Виктор Василев 211504

Програмски парадигми
Домашна задача 1

Под коментарот shared utils имаме предикати кои се среќаваат во повеќе задачи и ги реискористуваме повеќе пати. Тоа се предикатот my_concat/3 за спојување на две листи, br_el/2 за наоѓање на број на елементи на листа, како и is_even/1 и is_odd/1 за прверување дали даден број е парен или непарен.

Задача 1

Дефинираме предикат neparen_broj_el/1 што проверува дали листата има непарен број на елементи. Предикатот zemi_posleden/2 го зима последниот елемент во низата. trgni_posleden/2 го брише последниот елемент во листата и ја дава новата листа без тој елемент. palindrom/1 прави проверка на листата дали нејзините елементи создаваат палиндром. Предикатот neparen_palindrom/1 е решението на задачата.

Задача 2

Предикатот prefiks/2 проверува дали листата (втор параметар) е префикс на првата листа (прв параметар). kolku_pati_podniza/3 ни помага да дознаеме колку пати втората листа се појавува како подлиста на првата листа. Предикатот zemi_prvi_n_el/3 ги зема првите N елементи од листата како нова листа. koja_podniza_poveke_ja_ima/4 проверува дали втората или третата листа се појавуваат почесто во листата - прв параметар, и во Resenie ја става таа што повеќе се појавува. naj_podniza/3 е финалниот предикат кој ни го дава решението на задачата.

Задача 3

Предикатот dva_ili_poveke_el(L1) проверува дали листата има повеќе од 1 (два или повеќе) елементи. naizmenicnost/2 го проверува својството за наизменичност што се бара во задачата. На пр за листата [1, 5, 2, 4] важи 5 > 1, 2 < 5, 4 > 2 - значи, го исполнува условот бидејќи вториот елемент е поголем од првиот, третиот е помал од вториот, четвртиот е поголем од третиот итн… proveri/1 е финалниот предикат и ги спојува двете проверки со што го имаме решението на задачата.

Задача 4

Дефинираме предикат permutacii_inner/2 рекурзивно ги поместува елементите на лситата на различни позиции како што е Heap алгоритмот за барање на пермутации. За тоа му помага my_insert/3 што вметнува елемент X во сите можни позиции на листата. permutacii/2 е финалниот предикат.

Задача 5

Дефинираме soberi_basic/3 што ги собира сите цифри како во обичен декаден броен систем. Пример:

```
   101101
+  110101
   ----------
   211202
```

Предикатот index_na_posledna_2_ili_3 ни ја враќа позицијата на последната 2ка или 3ка во така добиениот збир. Во нашиот пример тоа би било индекс 5 - последната цифра. reshi_2_ili_3/2 дава во што би се претворила моменталната цифра по решавање во зависност дали е 2 или 3. reshi_na_indeks/4 ја средува цифрата на даден индекс (во зависност од дали е 2 или 3 со помоп на претходниот предикат), и ја префрла (додава) единицата на цифрата кон лево. Цифрата кон лево се собира со таа единица исто како во декаден систем - би добиле 2 или 3, што понатаму во рекурзијата би било решено. reshi_result/2 го решава целиот број (листа) кој го добивме како резултат на soberi_basic/3. sobiranje/3 е финалното собирање на два бинарни бореви.

Задача 6

Дефинираме предикат extract_column/3 за вадење на единечна колона од матрицата која ќе се претвори во ред, за потоа да може да се повика transpose/2 рекурзивно. matrix_rows/3 дозволува множење на редици на едната матрица со колоните на другата со што го добиваме matrix_multiply/3 за множење на две цели матрици. За множење ред со колона ни треба dot_product/3 што наоѓа dot product на две листи. presmetaj/2 е финалниот предикат кој ја превртува матрицата и ја множи со старата.

Задача 7

Предикатот najdi_prednost_koga_ista_golemina го користиме за да провериме која од двете листи со иста големина треба да влезе прва во решението на задачата. Таа ги проверува елемент по елемент додека кај едната од нив не најде различни елементи. Таа што го содржи поголемиот од тие 2 различни елементи влегува прва во решението. izbrishi_podlista/3 брише подлиста од parent листата. Тоа го користеме за кога сме внеле подлиста во решението и треба истата да ја тргнеме од почетната листа, за да можеме да ја најдеме следната најголема подлиста со najdi_najgolema_podlista/3. transform/2 е финалниот предикат за користење кој ја решава задачата.