Лабораторная работа 7

1. Загрузить среду программирования.

2. Выполнить задачи по варианту. Номер варианта равен номеру рабочего места.

3. Представить результат преподавателю.

Чтобы разобраться с различиями операторов =, ==, =:= и is, для каждого запроса сначала попытайтесь

понять, как должен ответить Пролог, а потом проверьте, правильны ли ваши предположения.

```
?- X = 1+2.

?- 3 = 1+2.

?- 2+1 = 1+2.

?- X == 1+2.

?- 3 == 1+2.

?- 2+1 == 1+2.

?- X =:= 1+2.

?- 3 =:= 1+2.

?- 2+1 =:= 1+2.

?- X is 1+2.

?- 3 is 1+2.

?- 2+1 is 1+2.
```

Кроме того, в некоторых заданиях пригодятся предикаты проверки типов. Их можно найти в разделе 4.5 (в новых версиях -- 4.6) помощи SWI-Prolog http://www.swi-prolog.org/pldoc/doc_for?object=section%282,%274.6%27,swi%28%27/doc/Manual/typetest.html%27%29%29:

var(Term) -- удаётся для свободных переменных

nonvar(Term) -- противоположный var

float(Term) -- удаётся для чисел с плавающей точкой

integer(Term) -- удаётся для целых чисел

number(Term) -- удаётся для любых чисел

atom(Term) -- удаётся для атомов

compound(Term) -- удаётся для составных термов

atomic(Term) -- удаётся, если Term -- атом или число (или значение встроенного типа "строка", который в нашем курсе не используется)

callable(Term) -- удаётся, если Term -- атом или составной терм

ground(Term) -- удаётся, если Term не содержит свободных переменных

Варианты:

1	1. N-ное треугольное число -- сумма чисел 1+2+3+...+N. Определите (рекурсивно, не используя формулу N*(N+1)/2) предикат triangle(N, NthTriangle), который выполняется, если NthTriangle -- N-ное треугольное число. ?- triangle(4, X). X = 10 2. Определите предикат dot(List1, List2, DotProduct), который выполняется, если длины списков Dot1 и Dot2 совпадают, а DotProduct -- их скалярное произведение. ?- dot([1,2,3], [4,5,6], Result). Result = 26 %% т.е. 1*4+2*5+3*6 3. Определите предикат polynom(Expr), который выполняется, если Expr -- многочлен в нормальной форме (т.е. слагаемые идут в порядке уменьшения степени и коэффициенты не равны 0). ?- polynom(2*x^3 + x). Yes 4. Определите предикат combination(List, K, Combination), который находит все комбинации по K элементов List как отдельные ответы. ?- combination([1,2,3], 2, C). C = [1,2] ; C = [1,3] ; C = [2,3] ; No
2	1. Определите предикат fib(N, NthFib), который выполняется, если NthFib -- N-ное число Фибоначчи (начиная с 1). ?- fib(3, X). X = 2 2. Определите предикат mean(List, Mean), который выполняется, если Mean -- арифметическое среднее списка List. ?- mean([1,2,3], Mean). Mean = 2.0 3. В числе операторов, определённых в Prolog, есть /\ (и), \/ (или), \+ (не) и -> (следует). Определите предикат dnf(Formula), который выполняется, если Formula

