Федеральное агентство связи

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

Кафедра прикладной математики и кибернетики

Лабораторная работа № 5 (язык SWI-Prolog)

**«Рекурсия, списки»**

по дисциплине «Функциональное и логическое программирование»

Бригада № 6

Выполнил: студент 3 курса группы ИП-811 Мироненко К. А

Проверил: доцент кафедры ПМиК Галкина М.Ю.

**Оглавление**

[1. Постановка задачи 3](#_Toc51530076)

[2. Примеры работы программы 4](#_Toc51530077)

[*Приложение* Листинг 6](#_Toc51530078)

# Постановка задачи

Каждая бригада выполняет все задачи.

1. Написать предикат, который печатает все нечётные числа из диапазона в порядке убывания. Границы диапазона вводятся с клавиатуры в процессе работы предиката.
2. Написать предикат, который находит числа Фибоначчи по их номерам, которые в цикле вводятся с клавиатуры. Запрос номера и нахождение соответствующего числа Фибоначчи должно осуществляться до тех пор, пока не будет введено отрицательное число.Циклический ввод организовать с помощью предиката **repeat**.

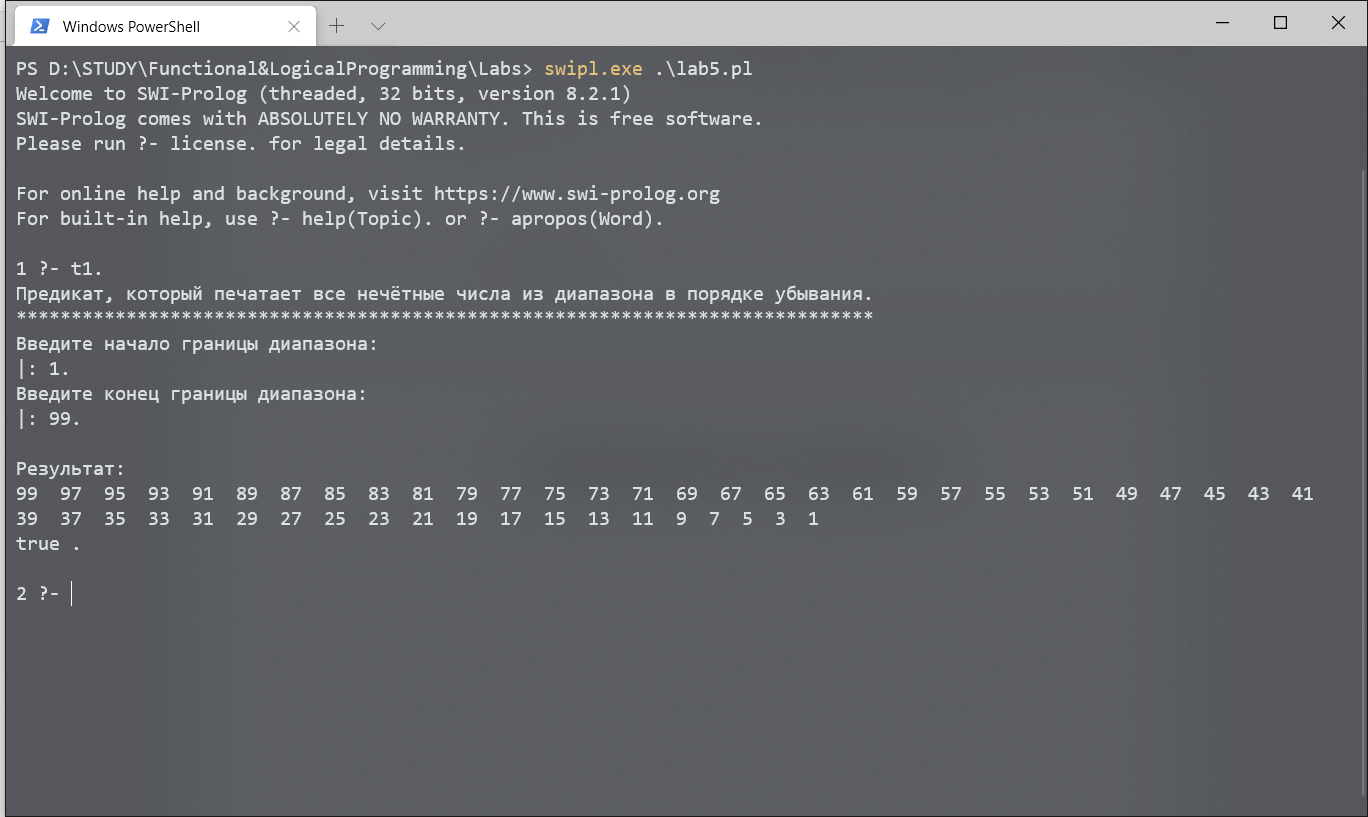
Числа Фибоначчи определяются по следующим формулам:

