5. Ցուցակների կիրառություններ

5.1. Տողերի մշակում

Տողը սահմանենք որպես սիմվոլների հաջորդականություն: Տողը ներկայացնենք սիմվոլներին համապատասխանող ատոմների ցուցակի տեսքով՝ սահմանելով սիմվոլների և ատոմների համապատասխանությունը հետևյալ կերպ.

- լատինական փոքրատառին կամ տասական թվանշանին համապատասխանող ատոմն է նույն այդ սիմվոլը,
- լատինական փոքրատառից և տասական թվանշանից տարբերվող սիմվոլին համապատասխանող ատոմն է ապաթարցերի մեջ վերցված սիմվոլը:

Օրինակ՝ եթե բացատանիշը նշանակենք ⊔-ով, ապա **⊔a(1)⊔⊔beta⊔** տողին համապատասխանում է

[’⊔’, a, ’(’, 1, ’)’, ’⊔’, ’⊔’, b, e, t, a, ’⊔’, ’;’]

ցուցակը:

5.1.1. Եռատեսակ փակագծերի հաշվեկշռի ստուգում

Կասենք, որ տողում առկա է եռատեսակ (դիցուք՝ *կլոր*, *քառակուսի* և *ձևավոր*) փակագծերի հաշվեկշիռ, եթե.

- տողում չկան նշված տեսակի փակագծեր,
- տողը ստացվում է՝ եռատեսակ փակագծերի հաշվեկշիռ ունեցող տողից վերցնելով այն այս կամ այն տեսակի փակագծերի մեջ,
- տողը ստացվում է եռատեսակ փակագծերի հաշվեկշիռ ունեցող երկու տողերից դրանք միմյանց կցագրելու արդյունքում:

Եռատեսակ փակագծերի հաշվեկշռի ստուգման խնդիրը լուծելու համար ներմուծենք պահունակ, որտեղ կհիշենք տողը մշակելու ընթացքում հանդիպած բացող փակագծերը: *Տող* և *պահունակ* զույգի համար սահմանենք փակագծերի հաշվեկշռի առկայություն հետևյալ կերպ.

- դատարկ տող և դատարկ պահունակ զույգի համար առկա է փակագծերի հաշվեկշիռ,
- բացող փակագծով սկսվող տող և պահունակ զույգի համար առկա է փակագծերի հաշվեկշիռ, եթե այն առկա է տողի պոչ և ընդլայնված պահունակ զույգի համար, որտեղ ընդլայնված պահունակը ստացվում է տողի առջևում գրառված բացող փակագիծը պահունակին ավելացնելու արդյունքում,

- փակող փակագծով սկսվող տող և ոչ դատարկ պահունակ զույգի համար առկա է փակագծերի հաշվեկշիռ, եթե.
  - պահունակի գագաթային սիմվոլն է տողի առջևում գրառված փակող փակագծին համապատասխանող բացող փակագիծը,
  - առկա է փակագծերի հաշվեկշիռ տողի պոչ և կրճատված պահունակ զույգի համար, որտեղ կրճատված պահունակը ստացվում է գագաթային սիմվոլը պահունակից հեռացնելու արդյունքում,
- փակագծից տարբերվող սիմվոլով սկսվող տող և պահունակ զույգի համար առկա է փակագծերի հաշվեկշիռ, եթե այն առկա է տողի պոչ և սկզբնական պահունակ զույգի համար:

Նշանակենք **checkBrackets(S)** պնդումը, ըստ որի՝ **S** տողում առկա է եռատեսակ փակագծերի հաշվեկշիռ: **checkBrackets** պրեդիկատը ներկայացնենք հետևյալ կերպ.

```
checkBrackets(S):- checkBrackets1(S, []).
```