	-- формула в дизъюнктивной нормальной форме (т.е. дизъюнкция ("или") конъюнкций переменных и их отрицаний, в качестве переменных используем атомы). ?- dnf((x /\ (\+ y) /\ z) \/ ((\+ x) /\ z)). Yes ?- dnf((x \/ (\+ y)) /\ (z \/ (\+ x) \/ y)). No 4. Определите предикат eval_logic(Formula, Values), который выполняется, если Formula -- логическая формула, Values -- список термов вида true(V) и false(V), где V -- атомы, которые используются в Formula, и в результате подстановки этих значений формула становится истинной. ?- eval_logic(\+ (x -> (\+ y)), [true(x), false(y)]). No
3	1. Определите (рекурсивно, не используя ^) предикат power(X, N, Power), который выполняется, если Power -- число X в степени N. ?- power(3, 2, X). X = 9 2. Представим многочлены как списки их коэффициентов в порядке убывания степени. Например, многочлен 2*x^2+1 будет представлен как [2,0,1]. Определите предикат eval_poly(Polynom, X, Result), выполняющийся, если Result -- значение многочлена Polynom при аргументе X. Для возведения в степень используйте оператор ^ или определённый в прошлой задаче power/3. Возможно также использование схемы Горнера. ?- eval_poly([2,0,1,0], 1, Result). % 2*X^3 + X = 2*1^3 + 1 = 3 Result = 3 3. В числе операторов, определённых в Prolog, есть /\ (и), \/ (или), \+ (не) и -> (следует). Определите предикат nnf(Formula), который выполняется, если Formula -- формула, в которой отрицания стоят только перед переменными (переменные в формуле представляем атомами). ?- nnf((x /\ (\+ y)) -> (\+ x) /\ z). Yes ?- nnf((x /\ (\+ y)) -> \+ (x /\ z)). No 4. Определите предикат nnf(Formula, NNF), который выполняется, если NNF -- формула, полученная из Formula эквивалентными преобразованиями, в которой отрицания стоят только перед переменными. ?- nnf(\+ ((x -> y) /\ z), NNF). NNF = (x /\ (\+ y)) \/ (\+ z)
4	1. Определите предикат factorial(N, FactN), который выполняется, если FactN -- факториал N. ?- fact(3, X).

	X = 6 2. Определите предикат occurrences(Elem, List, Number), который выполняется, если элемент Elem встречается в списке List N раз. ?- occurrences(3, [1,2,3,1,3], N). N = 2 3. :-, ; и , -- обычные операторы в Prolog. Определите предикат rule(Rule), который проверяет, является ли терм Rule правилом (без точки в конце), то есть: 1. Голова правила может быть вызвана (в смысле callable/1); 2. Тело правила состоит из (1 или более) callable термов, разделённых ',' и ';' (заметьте, что T1,T2 и T1;T2 -- сами callable, но их нужно обрабатывать отдельно). ?- rule(p(X, Y) :- q(X)). Yes ?- rule(p :- q(X), r, r). Yes ?- rule(4 :- p, q). No ?- rule(p :- 4, q). No 4. Определите предикат eval_arithmetic(Expr, Values, Result), который выполняется, если Expr -- арифметическое выражение (содержащее операторы +, *, -, / и атомы в качестве переменных), Values -- список термов Var = Val, где Var -- атом, а Val -- число, а Result -- результат вычисления Expr при подстановке вместо каждого атома соответствующего ему значения из Values. ?- eval_arithmetic(x + 3*y, [x = 2, y = 1], Result). Result = 5
5	1. N-ное треугольное число -- сумма чисел 1+2+3+...+N. Определите (рекурсивно, не используя формулу N*(N+1)/2) предикат triangle(N, NthTriangle), который выполняется, если NthTriangle -- N-ное треугольное число. ?- triangle(4, X). X = 10 2. Определите предикат dot(List1, List2, DotProduct), который выполняется, если длины списков Dot1 и Dot2 совпадают, а DotProduct -- их скалярное произведение. ?- dot([1,2,3], [4,5,6], Result). Result = 26 %% т.е. 1*4+2*5+3*6 3. Определите предикат polynom(Expr), который выполняется, если Expr -- многочлен в нормальной форме (т.е. слагаемые идут в порядке уменьшения степени и коэффициенты не равны 0). ?- polynom(2*x^3 + x). Yes

