ДОМАШНА ЗАДАЧА 1

ЛОГИЧКО ПРОГРАМИРАЊЕ

РАБОТА СО ЛИСТИ ВО PROLOG

Задача 1. **(10 поени)** Да се напише предикат во PROLOG **`neparen_palindrom(L)`** кој ќе провери дали дадена непразна листа има непарен број на елементи и претставува палиндром. На пример:

```
?-neparen_palindrom([d,e,l,e,v,e,l,e,d]).
```
ќе одговори:

```
yes;
```
(има непарен број на елементи 9, и е палиндром)

Задача 2. **(10 поени)** Да се напише предикат во PROLOG **`naj_podniza(L1,N,L2)`** кој ќе ја најде поднизата L2 со должина N која се појавува најмногу пати во влезната листа од атомични елементи L1. На пример:

```
?- naj_podniza([1,2,2,3,2,2,4,2,2,3],1,L).
```
ќе одговори:

```
L=[2];
```

```
?- naj_podniza([1,2,2,3,2,2,4,2,2,3],2,L).
```
ќе одговори:

```
L=[2,2];
```

```
?- naj_podniza([1,2,2,3,2,2,4,2,2,3],3,L).
```
ќе одговори:

```
L=[2,2,3];
```

Задача 3. **(10 поени)** Да се напише предикат во PROLOG **`proveri(L)`** кој за дадена листа ќе го проверува следното:

- доколку листата има помалку од два елементи предикатот да враќа no.
- доколку има два елементи, ако вториот елемент е поголем од првиот да врати yes, во спротивен случај да врати no.
- доколку листата има повеќе од два елементи предикатот да врати yes ако елементите го исполнуваат следното правило: вториот елемент да биде поголем од првиот, третиот елемент да биде помал од вториот, четвртиот да биде поголем од третиот и така натаму до крајот на листата (наизменично да се менуваат поголемо па помало).

На пример:

?-proveri([]). ќе одговори: **no**

?-proveri([23]). ќе одговори: **no**

?-proveri([4,3]). ќе одговори: **no**

?-proveri([4,5]). ќе одговори: **yes**

?-proveri([3,9,4,8,6,7]). ќе одговори: **yes**

Задача 4. **(10 поени)** Да се напише предикат во PROLOG **permutacii(L1,L2)** кој ќе ги најде сите пермутации на елементите на листата L1 и ќе ги врати како подлисти на листата L2. На пример:

?- permutacii([a,b,c],L). ќе одговори:

L=[[a,b,c],[a,c,b],[b,a,c],[b,c,a],[c,a,b],[c,b,a]];

Напомена: Редоследот по кој ќе се вратат различните пермутации е небитен.

Задача 5. **(10 поени)** Да се напишат предикати во PROLOG со кои ќе се реализираат аритметичките операции на собирање, одземање, множење и делење на бинарни броеви. Еден бинарен број се претставува како листа од 0 и 1 и големината на бројот е неограничена. Притоа се работи само со позитивни броеви односно доколку резултатот од одземањето е негативен број истиот се заменува со 0. Делењето е целобројно (без остаток). Пример на предикатите што треба да бидат реализирани:

?-sobiranje([1,1,0],[1,1],L). **ќе одговори:**

L=[1,0,0,1]

?-odzemanje([1,1,0],[1,1],L). **ќе одговори:**

L=[1,1]

?-mnozenje([1,1,0],[1,1],L). **ќе одговори:**

L=[1,0,0,1,1]

?-delenje([1,1,0],[1,1],L). **ќе одговори:**

L=[1,0]

Задача 6. **(10 поени)** Да се напише предикат во PROLOG, **presmetaj(M,R)**, којшто за дадена квадратна матрица **M** како резултат враќа нова матрица **R** која се добива како **R=M*M^T** (M^T е транспонираната матрица, односно матрицата што се добива ако колоните станат редици, а редиците колони). Секоја матрица се проследува како листа при што секој ред се дефинира како посебна подлиста (видете го примерот).