Ըստ **checkBrackets1(S, Stack)** պնդման **S** տող և **Stack** պահունակ զույգի համար առկա է փակագծերի հաշվեկշիռ: Հիմք ընդունելով փակագծերի հաշվեկշռի ստուգման վերը բերված ալգորիթմը՝ **checkBrackets1** պրեդիկատը սահմանենք հետևյալ կերպ.

```
checkBrackets1([], []):-                 !.

checkBrackets1(['('|S], Stack):-         checkBrackets1(S, ['('|Stack]), !.
checkBrackets1(['['|S], Stack):-         checkBrackets1(S, ['['|Stack]), !.
checkBrackets1(['{'|S], Stack):-         checkBrackets1(S, ['{'|Stack]), !.

checkBrackets1([')'|S], ['('|Stack]):- checkBrackets1(S, Stack), !.
checkBrackets1([']'|S], ['['|Stack]):- checkBrackets1(S, Stack), !.
checkBrackets1(['}'|S], ['{'|Stack]):- checkBrackets1(S, Stack), !.

checkBrackets1([')'|_], _):-             !, fail.
checkBrackets1([']'|_], _):-             !, fail.
checkBrackets1(['}'|_], _):-             !, fail.

checkBrackets1([_|S], Stack):-   checkBrackets1(S, Stack).
```

Վարժություններ

1. Որոշել, կա՞ արդյոք տրված տողում *կլոր* փակագծերի հաշվեկշիռ՝ օգտագործելով բացող և փակող կլոր փակագծերի քանակների տարբերությունը:
2. Որոշել, կա՞ արդյոք տրված տողում ցանկացած քանակությամբ տեսակների փակագծերի հաշվեկշիռ՝ օգտագործելով համապատասխան *բացող* և *փակող* փակագծերի զույգերից կազմված ցուցակ:

5.2. Նոսր բազմանդամների մշակում

x փոփոխականից կախված բազմանդամն անվանենք ***նոսր***, եթե դրա ոչ զրոյական միանդամների թիվը փոքր է բազմանդամի աստիճանի համեմատ: Մենք կենթադրենք, որ այս բաժնում դիտարկվող բազմանդամները նոսր են՝ չնշելով «նոսր» բառը:

Ընտրենք բազմանդամների հետևյալ ներկայացումը:

Ենթադրենք, որ **C** իրական գործակցով և **K** աստիճանի ցուցչով **x** փոփոխականից կախված միանդամը ներկայացված է **x(C, K)** կառուցվածքի միջոցով: Բազմանդամը ներկայացնենք աստիճանների նվազման կարգով ընթացող իր ոչ զրոյական միանդամների ցուցակով: Օրինակ՝ $\mathbf{5.5x^{100} - 3.5x^{50} + 10.0}$ բազմանդամի ներկայացումն է.

[x(5.5, 100), x(-3.5, 50), x(10.0, 0)]:

Նշենք, որ **0** բազմանդամը կներկայացվի դատարկ ցուցակով:

Ստորև դիտարկենք բազմանդամների նկատմամբ կիրառվող որոշ գործողությունների իրականացումը Prolog լեզվով:

5.2.1. Գումարում

Քանի որ բազմանդամները ոչ զրոյական միանդամների կարգավորված ցուցակներ են, ուստի բազմանդամների գումարման խնդիրը վերածվում է կարգավորված ցուցակների միաձուլման խնդրին այն տարբերությամբ, որ հավասար աստիճան ունեցող միանդամներ հանդիպելիս պատասխան ցուցակում անհրաժեշտ է ընդգրկել դրանց գումարը ներկայացնող միանդամը, եթե այն զրո չէ (այն է՝ միանդամների գործակիցները գումարվում են, ցուցիչը մնում է նույնը):

Նշանակենք **addition(P1, P2, P)** պնդումը, ըստ որի՝ **P1** և **P2** բազմանդամների գումարն է **P**-ն: **addittion** պրեդիկատը սահմանենք հետևյալ կերպ.

```
addition([], P, P):- !.
addition(P, [], P):- !.
addition([x(C1, K1)|P1], [x(C2, K2)|P2], [x(C1, K1)|P]):-
                                        K1 > K2, addition(P1, [x(C2, K2)|P2], P), !.
addition([x(C1, K1)|P1], [x(C2, K2)|P2], [x(C2, K2)|P]):-
                                        K2 > K1, addition([x(C1, K1)|P1], P2, P), !.
addition([x(C1, K)|P1], [x(C2, K)|P2], [x(C, K)|P]):-
                                        C is C1 + C2, C <> 0, addition(P1, P2, P), !.

addition([_|P1], [_|P2], P):-  addition(P1, P2, P).
```

5.2.2. Բազմապատկում

Դիտարկենք երկու բազմանդամների (դիցուք՝ **P1** և **P2**) *բազմապատկման* հետևյալ անդրադարձ ալգորիթմը.