	4. Определите предикат combination(List, K, Combination), который находит все комбинации по K элементов List как отдельные ответы. ?- combination([1,2,3], 2, C). C = [1,2] ; C = [1,3] ; C = [2,3] ; No
6	1. Определите предикат fib(N, NthFib), который выполняется, если NthFib -- N-ное число Фибоначчи (начиная с 1). ?- fib(3, X). X = 2 2. Определите предикат mean(List, Mean), который выполняется, если Mean -- арифметическое среднее списка List. ?- mean([1,2,3], Mean). Mean = 2.0 3. В числе операторов, определённых в Prolog, есть /\ (и), \/ (или), \+ (не) и -> (следует). Определите предикат dnf(Formula), который выполняется, если Formula -- формула в дизъюнктивной нормальной форме (т.е. дизъюнкция ("или") конъюнкций переменных и их отрицаний, в качестве переменных используем атомы). ?- dnf((x /\ (\+ y) /\ z) \/ ((\+ x) /\ z)). Yes ?- dnf((x \/ (\+ y)) /\ (z \/ (\+ x) \/ y)). No 4. Определите предикат eval_logic(Formula, Values), который выполняется, если Formula -- логическая формула, Values -- список термов вида true(V) и false(V), где V -- атомы, которые используются в Formula, и в результате подстановки этих значений формула становится истинной. ?- eval_logic(\+ (x -> (\+ y)), [true(x), false(y)]). No
7	1. Определите (рекурсивно, не используя ^) предикат power(X, N, Power), который выполняется, если Power -- число X в степени N. ?- power(3, 2, X). X = 9 2. Представим многочлены как списки их коэффициентов в порядке убывания степени. Например, многочлен 2*x^2+1 будет представлен как [2,0,1]. Определите предикат eval_poly(Polynom, X, Result), выполняющийся, если Result -- значение многочлена Polynom при аргументе X. Для возведения в степень используйте оператор ^ или определённый в прошлой задаче power/3. Возможно также использование схемы Горнера.

	?- eval_poly([2,0,1,0], 1, Result). % 2*X^3 + X = 2*1^3 + 1 = 3 Result = 3 3. В числе операторов, определённых в Prolog, есть /\ (и), \/ (или), \+ (не) и -> (следует). Определите предикат nnf(Formula), который выполняется, если Formula -- формула, в которой отрицания стоят только перед переменными (переменные в формуле представляем атомами). ?- nnf((x /\ (\+ y)) -> (\+ x) /\ z). Yes ?- nnf((x /\ (\+ y)) -> \+ (x /\ z)). No 4. Определите предикат nnf(Formula, NNF), который выполняется, если NNF -- формула, полученная из Formula эквивалентными преобразованиями, в которой отрицания стоят только перед переменными. ?- nnf(\+ ((x -> y) /\ z), NNF). NNF = (x /\ (\+ y)) \/ (\+ z)
8	1. Определите предикат factorial(N, FactN), который выполняется, если FactN -- факториал N. ?- fact(3, X). X = 6 2. Определите предикат occurrences(Elem, List, Number), который выполняется, если элемент Elem встречается в списке List N раз. ?- occurrences(3, [1,2,3,1,3], N). N = 2 3. :-, ; и , -- обычные операторы в Prolog. Определите предикат rule(Rule), который проверяет, является ли терм Rule правилом (без точки в конце), то есть: 1. Голова правила может быть вызвана (в смысле callable/1); 2. Тело правила состоит из (1 или более) callable термов, разделённых ',' и ';' (заметьте, что T1,T2 и T1;T2 -- сами callable, но их нужно обрабатывать отдельно). ?- rule(p(X, Y) :- q(X)). Yes ?- rule(p :- q(X), r, r). Yes ?- rule(4 :- p, q). No ?- rule(p :- 4, q). No 4. Определите предикат eval_arithmetic(Expr, Values, Result), который выполняется, если Expr -- арифметическое выражение (содержащее операторы +, *, -, / и атомы в качестве переменных), Values -- список термов Var = Val, где Var -- атом, а Val -- число, а Result -- результат вычисления Expr при подстановке вместо каждого атома соответствующего ему значения из Values.

	?- eval_arithmetic(x + 3*y, [x = 2, y = 1], Result). Result = 5
9	1. N-ное треугольное число -- сумма чисел 1+2+3+...+N. Определите (рекурсивно, не используя формулу N*(N+1)/2) предикат triangle(N, NthTriangle), который выполняется, если NthTriangle -- N-ное треугольное число. ?- triangle(4, X). X = 10 2. Определите предикат dot(List1, List2, DotProduct), который выполняется, если длины списков Dot1 и Dot2 совпадают, а DotProduct -- их скалярное произведение. ?- dot([1,2,3], [4,5,6], Result). Result = 26 %% т.е. 1*4+2*5+3*6 3. Определите предикат polynom(Expr), который выполняется, если Expr -- многочлен в нормальной форме (т.е. слагаемые идут в порядке уменьшения степени и коэффициенты не равны 0). ?- polynom(2*x^3 + x). Yes 4. Определите предикат combination(List, K, Combination), который находит все комбинации по K элементов List как отдельные ответы. ?- combination([1,2,3], 2, C). C = [1,2] ; C = [1,3] ; C = [2,3] ; No
10	1. Определите предикат fib(N, NthFib), который выполняется, если NthFib -- N-ное число Фибоначчи (начиная с 1). ?- fib(3, X). X = 2 2. Определите предикат mean(List, Mean), который выполняется, если Mean -- арифметическое среднее списка List. ?- mean([1,2,3], Mean). Mean = 2.0 3. В числе операторов, определённых в Prolog, есть /\ (и), \/ (или), \+ (не) и -> (следует). Определите предикат dnf(Formula), который выполняется, если Formula -- формула в дизъюнктивной нормальной форме (т.е. дизъюнкция ("или") конъюнкций переменных и их отрицаний, в качестве переменных используем атомы). ?- dnf((x /\ (\+ y) /\ z) \/ ((\+ x) /\ z)). Yes ?- dnf((x \/ (\+ y)) /\ (z \/ (\+ x) \/ y)).

