**Лабораторна робота №3**

**Списки**

**Цілі роботи:**

* ознайомлення з основами мові Prolog;
* вивчення правил уніфікації змінних;
* одержання практичних навичок складання найпростіших правил на

мові Prolog.

* ознайомлення з механизмами управління процесом пошуку рішень

у мові Prolog;

**Завдання**

**Цілі роботи:**

* ознайомлення з складними структурами даних у мові Prolog
* одержання практичних навичок роботи зі списками та структурами при написанні правил на мові Prolog

**Завдання:**

Створити базу даних програми з фактів, які описують результати сесії студентів у вигляді:

student(Group, Fam, [D1/M1, D2/M2, D3/M3, D4/M4]).

де: D – назва дисципліни, M – відповідна оцінка, Group – навчальна група, Fam – прізвище студента. Отримати список об’єктів даних відповідно завдання та вивести його у табличній формі.



Короткі теоретичні відомості

***Визначення списку у мові Prolog***

Список – це послідовність з будь-якої кількості елементів. Кожний елемент може бути будь-яким термом, припустимим у Prolog, в т.ч. списком.

Список має структуру бінарного дерева. Для позначки списку використовується функтор «./2» (крапка) з двома аргументами:

.(Head, Tail),

Де:

Head – голова списку, у якості якого може бути будь-який терм;

Tail – хвіст списку, у якості якого може бути тільки список.

Таким чином, послідовність з чотирьох констант a, b, c, d, що складають список, можна представити у вигляді структури:

.(a, . (b, . (c, . (d, [])))),

де [] – пустий список.

Для більш зручної позначки списку застосовується інша форма запису – у квадратних дужках перелічуються елементи через кому:

[a, b, c, d]

Якщо потрібно виділити хвіст списку, використовується символ «|» (вертикальна риса), який розділяє один або декілька елементів списку від хвоста:

|  |  |
| --- | --- |
| [a | T] | список містить голову і хвіст **T** з будь-якої кількості елементів |
| [a, b, c | T] | список містить три константи у початку і хвіст **T** з будь-якої кількості елементів |

***Обробка списків***

Структура списків дає можливість застосування рекурсії для обробки списків, тому що дії, які виконуються стосовно голови можна повторити для хвоста списку. Одне з правил повинна бути термінальною, тобто виконується конкретизація зміни заздалегідь передбаченим значенням. Альтернативні правила повинні мати у складі умовної частини рекурсивну ціль. Розглянемо простий приклад – отримання останнього елементу списку.

***Збереження рішень у списках***

Prolog система забезпечує отримання результатів рішень за допомогою механізму повернення. Однак кожне рішення необхідно використати одразу після отримання, інакше воно пропаде. Вбудовані предикати bagof, setof, findall дозволяють накопити рішення у списку для подальшої їх обробки.

Всі предикати мають формат:

F(X, P, L).

де:

F – один з вищеназваних функторів;

X – одна або декілька змін, що уособлюють ті об’єкти, які потрібно знайти;

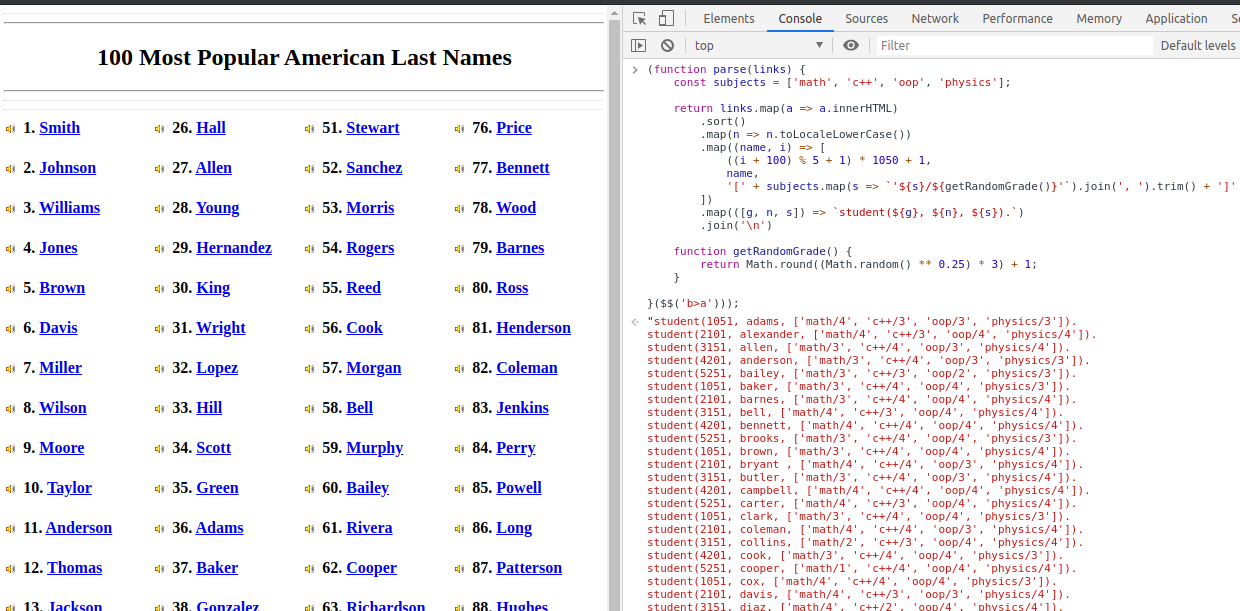
P – предикат, у залежності від якого виконується пошук;

