se aplică la momentul curent trebuie aleasă random dintre producțiile aplicabile la momentul curent - **5p**

5. Atunci, când nu mai sunt aplicabile producții, se verifică faptului că avem un cuvânt format doar din terminale - **1p**

**Un punct din oficiu**

Pentru un program funcțional care conține doar citirea gramaticii și afișarea elementelor gramaticii -se dă nota 3.

**Sugestii de implementare:**

* Se pot citi din fișier VN, VT, S, și P astfel: se pun pe prima linie elementele din VN, pe a doua elementele din VT și pe a treia simbolul. Apoi se poate pune numărul de producții și pentru fiecare producție stanga spațiu dreapta.
* Pentru afișare: o producție poate fi afișată astfel S->abS
* VN și VT pot fi considerați vectori de caractere, S poate fi considerat de tip caracter
* Pentru producții se pot folosi doi vectori de string-uri - unul de părți stângi și altul de părți drepte sau o clasă / structură - producție
* pentru lambda se poate utiliza un caracter rezervat
* pentru punctul 4 - generarea unui cuvânt aplicând în mod aleator reguli se dă un exemplu mai jos

**Exemplu:**

Consider gramatica cu producțiile

1. S->abS

2. S->aSAb

3. S->aA

4. A->aAb

5. A->a

Pornim generearea unui cuvânt de la S.

În acest moment sunt aplicabile producțiile 1, 2, și 3. Se alege random una dintre ele și se aplică, de ex. dacă funcția random alege pe a doua, se obține:

S=>aSAb.

În acest moment sunt aplicabile toate producțiile. Se alege random una dintre ele, de exemplu ultima (5): S=>aSAb=>aSab.

În același mod putem obține: S=>aSAb=>aSab=>aAab=>aaab.