	No 4. Определите предикат eval_logic(Formula, Values), который выполняется, если Formula -- логическая формула, Values -- список термов вида true(V) и false(V), где V -- атомы, которые используются в Formula, и в результате подстановки этих значений формула становится истинной. ?- eval_logic(\+ (x -> (\+ y)), [true(x), false(y)]). No
11	1. Определите (рекурсивно, не используя ^) предикат power(X, N, Power), который выполняется, если Power -- число X в степени N. ?- power(3, 2, X). X = 9 2. Представим многочлены как списки их коэффициентов в порядке убывания степени. Например, многочлен 2*x^2+1 будет представлен как [2,0,1]. Определите предикат eval_poly(Polynom, X, Result), выполняющийся, если Result -- значение многочлена Polynom при аргументе X. Для возведения в степень используйте оператор ^ или определённый в прошлой задаче power/3. Возможно также использование схемы Горнера. ?- eval_poly([2,0,1,0], 1, Result). % 2*X^3 + X = 2*1^3 + 1 = 3 Result = 3 3. В числе операторов, определённых в Prolog, есть /\ (и), \/ (или), \+ (не) и -> (следует). Определите предикат nnf(Formula), который выполняется, если Formula -- формула, в которой отрицания стоят только перед переменными (переменные в формуле представляем атомами). ?- nnf((x /\ (\+ y)) -> (\+ x) /\ z). Yes ?- nnf((x /\ (\+ y)) -> \+ (x /\ z)). No 4. Определите предикат nnf(Formula, NNF), который выполняется, если NNF -- формула, полученная из Formula эквивалентными преобразованиями, в которой отрицания стоят только перед переменными. ?- nnf(\+ ((x -> y) /\ z), NNF). NNF = (x /\ (\+ y)) \/ (\+ z)
12	1. Определите предикат factorial(N, FactN), который выполняется, если FactN -- факториал N. ?- fact(3, X). X = 6 2. Определите предикат occurrences(Elem, List, Number), который выполняется, если элемент Elem встречается в списке List N раз. ?- occurrences(3, [1,2,3,1,3], N). N = 2

	3. :-, ; и , -- обычные операторы в Prolog. Определите предикат rule(Rule), который проверяет, является ли терм Rule правилом (без точки в конце), то есть: 1. Голова правила может быть вызвана (в смысле callable/1); 2. Тело правила состоит из (1 или более) callable термов, разделённых ',' и ';' (заметьте, что T1,T2 и T1;T2 -- сами callable, но их нужно обрабатывать отдельно). ?- rule(p(X, Y) :- q(X)). Yes ?- rule(p :- q(X), r, r). Yes ?- rule(4 :- p, q). No ?- rule(p :- 4, q). No 4. Определите предикат eval_arithmetic(Expr, Values, Result), который выполняется, если Expr -- арифметическое выражение (содержащее операторы +, *, -, / и атомы в качестве переменных), Values -- список термов Var = Val, где Var -- атом, а Val -- число, а Result -- результат вычисления Expr при подстановке вместо каждого атома соответствующего ему значения из Values. ?- eval_arithmetic(x + 3*y, [x = 2, y = 1], Result). Result = 5
13	1. N-ное треугольное число -- сумма чисел 1+2+3+...+N. Определите (рекурсивно, не используя формулу N*(N+1)/2) предикат triangle(N, NthTriangle), который выполняется, если NthTriangle -- N-ное треугольное число. ?- triangle(4, X). X = 10 2. Определите предикат dot(List1, List2, DotProduct), который выполняется, если длины списков Dot1 и Dot2 совпадают, а DotProduct -- их скалярное произведение. ?- dot([1,2,3], [4,5,6], Result). Result = 26 %% т.е. 1*4+2*5+3*6 3. Определите предикат polynom(Expr), который выполняется, если Expr -- многочлен в нормальной форме (т.е. слагаемые идут в порядке уменьшения степени и коэффициенты не равны 0). ?- polynom(2*x^3 + x). Yes 4. Определите предикат combination(List, K, Combination), который находит все комбинации по K элементов List как отдельные ответы. ?- combination([1,2,3], 2, C). C = [1,2] ; C = [1,3] ; C = [2,3] ; No