L – список для накопичення результатів.

В таблиці 7.3 наведені приклади застосування предикатів для родинних відносин, зображених на рисунку 5.1 – необхідно отримати список всіх об’єктів бази даних, які є батьками (*parent*). Видно, що результати, які отримані предикатами *bagof* і *setof*, залежать від значення другого аргументу предиката *parent*, а предикат *findall* формує повний список об’єктів. Предикат *setof* у відміну від *bagof* упорядковує список.

**Рішення**

1. Формування бази даних програми



Текст програми

hasBadGrades(N) :- student(\_, N, G), containsBadGrades(G).

containsBadGrades([H|T]) :- wildcard\_match('\*/[1-2]', H); containsBadGrades(T).

printStudent(N, I) :- student(G, N, [M, C, O, P]),

   format("| ~a~t~5+ | ~a~t~5+ | ~a~t~16+ | ~a, ~a, ~a, ~a~t~41+ |~n", [I, G, N, M, C, O, P]).

printTHead :-

   printLine(),

   format("| ~a~t~5+ | ~a~t~5+ | ~a~t~16+ | ~a~t~40+ |~n", ['#', group, name, grades]),

   printLine().

printLine :- format("|~`-t~70||~n").

printList([S|T], N) :- printStudent(S, N), printLine(), K is N + 1, printList(T, K).

printFailedStudents :- printTHead(), setof(N, hasBadGrades(N), L), printList(L, 1).

student(1051, adams, ['math/3', 'c++/4', 'oop/4', 'physics/4']).

student(2101, alexander, ['math/3', 'c++/4', 'oop/3', 'physics/4']).

student(3151, allen, ['math/4', 'c++/3', 'oop/4', 'physics/4']).

student(4201, anderson, ['math/4', 'c++/4', 'oop/3', 'physics/3']).

student(5251, bailey, ['math/3', 'c++/3', 'oop/4', 'physics/4']).

student(1051, baker, ['math/3', 'c++/3', 'oop/4', 'physics/3']).

student(2101, barnes, ['math/3', 'c++/4', 'oop/3', 'physics/3']).

student(3151, bell, ['math/4', 'c++/4', 'oop/3', 'physics/4']).

student(4201, bennett, ['math/3', 'c++/3', 'oop/4', 'physics/3']).

student(5251, brooks, ['math/3', 'c++/3', 'oop/4', 'physics/4']).

student(1051, brown, ['math/4', 'c++/2', 'oop/3', 'physics/4']).

student(2101, bryant , ['math/3', 'c++/3', 'oop/3', 'physics/4']).

student(3151, butler, ['math/4', 'c++/4', 'oop/3', 'physics/4']).

student(4201, campbell, ['math/4', 'c++/4', 'oop/4', 'physics/4']).

student(5251, carter, ['math/4', 'c++/3', 'oop/4', 'physics/4']).

student(1051, clark, ['math/4', 'c++/4', 'oop/3', 'physics/4']).

student(2101, coleman, ['math/4', 'c++/3', 'oop/4', 'physics/4']).

student(3151, collins, ['math/4', 'c++/3', 'oop/4', 'physics/4']).

student(4201, cook, ['math/3', 'c++/4', 'oop/3', 'physics/4']).

student(5251, cooper, ['math/4', 'c++/4', 'oop/4', 'physics/4']).

student(1051, cox, ['math/3', 'c++/4', 'oop/4', 'physics/3']).

student(2101, davis, ['math/4', 'c++/4', 'oop/3', 'physics/3']).

student(3151, diaz, ['math/4', 'c++/4', 'oop/2', 'physics/4']).

student(4201, edwards, ['math/4', 'c++/4', 'oop/4', 'physics/3']).

student(5251, evans, ['math/4', 'c++/3', 'oop/4', 'physics/4']).

student(1051, flores, ['math/4', 'c++/4', 'oop/3', 'physics/4']).

student(2101, foster, ['math/3', 'c++/3', 'oop/3', 'physics/4']).

student(3151, garcia, ['math/3', 'c++/4', 'oop/4', 'physics/4']).

student(4201, gonzales, ['math/2', 'c++/3', 'oop/4', 'physics/3']).

student(5251, gonzalez, ['math/3', 'c++/4', 'oop/2', 'physics/4']).

student(1051, gray, ['math/3', 'c++/2', 'oop/3', 'physics/3']).

student(2101, green, ['math/3', 'c++/3', 'oop/4', 'physics/4']).

student(3151, griffin , ['math/4', 'c++/4', 'oop/4', 'physics/4']).

student(4201, hall, ['math/3', 'c++/3', 'oop/4', 'physics/4']).

student(5251, harris, ['math/4', 'c++/4', 'oop/4', 'physics/4']).

student(1051, hayes, ['math/3', 'c++/3', 'oop/4', 'physics/4']).

student(2101, henderson, ['math/4', 'c++/3', 'oop/3', 'physics/3']).

student(3151, hernandez, ['math/3', 'c++/3', 'oop/3', 'physics/4']).

student(4201, hill, ['math/3', 'c++/4', 'oop/4', 'physics/3']).

student(5251, howard, ['math/4', 'c++/4', 'oop/4', 'physics/4']).

student(1051, hughes, ['math/2', 'c++/1', 'oop/4', 'physics/3']).

student(2101, jackson, ['math/3', 'c++/4', 'oop/4', 'physics/4']).

student(3151, james, ['math/4', 'c++/3', 'oop/4', 'physics/3']).

student(4201, jenkins, ['math/3', 'c++/3', 'oop/4', 'physics/3']).

student(5251, johnson, ['math/4', 'c++/4', 'oop/3', 'physics/4']).

student(1051, jones, ['math/3', 'c++/4', 'oop/3', 'physics/4']).

student(2101, kelly, ['math/4', 'c++/2', 'oop/3', 'physics/4']).

student(3151, king, ['math/4', 'c++/3', 'oop/3', 'physics/4']).

student(4201, lee, ['math/3', 'c++/4', 'oop/3', 'physics/3']).

student(5251, lewis, ['math/4', 'c++/3', 'oop/3', 'physics/4']).

student(1051, long, ['math/4', 'c++/3', 'oop/3', 'physics/4']).

student(2101, lopez, ['math/1', 'c++/4', 'oop/3', 'physics/3']).

student(3151, martin, ['math/4', 'c++/3', 'oop/3', 'physics/2']).

student(4201, martinez, ['math/3', 'c++/4', 'oop/4', 'physics/4']).

student(5251, miller, ['math/3', 'c++/4', 'oop/4', 'physics/3']).

student(1051, mitchell, ['math/3', 'c++/4', 'oop/3', 'physics/4']).

student(2101, moore, ['math/4', 'c++/4', 'oop/3', 'physics/4']).

student(3151, morgan, ['math/4', 'c++/3', 'oop/4', 'physics/3']).

student(4201, morris, ['math/3', 'c++/3', 'oop/3', 'physics/4']).

student(5251, murphy, ['math/4', 'c++/3', 'oop/3', 'physics/3']).

student(1051, nelson, ['math/3', 'c++/3', 'oop/3', 'physics/3']).

student(2101, parker, ['math/3', 'c++/3', 'oop/3', 'physics/4']).

student(3151, patterson, ['math/4', 'c++/3', 'oop/4', 'physics/2']).

student(4201, perez, ['math/4', 'c++/4', 'oop/3', 'physics/4']).

student(5251, perry, ['math/2', 'c++/4', 'oop/3', 'physics/4']).

student(1051, peterson, ['math/3', 'c++/4', 'oop/3', 'physics/3']).

student(2101, phillips, ['math/4', 'c++/4', 'oop/3', 'physics/4']).

student(3151, powell, ['math/3', 'c++/3', 'oop/4', 'physics/3']).

student(4201, price, ['math/4', 'c++/4', 'oop/4', 'physics/2']).

student(5251, ramirez, ['math/4', 'c++/3', 'oop/3', 'physics/3']).

student(1051, reed, ['math/2', 'c++/4', 'oop/4', 'physics/4']).

student(2101, richardson, ['math/4', 'c++/4', 'oop/3', 'physics/3']).

student(3151, rivera, ['math/4', 'c++/4', 'oop/4', 'physics/4']).

student(4201, roberts, ['math/4', 'c++/4', 'oop/4', 'physics/3']).

student(5251, robinson, ['math/2', 'c++/4', 'oop/4', 'physics/4']).

student(1051, rodriguez, ['math/3', 'c++/4', 'oop/3', 'physics/3']).

student(2101, rogers, ['math/3', 'c++/4', 'oop/4', 'physics/4']).

student(3151, ross, ['math/4', 'c++/4', 'oop/4', 'physics/4']).

student(4201, russell, ['math/4', 'c++/3', 'oop/3', 'physics/4']).

student(5251, sanchez, ['math/3', 'c++/2', 'oop/4', 'physics/3']).

student(1051, sanders, ['math/4', 'c++/4', 'oop/3', 'physics/4']).

student(2101, scott, ['math/3', 'c++/4', 'oop/4', 'physics/4']).

student(3151, simmons, ['math/4', 'c++/4', 'oop/4', 'physics/4']).

student(4201, smith, ['math/4', 'c++/3', 'oop/4', 'physics/4']).

student(5251, stewart, ['math/4', 'c++/3', 'oop/3', 'physics/3']).

student(1051, taylor, ['math/2', 'c++/4', 'oop/3', 'physics/4']).

student(2101, thomas, ['math/4', 'c++/4', 'oop/4', 'physics/3']).

student(3151, thompson, ['math/3', 'c++/4', 'oop/3', 'physics/4']).

student(4201, torres, ['math/4', 'c++/3', 'oop/3', 'physics/4']).

student(5251, turner, ['math/4', 'c++/3', 'oop/4', 'physics/4']).

student(1051, walker, ['math/3', 'c++/4', 'oop/4', 'physics/4']).

student(2101, ward, ['math/4', 'c++/3', 'oop/4', 'physics/4']).

student(3151, washington, ['math/4', 'c++/4', 'oop/4', 'physics/4']).

student(4201, watson, ['math/4', 'c++/4', 'oop/4', 'physics/2']).

student(5251, white, ['math/3', 'c++/3', 'oop/3', 'physics/3']).

student(1051, williams, ['math/3', 'c++/4', 'oop/4', 'physics/4']).

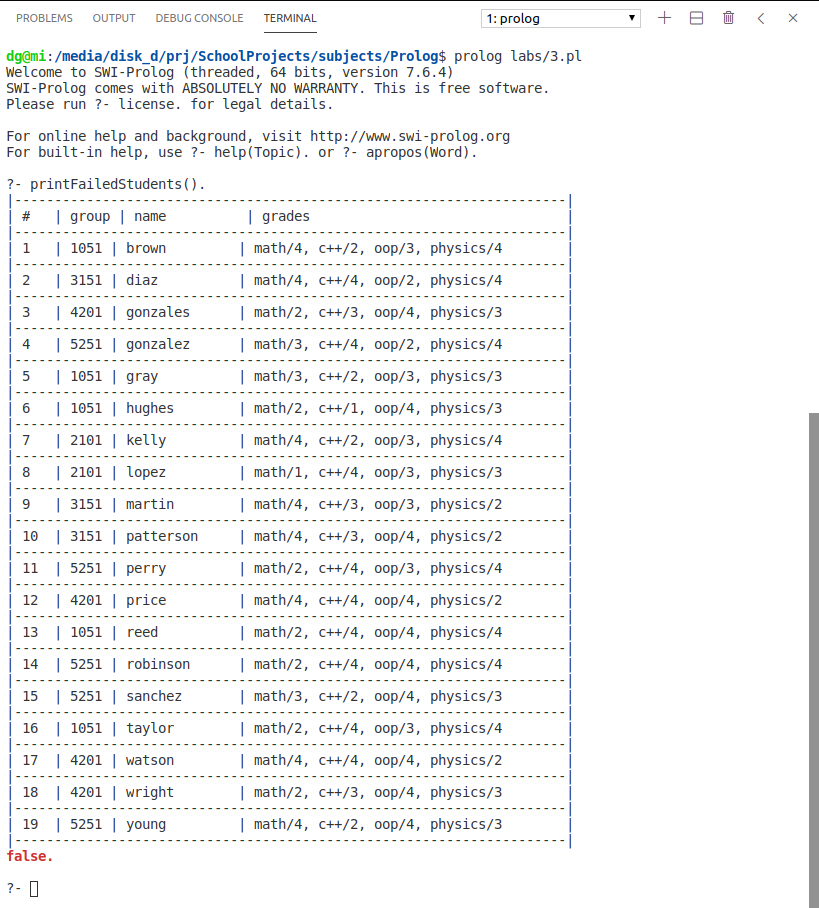
student(2101, wilson, ['math/3', 'c++/4', 'oop/4', 'physics/4']).

student(3151, wood, ['math/3', 'c++/3', 'oop/4', 'physics/3']).

student(4201, wright, ['math/2', 'c++/3', 'oop/4', 'physics/3']).

student(5251, young, ['math/4', 'c++/2', 'oop/4', 'physics/3']).

Результат виконання програми



**Висновок:** на цій лабораторній роботі я отримав навички програмування на мові F# використовуючи принципи функціонального програмування; засвоїв методи рішення складних задач на основі декомпозиції.