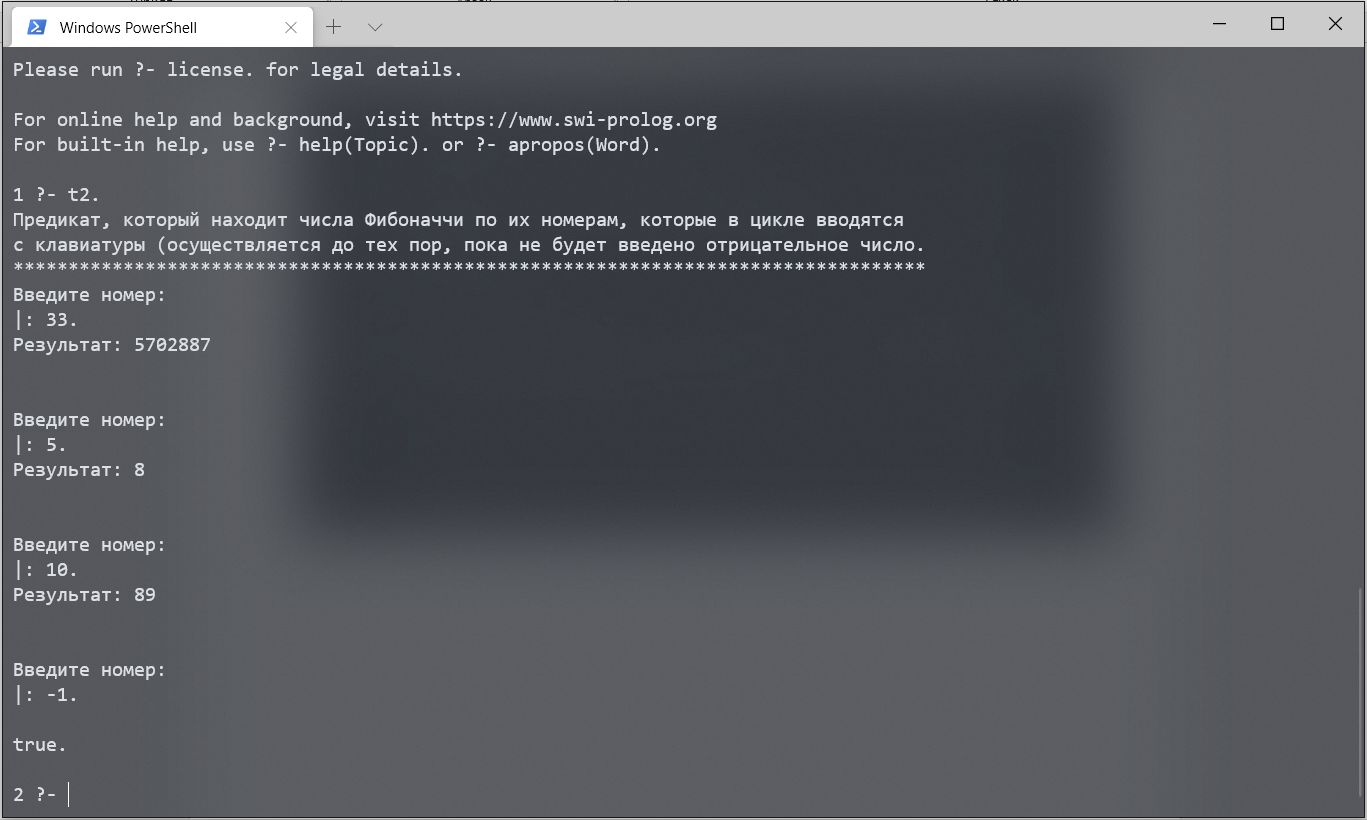
F(0)=1, F(1)=1, F(i)=F(i-2)+F(i-1) (i=2, 3, 4, ...).

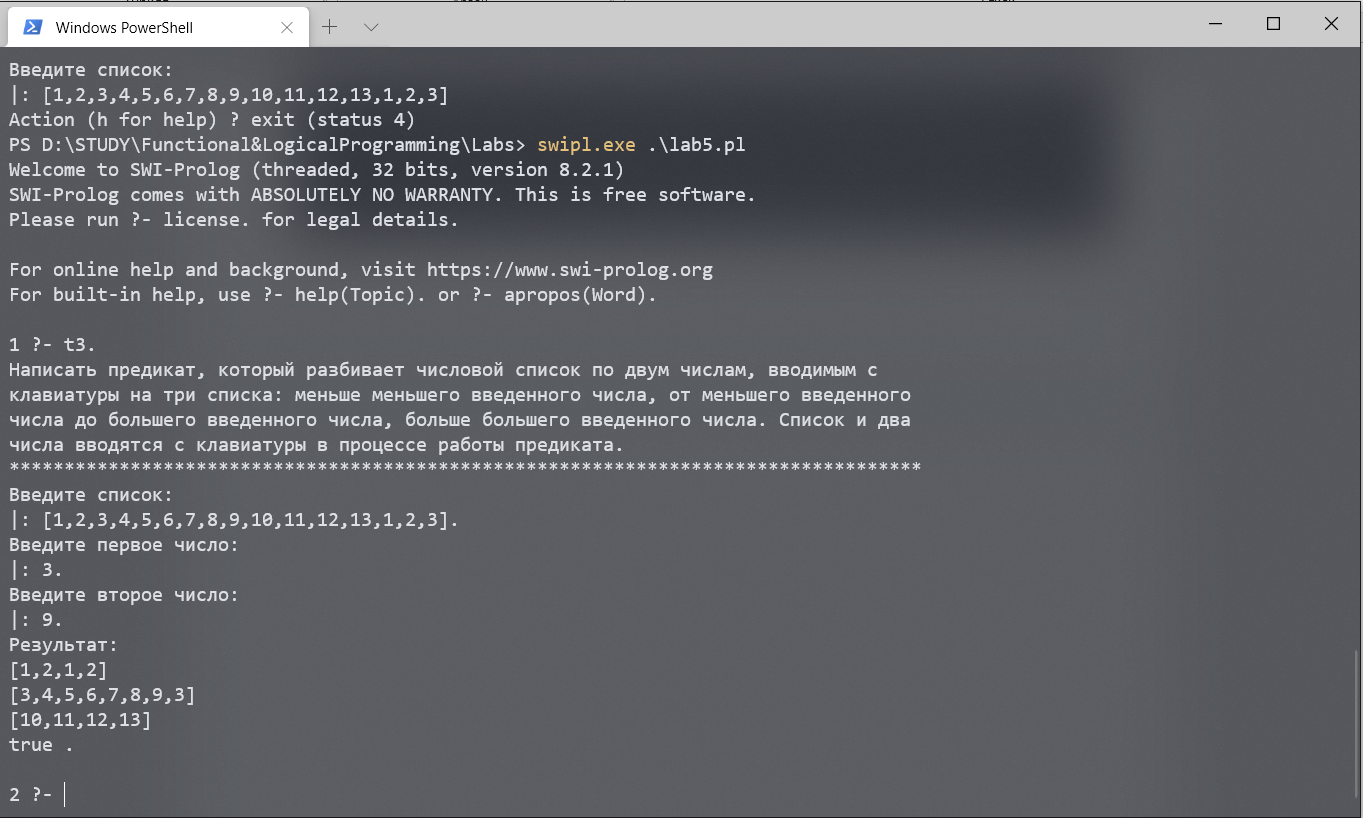
1. Написать предикат, который разбивает числовой список по двум числам, вводимым с клавиатуры на три списка: меньше меньшего введенного числа, от меньшего введенного числа до большего введенного числа, больше большего введенного числа. Список и два числа вводятся с клавиатуры в процессе работы предиката.
2. Написать предикат, который формирует список из наиболее часто встречающихся элементов списка. Список вводится с клавиатуры в процессе работы предиката.