14	1. Определите предикат fib(N, NthFib), который выполняется, если NthFib -- N-ное число Фибоначчи (начиная с 1). ?- fib(3, X). X = 2 2. Определите предикат mean(List, Mean), который выполняется, если Mean -- арифметическое среднее списка List. ?- mean([1,2,3], Mean). Mean = 2.0 3. В числе операторов, определённых в Prolog, есть /\ (и), \/ (или), \+ (не) и -> (следует). Определите предикат dnf(Formula), который выполняется, если Formula -- формула в дизъюнктивной нормальной форме (т.е. дизъюнкция ("или") конъюнкций переменных и их отрицаний, в качестве переменных используем атомы). ?- dnf((x /\ (\+ y) /\ z) \/ ((\+ x) /\ z)). Yes ?- dnf((x \/ (\+ y)) /\ (z \/ (\+ x) \/ y)). No 4. Определите предикат eval_logic(Formula, Values), который выполняется, если Formula -- логическая формула, Values -- список термов вида true(V) и false(V), где V -- атомы, которые используются в Formula, и в результате подстановки этих значений формула становится истинной. ?- eval_logic(\+ (x -> (\+ y)), [true(x), false(y)]). No
15	1. Определите (рекурсивно, не используя ^) предикат power(X, N, Power), который выполняется, если Power -- число X в степени N. ?- power(3, 2, X). X = 9 2. Представим многочлены как списки их коэффициентов в порядке убывания степени. Например, многочлен 2*x^2+1 будет представлен как [2,0,1]. Определите предикат eval_poly(Polynom, X, Result), выполняющийся, если Result -- значение многочлена Polynom при аргументе X. Для возведения в степень используйте оператор ^ или определённый в прошлой задаче power/3. Возможно также использование схемы Горнера. ?- eval_poly([2,0,1,0], 1, Result). % 2*X^3 + X = 2*1^3 + 1 = 3 Result = 3 3. В числе операторов, определённых в Prolog, есть /\ (и), \/ (или), \+ (не) и -> (следует). Определите предикат nnf(Formula), который выполняется, если Formula -- формула, в которой отрицания стоят только перед переменными (переменные в формуле представляем атомами).

	?- nnf((x /\ (\+ y)) -> (\+ x) /\ z). Yes ?- nnf((x /\ (\+ y)) -> \+ (x /\ z)). No 4. Определите предикат nnf(Formula, NNF), который выполняется, если NNF -- формула, полученная из Formula эквивалентными преобразованиями, в которой отрицания стоят только перед переменными. ?- nnf(\+ ((x -> y) /\ z), NNF). NNF = (x /\ (\+ y)) \/ (\+ z)
16	1. Определите предикат factorial(N, FactN), который выполняется, если FactN -- факториал N. ?- fact(3, X). X = 6 2. Определите предикат occurrences(Elem, List, Number), который выполняется, если элемент Elem встречается в списке List N раз. ?- occurrences(3, [1,2,3,1,3], N). N = 2 3. :-, ; и , -- обычные операторы в Prolog. Определите предикат rule(Rule), который проверяет, является ли терм Rule правилом (без точки в конце), то есть: 1. Голова правила может быть вызвана (в смысле callable/1); 2. Тело правила состоит из (1 или более) callable термов, разделённых ',' и ';' (заметьте, что T1,T2 и T1;T2 -- сами callable, но их нужно обрабатывать отдельно). ?- rule(p(X, Y) :- q(X)). Yes ?- rule(p :- q(X), r, r). Yes ?- rule(4 :- p, q). No ?- rule(p :- 4, q). No 4. Определите предикат eval_arithmetic(Expr, Values, Result), который выполняется, если Expr -- арифметическое выражение (содержащее операторы +, *, -, / и атомы в качестве переменных), Values -- список термов Var = Val, где Var -- атом, а Val -- число, а Result -- результат вычисления Expr при подстановке вместо каждого атома соответствующего ему значения из Values. ?- eval_arithmetic(x + 3*y, [x = 2, y = 1], Result). Result = 5
17	1. N-ное треугольное число -- сумма чисел 1+2+3+...+N. Определите (рекурсивно, не используя формулу N*(N+1)/2) предикат triangle(N, NthTriangle), который выполняется, если NthTriangle -- N-ное треугольное число. ?- triangle(4, X). X = 10