- եթե **P1 = 0**, ապա բազմապատկման արդյունքն է **0**-ն,
- հակառակ դեպքում բազմապատկման արդյունքը սահմանվում է հետևյալ կերպ. եթե առաջին բազմանդամի ավագ միանդամն է **M**-ը, իսկ մնացած միանդամներից կազմված բազմանդամն է **Q**-ն, ապա բազմապատկման արդյունքն է **M*P2 + Q*P2** բազմանդամը:

Նշանակենք **multiplication(P1, P2, P)** պնդումը, ըստ որի՝ **P1** և **P2** բազմանդամների արտադրյալն է **P**-ն: **multiplication** պրեդիկատը սահմանենք հետևյալ կերպ.

```
multiplication([],_,[]):-  !.
multiplication([x(C1, K1)|P1],P2, P):-
                              multiplication1(x(C1, K1), P2, Q1),
                              multiplication(P1, P2, Q2),
                              addition(Q1, Q2, P).
```

Ըստ **multiplication1(M, P1, P)** պնդման՝ **P** բազմանդամը **M** միանդամի և **P1** բազմանդամի արտադրյալն է: **multiplication1** պրեդիկատի սահմանումն է.

```
multiplication1(_,[], []): -!.
multiplication1(x(C1, K1), [x(C2, K2)|P2], [x(C, K)|P]):-
                              C is C1*C2,
                              K is K1 + K2,
                              multiplication1(x(C1, K1), P2, P).
```

Վարժություններ

1. Կառուցել տրված բազմանդամի ածանցյալը;
2. Կառուցել տրված բազմանդամի արժեքը տրված կետում:

5.3. Գործողություններ մատրիցների հետ

A = {a_{ij}}, 1 ≤ i ≤ m, 1 ≤ j ≤ n, m×n չափի *մատրիցը* սահմանվում է որպես տարրերի աղյուսակ, որի տողերն ունեն **n**, իսկ սյունակները՝ **m** երկարություն.

$$A = \begin{pmatrix} a_{11} \cdots a_{1n} \\ \cdots \\ a_{m1} \cdots a_{mn} \end{pmatrix}$$

Պայմանավորվենք մատրիցի տողերը ներկայացնել տարրերի ցուցակների, իսկ մատրիցը՝ տողերի ցուցակի տեսքով.

[[a_{11}, … , a_{1n}], …, [a_{m1}, … , a_{mn}]]

Օրինակ՝

$$\begin{pmatrix} 2 & 3 & 4 \\ -1 & 0 & 5 \end{pmatrix}$$

2×3 չափի մատրիցի ներկայացումն է

[[2, 3, 4], [-1, 0, 5]]

ցուցակների ցուցակը:

Ստորև դիտարկենք մատրիցների մշակման մի քանի խնդիրներ:

5.3.1. Անկյունագծի որոշում

n×n չափի (քառակուսի) մատրիցի *գլխավոր անկյունագիծը* սահմանենք որպես

[a_{11}, … , a_{nn}]

ցուցակ:

Դիտարկենք քառակուսի մատրիցի անկյունագծի որոշման հետևյալ անդրադարձ ալգորիթմը.

