**The laboratory work 3**

Control search with a return: predicates **fail** and clipping.

**fail** is an identically false predicate that artificially creates a failure situation. After this predicate is executed, the control is passed to the rollback point and the search continues. Using the fail predicate allows you to find all the solutions to the problem.

To limit the search space and interrupt the search for solutions when any condition is met, the clipping predicate is used (it is denoted as !) Once passing through the clipping, it is impossible to go back, because this predicate is identically true. The process can only go to the next sub-goal, if one exists.

**An example 1**

The database contains facts of the following view **student (name, course)**. Create a project that will allow us to find students of the 1st course.

**A solution**

PREDICATES

student(symbol,integer).

spisok.

CLAUSES

student(“vova”,3).

student(“lena”,1).

student(“dima”,1).

student(“ira”,2).

student(“marina”,1).

spisok:-student(X,1),write(X),nl,

fail.

GOAL

write("The list of the students of the first course is "),nl,spisok.

**The result of the program’s work**

**A list of students of the 1st course**

lena, dima, marina

**An example 2**

The database contains the following facts **father(name, name)**. Create a project that allows you to determine who are fathers of others.

**A solution**

DOMAINS

name=symbol.

PREDICATES

father (name, name).

everybody.

CLAUSES

father ("Pavel", "Petr").

father ("Petr", "Michael").

father ("Petr", "Ivan").

everybody:- father (X, Y), write(X, " is the father of ",Y),nl,

fail.

GOAL

everybody.

**The result of executing the program**

**An example 3**

1. Create the project that realizes a railway directory. There is the following information about each train: a number of a train, a destination point and time of departure.

**The solution:**

DOMAINS

nom=integer

p, t=string

PREDICATES

train (nom, p, t)

CLAUSES

train (233, Moscow, "12-30").

train (257, Moscow, "22-40").

train (133, Armavir, "10-20").

train (353, Armavir, "20-40").

train (353, Adler, "02-30").

train (413, Adler, "11-10").

train (256, Petersburg, "21-30").

GOAL

write(" A schedule of trains"), nl,

write(" A number of a point of the arrival, time of the departure"),

nl, train (N, P, T), write(N," ",P," ",T),nl,fail.

2. create the information about trains that depart in the following time

**The solution**:

GOAL

write("Time of departure :"), nl,

write("from "), Readln(T1),

write("till "), Readln(T2), nl,

write("Number The departure point Time of the departure"),

nl, train(N,P,T),T>=T1,T<=T2,write(N," ",P," ", T),

nl, fail.

**An example 4**

There is the database that includes information about sportsmen: a name and a type of sports. Define possible pairs of one sportsman a tennis player with other tennis players.

**The solution**:

DOMAINS

name, typeOfSport=string

PREDICATES

plays (name, typeOfSport).

sportList.

CLAUSES

plays (“Sasha”, “tennis”).

plays (“Ann”, “volleyball”).

plays (“Oleg”, “football”).

plays (“Kolya”, “tennis”).

plays (“Sasha”, “football”).

plays (“Andrey”, “tennis”).

sportList:-plays (X, “tennis”), !, plays (Y, “tennis”), X<>Y, write (X, “-”, Y), nl, fail.

GOAL

write("Tennis pairs"),nl,

sportList.

**An example 5**

A student gets the following grades depending of the points got:

Grade A if 90 <=Z<=100

Grade B if 75<=Z<=89

Grade C if 60<=Z=<74

Grade D if 50<=Z<=59

Create a program that will define a grade depending of the input value of Z.

**The solution:**

To solve the problem, we formulate the rule grade, which defines the grade depending on the the number of points Z. The rule consists of several parts. The first two parts provide verification of invalid Z values with the output of the corresponding message. The remaining parts of the rule determine the qualification, depending on the value of Z.

PREDICATES

grade(integer z).

CLAUSES

grade(Z):-Z<0,!, write("The wrong input!").

grade(Z):-Z>100,!,write("The wrong input!").

grade(Z):-Z>=90,!.

grade(Z):-Z>=75,!.

grade(Z):-Z>=60,!.

grade(Z).

GOAL

write("Z="), readint(Z), grade(Z).

**Comments**: readint is a standard predicate of the input values.

**Assignments**

1. The database contains facts of the form: rest (name, city), ukraine (city), russia (city), woman (name), man (name).  
   a) list the women who have a vacation in Russia;  
   b) list the men who have a vacation in Ukraine.
2. The database contains facts of the form: book (author, title, publisher, year\_publishing), ukraine (city).  
   a) display the entire list of books;  
   b) display a list of books by the authors of Pushkin and Chekhov;  
   c) list the books published in the publishing house "Peter" not earlier than 2000.

3. Make a program that realizes an air guide. The directory contains the following information about each flight: flight number, destination, time of departure, days (daily, even, odd). Output:  
a) all information from the directory;  
b) information about aircraft flying to a given point on even days;  
c) information about aircraft flying daily no later than the specified time.

4. Make a program that implements the geographical directory. The directory contains the following information about each country: the name of the country, the name of the capital, the population, geographical location (Europe or Asia). Output:  
a) all information from the directory;  
b) information on countries whose population exceeds the given value;  
c) information about European countries whose population does not exceed a given value.

5. The database contains facts of the form: student (name, class) and interest (name, hobby). Create a program that outputs:  
a) a list of all students and their hobbies;  
b) selects one of the students of this class who like cinema, and finds a couple for him. Output all possible pairs.