	2. Определите предикат dot(List1, List2, DotProduct), который выполняется, если длины списков Dot1 и Dot2 совпадают, а DotProduct -- их скалярное произведение. ?- dot([1,2,3], [4,5,6], Result). Result = 26 %% т.е. 1*4+2*5+3*6 3. Определите предикат polynom(Expr), который выполняется, если Expr -- многочлен в нормальной форме (т.е. слагаемые идут в порядке уменьшения степени и коэффициенты не равны 0). ?- polynom(2*x^3 + x). Yes 4. Определите предикат combination(List, K, Combination), который находит все комбинации по K элементов List как отдельные ответы. ?- combination([1,2,3], 2, C). C = [1,2] ; C = [1,3] ; C = [2,3] ; No
18	1. Определите предикат fib(N, NthFib), который выполняется, если NthFib -- N-ное число Фибоначчи (начиная с 1). ?- fib(3, X). X = 2 2. Определите предикат mean(List, Mean), который выполняется, если Mean -- арифметическое среднее списка List. ?- mean([1,2,3], Mean). Mean = 2.0 3. В числе операторов, определённых в Prolog, есть /\ (и), \/ (или), \+ (не) и -> (следует). Определите предикат dnf(Formula), который выполняется, если Formula -- формула в дизъюнктивной нормальной форме (т.е. дизъюнкция ("или") конъюнкций переменных и их отрицаний, в качестве переменных используем атомы). ?- dnf((x /\ (\+ y) /\ z) \/ ((\+ x) /\ z)). Yes ?- dnf((x \/ (\+ y)) /\ (z \/ (\+ x) \/ y)). No 4. Определите предикат eval_logic(Formula, Values), который выполняется, если Formula -- логическая формула, Values -- список термов вида true(V) и false(V), где V -- атомы, которые используются в Formula, и в результате подстановки этих значений формула становится истинной. ?- eval_logic(\+ (x -> (\+ y)), [true(x), false(y)]).

	No
19	1. Определите (рекурсивно, не используя ^) предикат power(X, N, Power), который выполняется, если Power -- число X в степени N. ?- power(3, 2, X). X = 9 2. Представим многочлены как списки их коэффициентов в порядке убывания степени. Например, многочлен 2*x^2+1 будет представлен как [2,0,1]. Определите предикат eval_poly(Polynom, X, Result), выполняющийся, если Result -- значение многочлена Polynom при аргументе X. Для возведения в степень используйте оператор ^ или определённый в прошлой задаче power/3. Возможно также использование схемы Горнера. ?- eval_poly([2,0,1,0], 1, Result). % 2*X^3 + X = 2*1^3 + 1 = 3 Result = 3 3. В числе операторов, определённых в Prolog, есть /\ (и), \/ (или), \+ (не) и -> (следует). Определите предикат nnf(Formula), который выполняется, если Formula -- формула, в которой отрицания стоят только перед переменными (переменные в формуле представляем атомами). ?- nnf((x /\ (\+ y)) -> (\+ x) /\ z). Yes ?- nnf((x /\ (\+ y)) -> \+ (x /\ z)). No 4. Определите предикат nnf(Formula, NNF), который выполняется, если NNF -- формула, полученная из Formula эквивалентными преобразованиями, в которой отрицания стоят только перед переменными. ?- nnf(\+ ((x -> y) /\ z), NNF). NNF = (x /\ (\+ y)) \/ (\+ z)
20	1. Определите предикат factorial(N, FactN), который выполняется, если FactN -- факториал N. ?- fact(3, X). X = 6 2. Определите предикат occurrences(Elem, List, Number), который выполняется, если элемент Elem встречается в списке List N раз. ?- occurrences(3, [1,2,3,1,3], N). N = 2 3. :-, ; и , -- обычные операторы в Prolog. Определите предикат rule(Rule), который проверяет, является ли терм Rule правилом (без точки в конце), то есть: 1. Голова правила может быть вызвана (в смысле callable/1); 2. Тело правила состоит из (1 или более) callable термов, разделённых ',' и ';' (заметьте, что T1,T2 и T1;T2 -- сами callable, но их нужно обрабатывать отдельно). ?- rule(p(X, Y) :- q(X)).