- դատարկ մատրիցի անկյունագիծն է դատարկ ցուցակը,
- ոչ դատարկ մատրիցի անկյունագիծն է այն ցուցակը, որի գլուխն է մատրիցի առաջին տողի առաջին տարրը, պոչն է տրված մատրիցի առաջին տողը և առաջին սյունակը հեռացնելու արդյունքում ստացված քառակուսի մատրիցի անկյունագիծը:

Նշանակենք **diagonal(M, D)** պնդումը, ըստ որի՝ **D**-ն **M** քառակուսի մատրիցի անկյունագիծն է: **diagonal** պրեդիկատը սահմանենք հետևյալ կերպ.

```
diagonal([], []):-                    !.
diagonal([[X|_]|M], [X|D]):-  removeFirstColumn(M, M1), diagonal(M1, D).
```

Այստեղ **removeFirstColumn(M, M1)**-ը պնդումն է, ըստ որի՝ **M1** մատրիցը ստացվում է **M** մատրիցի առաջին սյունակը հեռացնելու արդյունքում: Այս գործողությունը նկարագրենք հետևյալ անդրադարձ եղանակով.

- դատարկ կամ մեկ սյունակ ունեցող մատրիցի առաջին սյունակը հեռացնելու արդյունքում ստացվում է դատարկ մատրից,
- առնվազն երկու սյունակ ունեցող մատրիցի առաջին սյունակը հեռացնելու արդյունքում ստացվում է մատրից (տողերի ցուցակ), որի գլուխն է տրված մատրիցի առաջին տողի պոչը, պոչն է տրված մատրիցից առաջին տողը և առաջին սյունակը հեռացնելու արդյունքում ստացված մատրիցը:

```
removeFirstColumn([], []):-                    !.
removeFirstColumn([[_]|_], []):-               !.
removeFirstColumn([[_|R]|M], [R|M1]):-   removeFirstColumn(M, M1).
```

removeFirstColumn պրեդիկատը սահմանենք հետևյալ կերպ.

5.3.2. Տրանսպոնացում

A = {a_{ij}}, 1 ≤ i ≤ m, 1 ≤ j ≤ n, m×n չափի մատրիցի ***տրանսպոնացումը*** սահմանվում է որպես **A′ = {a_{ji}}, 1 ≤ j ≤ n, 1 ≤ i ≤ m, n×m** չափի մատրից:

Դիտարկենք տրանսպոնացված մատրիցի կառուցման հետևյալ անդրադարձ ալգորիթմը.

- դատարկ մատրիցի տրանսպոնացումն է դատարկ մատրիցը,
- ոչ դատարկ մատրիցի տրանսպոնացումն է տրանսպոնացված առաջին տողի կցագրումը մատրիցի առաջին տողը հեռացնելուց հետո ստացված մատրիցի տրանսպոնացմանը:

Նշանակենք **transposition(M, M1)** պնդումը, ըստ որի՝ **M1** մատրիցը **M** մատրիցի տրանսպոնացումն է: **transposition** պրեդիկատը սահմանենք հետևյալ կերպ.

```
transposition([], []):-    !.
transposition([R|M], M1):-   createColumn(R, C),
                             transposition(M,M2),
                             addColumnToMatrix(C, M2, M1).
```

Այստեղ ըստ **createColumn(R, C)** պնդման, **C** սյունակը ստացվում է **R** տողի տրանսպոնացման արդյունքում, իսկ ըստ **addMatrixToColumn(C, M1, M)**-ը՝ պնդման **M** մատրիցը ստացվում է **C** սյունակը **M1** մատրիցին կցագրելու արդյունքում: Ստորև բերված են **createColumn** և **addMatrixToColumn** պրեդիկատների սահմանումները.

```
createColumn([], []):-                  !.
createColumn([X|R], [[X]|C]):-          createColumn(R, C).

addColumnToMatrix(C, [], C):-           !.
addColumnToMatrix([[X]|C], [R|M1], [[X|R]|M]):- addColumnToMatrix(C, M1, M).
```

Վարժություններ

1. Կառուցել *n*-չափանի *միավոր մատրիցը*, որի անկյունագծի տարրերը 1-են, իսկ մնացած տարրերը՝ 0:
2. Ստուգել ճի՞շտ է արդյոք, որ տրված մատրիցը *Յունգի աղյուսակ է*, այն է՝ յուրաքանչյուր տողի և յուրաքանչյուր սյունակի տարրերը ընթանում են չնվազման կարգով: