5. Управління пошуком з поверненням: предикати fail і відсікання.

fail - це тотожно-помилковий предикат, який штучно створює ситуацію неуспіху. Після виконання цього предиката управління передається в точку відкату і пошук триває. Використання предиката *fail* дозволяє знайти всі розвязання завдання.

Щоб обмежити простір пошуку і перервати пошук розвязань при виконанні будь-якої умови, використовується предикат *відсікання* (позначається **!**). Одного разу пройшовши через відсікання, неможливо повернутися назад, тому цей предикат є тотожно-істинним. Процес може тільки перейти до наступної підцілі, якщо така є.

Наприклад, p: - p1, p2,!, P3.

Якщо досягнуті цілі p1 і p2, то повернення до них для пошуку нових рішень неможливе.

Приклад 1

База даних містить факти виду: **student (ім'я, курс)**. Створити проект, що дозволяє сформувати список студентів 1-го курсу.

Розв'язання:

```
class predicates
student : (імя, номер) nondeterm anyflow.
spysok : (імя, номер) nondeterm anyflow.
clauses
student ("Володя", 3).
student ("Олена", 1).
tudent ("Ірина", 2).
student ("Марина", 1).
spysok(X,Y):-student(X,Y).
```

Результат виконання програми:

Список студентів 1-курсу
Олена
Дмитро
Марина

Приклад 2

База даних містить факти виду **father (**імя**,** імя**)**. Створити проект, що дозволяє визначити хто чий батько.

Розв'язання:

```
domains
імя = string.
class predicates
father : (імя, імя) nondeterm anyflow.
```

```
clauses
father ("Павло", "Петро").
father ("Петро", "Михайло").
father ("Петро", "Іван").
```

Результат виконання програми:

Павло - це батько Петра
Петро - це батько Михайла
Петро - це батько Івана

Приклад 3

Створити проект, який реалізує залізничний довідник. У довіднику міститься наступна інформація про кожний поїзд: номер поїзда, пункт призначення і час відправлення.

а) вивести всю інформацію з довідника.

Розв'язання:

```
class predicates
poezd : (a,p,t) nondeterm anyflow.
fff : (a,p,t) nondeterm anyflow.
clauses
poezd(233,"Київ","12-30").
poezd(257,"Львів","09-20").
poezd(267,"Дніпропет","20-40").
poezd(358,"Франківськ","02-30").
poezd(413,"Харків","11-10").
poezd(283,"Львів","14-50").
poezd(256,"Самбір","21-30").
fff(X,Y,Z):-poezd (X,Y,Z).
```

Результат виконання програми:

Розклад поїздів

Номер	Пункт прибуття	Час відправлення
233	Київ	12-30
257	Львів	09-20
267	Дніпропет	20-40
358	Франківськ	02-30
413	Харків	11-10
283	Львів	14-50
256	Самбір	21-30

б) організувати пошук поїзда по пункту призначення.

```
class predicates
поїзд : (a,p,t) nondeterm anyflow.
clauses
поїзд(233,"Київ","12-30").
поїзд(257,"Львів","09-20").
поїзд(267,"Дніпропет","20-40").
поїзд(358,"Франківськ","02-30").
поїзд(413,"Харків","11-10").
поїзд(283,"Львів","14-50").
поїзд(256,"Самбір","21-30").
```

Результат виконання програми:

```
Розклад поїздів
Пункт призначення: Львів
Номер Час відправлення
257             09-20
Розклад поїздів
Пункт призначення: Львів
Номер Час відправлення
283             14-50
```