	Yes ?- rule(p :- q(X), r, r). Yes ?- rule(4 :- p, q). No ?- rule(p :- 4, q). No 4. Определите предикат eval_arithmetic(Expr, Values, Result), который выполняется, если Expr -- арифметическое выражение (содержащее операторы +, *, -, / и атомы в качестве переменных), Values -- список термов Var = Val, где Var -- атом, а Val -- число, а Result -- результат вычисления Expr при подстановке вместо каждого атома соответствующего ему значения из Values. ?- eval_arithmetic(x + 3*y, [x = 2, y = 1], Result). Result = 5
21	1. N-ное треугольное число -- сумма чисел 1+2+3+...+N. Определите (рекурсивно, не используя формулу N*(N+1)/2) предикат triangle(N, NthTriangle), который выполняется, если NthTriangle -- N-ное треугольное число. ?- triangle(4, X). X = 10 2. Определите предикат dot(List1, List2, DotProduct), который выполняется, если длины списков Dot1 и Dot2 совпадают, а DotProduct -- их скалярное произведение. ?- dot([1,2,3], [4,5,6], Result). Result = 26 %% т.е. 1*4+2*5+3*6 3. Определите предикат polynom(Expr), который выполняется, если Expr -- многочлен в нормальной форме (т.е. слагаемые идут в порядке уменьшения степени и коэффициенты не равны 0). ?- polynom(2*x^3 + x). Yes 4. Определите предикат combination(List, K, Combination), который находит все комбинации по K элементов List как отдельные ответы. ?- combination([1,2,3], 2, C). C = [1,2] ; C = [1,3] ; C = [2,3] ; No
22	1. Определите предикат fib(N, NthFib), который выполняется, если NthFib -- N-ное число Фибоначчи (начиная с 1). ?- fib(3, X). X = 2 2. Определите предикат mean(List, Mean), который выполняется, если Mean -- арифметическое среднее

	списка List. ?- mean([1,2,3], Mean). Mean = 2.0 3. В числе операторов, определённых в Prolog, есть /\ (и), \/ (или), \+ (не) и -> (следует). Определите предикат dnf(Formula), который выполняется, если Formula -- формула в дизъюнктивной нормальной форме (т.е. дизъюнкция ("или") конъюнкций переменных и их отрицаний, в качестве переменных используем атомы). ?- dnf((x /\ (\+ y) /\ z) \/ ((\+ x) /\ z)). Yes ?- dnf((x \/ (\+ y)) /\ (z \/ (\+ x) \/ y)). No 4. Определите предикат eval_logic(Formula, Values), который выполняется, если Formula -- логическая формула, Values -- список термов вида true(V) и false(V), где V -- атомы, которые используются в Formula, и в результате подстановки этих значений формула становится истинной. ?- eval_logic(\+ (x -> (\+ y)), [true(x), false(y)]). No
23	1. Определите (рекурсивно, не используя ^) предикат power(X, N, Power), который выполняется, если Power -- число X в степени N. ?- power(3, 2, X). X = 9 2. Представим многочлены как списки их коэффициентов в порядке убывания степени. Например, многочлен 2*x^2+1 будет представлен как [2,0,1]. Определите предикат eval_poly(Polynom, X, Result), выполняющийся, если Result -- значение многочлена Polynom при аргументе X. Для возведения в степень используйте оператор ^ или определённый в прошлой задаче power/3. Возможно также использование схемы Горнера. ?- eval_poly([2,0,1,0], 1, Result). % 2*X^3 + X = 2*1^3 + 1 = 3 Result = 3 3. В числе операторов, определённых в Prolog, есть /\ (и), \/ (или), \+ (не) и -> (следует). Определите предикат nnf(Formula), который выполняется, если Formula -- формула, в которой отрицания стоят только перед переменными (переменные в формуле представляем атомами). ?- nnf((x /\ (\+ y)) -> (\+ x) /\ z). Yes ?- nnf((x /\ (\+ y)) -> \+ (x /\ z)). No 4. Определите предикат nnf(Formula, NNF), который выполняется, если NNF -- формула, полученная из Formula эквивалентными преобразованиями, в которой отрицания стоят только перед переменными.