На пример:

$$M=\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\end{bmatrix}, M^T=\begin{bmatrix}1 & 4 & 7\\ 2 & 5 & 8\\ 3 & 6 & 9\end{bmatrix},$$

$$R=M*M^T=\begin{bmatrix}1*1+2*2+3*3 & 1*4+2*5+3*6 & 1*7+2*8+3*9\\ 4*1+5*2+6*3 & 4*4+5*5+6*6 & 4*7+5*8+6*9\\ 7*1+8*2+9*3 & 7*4+8*5+9*6 & 7*7+8*8+9*9\end{bmatrix}=\begin{bmatrix}14 & 32 & 50\\ 32 & 77 & 122\\ 50 & 122 & 194\end{bmatrix}$$

?-presmetaj([[1,2,3],[4,5,6],[7,8,9]],R).

ќе одговори:

R=[[14,32,50],[32,77,122],[50,122,194]]

Напомена: Матриците можат да бидат со произволна големина. Не смеат да се користат било какви математички средувања на изразот што треба да се пресмета.

Задача 7. **(20 поени)** Да се напише предикат во PROLOG, **transform(L1,L2)**, кој дадена листа L1 составена од подлисти ќе ја трансформира во листа L2 во која подлистите од влезната листа се подредени според бројот на елементи по опаѓачки редослед. Може да се претпостави дека листата L1 не е празна. Доколку има две подлисти со ист број на елементи за "поголема" се смета онаа која има поголем прв елемент. Доколку и првите елементи им се еднакви се споредуваат вторите елементи итн. се додека не се најде барем

еден елемент во една од подлистите кој е поголем од елементот на соодветната позиција во втората подлиста. Доколку двете подлисти се идентични тогаш едната се отстранува од резултатот. Може да се претпостави дека нема повеќе од две подлисти со ист број елементи.

На пример:

```
?-
transform([[3,10],[2,4,6,33,1,8],[4,1,3,6],[3],[2,4,6,33,1,8],[
7,12],[4,1,2,7],[6,7,9]],L).
```

ќе одговори:

```
L=[[2,4,6,33,1,8],[4,1,3,6],[4,1,2,7],[6,7,9],[7,12],[3,10],[3]]
```

Задача 8. **(20 поени)** Да се напише предикат во PROLOG, **`brisi_sekoe_vtoro(L,R)`**, којшто од дадена листа **L** која содржи подлисти ќе го избрише секое второ појавување на некој елемент и резултатот ќе го смести во листа **R**. **Резултантната листа треба да ја задржи структурата на оригиналната листа**.

```
?-brisi_sekoe_vtoro([1,[2,1,[2],[3,3,[[3]],1,[1]],1],2,1,2,1,3],R).
```

ќе одговори:

```
R=[1,[2,[],[3,[[3]],1,[]],1],2,1]
```

ПРИКАЧУВАЊЕ И ПРОВЕРКА НА ВАШЕТО РЕШЕНИЕ НА ДОМАШНАТА ЗАДАЧА

Како решение за домашната задача треба да креирате две датотеки: `domasna1.pl` (со изворниот код за вашите предикати) и `domasna1-dokumentacija.pdf` (со документација за секој задача и тоа: кои предикати ги имате дефинирано за да ја решите задачата и кратко објаснување за суштината на секој предикат – што работи и како го постигнува тоа). Двете датотеки спакувајте ги во единствена архива која ќе ја именувате со вашиот број на индекс `XXXXXX.zip` (каде XXXXXX е вашиот број на индекс) и таквата архива прикачете ја на соодветниот линк на страницата на курсот за предметот.

Плагијаторство е најстрого забрането! Истото ќе се проверува за секоја задача посебно. Решенија на задачи за кои ќе се утврди дека се плагијати по автоматизам ќе бидат вреднувани со 0 поени (без оглед на тоа кој е оригиналниот автор на решението). Дополнително секоја задача плагијат ќе ви го преполови вкупниот број на освоени поени (на пример: Од вашите оригинални решенија освоите 60 поени и имате една задача плагијат крајните поени кои ќе ги добиете ќе бидат 30, ако имате две задачи плагијат крајните поени кои ќе ги добиете ќе бидат 15, итн). Задачите ќе се вреднуваат според прогресот кон целосно решение (што значи не мора да ја решите целосно задачата за да добиете поени за истата). Задачите за кои нема документација воопшто нема да се прегледуваат!