	?- nnf(\+ ((x -> y) /\ z), NNF). NNF = (x /\ (\+ y)) \/ (\+ z)
24	1. Определите предикат factorial(N, FactN), который выполняется, если FactN -- факториал N. ?- fact(3, X). X = 6 2. Определите предикат occurrences(Elem, List, Number), который выполняется, если элемент Elem встречается в списке List N раз. ?- occurrences(3, [1,2,3,1,3], N). N = 2 3. :-, ; и , -- обычные операторы в Prolog. Определите предикат rule(Rule), который проверяет, является ли терм Rule правилом (без точки в конце), то есть: 1. Голова правила может быть вызвана (в смысле callable/1); 2. Тело правила состоит из (1 или более) callable термов, разделённых ',' и ';' (заметьте, что T1,T2 и T1;T2 -- сами callable, но их нужно обрабатывать отдельно). ?- rule(p(X, Y) :- q(X)). Yes ?- rule(p :- q(X), r, r). Yes ?- rule(4 :- p, q). No ?- rule(p :- 4, q). No 4. Определите предикат eval_arithmetic(Expr, Values, Result), который выполняется, если Expr -- арифметическое выражение (содержащее операторы +, *, -, / и атомы в качестве переменных), Values -- список термов Var = Val, где Var -- атом, а Val -- число, а Result -- результат вычисления Expr при подстановке вместо каждого атома соответствующего ему значения из Values. ?- eval_arithmetic(x + 3*y, [x = 2, y = 1], Result). Result = 5
25	1. N-ное треугольное число -- сумма чисел 1+2+3+...+N. Определите (рекурсивно, не используя формулу N*(N+1)/2) предикат triangle(N, NthTriangle), который выполняется, если NthTriangle -- N-ное треугольное число. ?- triangle(4, X). X = 10 2. Определите предикат dot(List1, List2, DotProduct), который выполняется, если длины списков Dot1 и Dot2 совпадают, а DotProduct -- их скалярное произведение. ?- dot([1,2,3], [4,5,6], Result). Result = 26 %% т.е. 1*4+2*5+3*6

	3. Определите предикат polynom(Expr), который выполняется, если Expr -- многочлен в нормальной форме (т.е. слагаемые идут в порядке уменьшения степени и коэффициенты не равны 0). ?- polynom(2*x^3 + x). Yes 4. Определите предикат combination(List, K, Combination), который находит все комбинации по K элементов List как отдельные ответы. ?- combination([1,2,3], 2, C). C = [1,2] ; C = [1,3] ; C = [2,3] ; No

Дополнительные задания:

5. Определите предикат prime_factors(Num, Factors), находящий все простые делители числа Num и их кратность.

?- prime_factors(315, L).

L = [factor(3,2),factor(5,1),factor(7,1)] %% означает, что 315 = 3^2 * 5^1 * 7^1

6. Определите предикат polynomize(Expr, Poly), который выполняется, если Expr -- арифметическое выражение (включающее операции +, *, ^ в константную степень), а Poly -- многочлен в нормальной форме (см. 1.3), получающийся в результате его упрощения.

?- polynomize((x + x)*(x + 1), Poly).

Poly = 2*x^2 + 2*x

Критерии оценивания

	Задание сдано в срок	**Задание сдано позже**
Задача 1 выполнена верно		
Задача 2 выполнена верно		
Задача 3 выполнена верно		
Задача 4 выполнена верно		
Верно выполнено дополнительное задание 5		
Верно выполнено дополнительное задание 6		
Итого	7	3,5