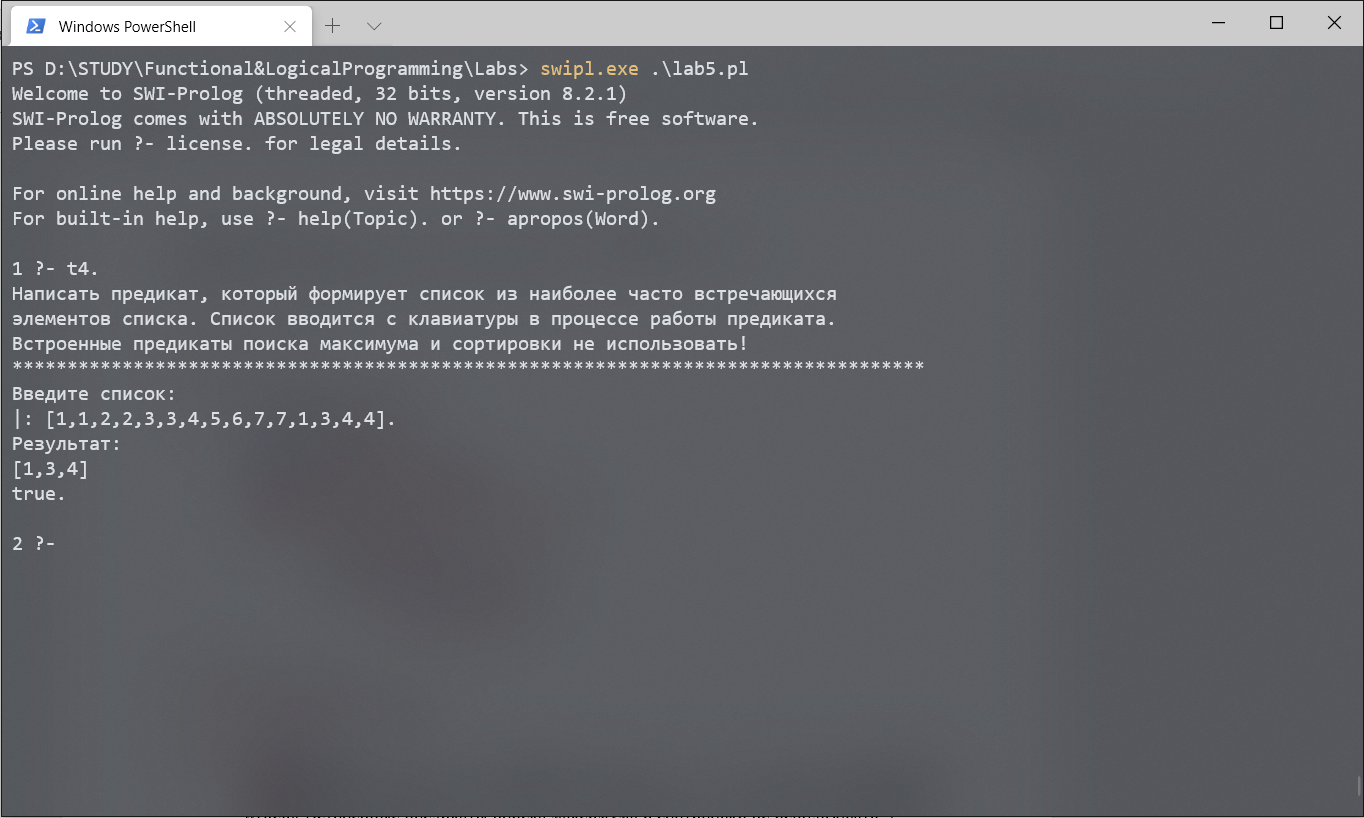
Встроенные предикаты поиска максимума и сортировки не использовать!

# Примеры работы программы









# *Приложение* Листинг

% Автор: kchipson

% Дата: 31.10.2020

/\* Каждая бригада выполняет все задачи \*/

/\* 1. Написать предикат, который печатает все нечётные числа из диапазона в порядке

убывания. Границы диапазона вводятся с клавиатуры в процессе работы предиката.\*/

uneven(X) :-

X mod 2 =\= 0.

uneven(X, Y) :-

Y >= X,

uneven(Y),

write(Y), write(' '), fail.

uneven(X, Y) :-

Y > X, Y1 is Y - 1, uneven(X, Y1).

uneven(X, Y) :-

Y == X.

print\_uneven() :-

writeln('Предикат, который печатает все нечётные числа из диапазона в порядке убывания.'),

writeln('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'),

writeln('Введите начало границы диапазона:'), read(X),

writeln('Введите конец границы диапазона:'), read(Y),

nl, writeln('Результат:'), uneven(X, Y).

t1() :- print\_uneven().

/\* 2. Написать предикат, который находит числа Фибоначчи по их номерам, которые в цикле

вводятся с клавиатуры. Запрос номера и нахождение соответствующего числа Фибоначчи

должно осуществляться до тех пор, пока не будет введено отрицательное число

Циклический ввод организовать с помощью предиката repeat.

Числа Фибоначчи определяются по следующим формулам:

F(0)=1, F(1)=1, F(i)=F(i-2)+F(i-1) (i=2, 3, 4, ...)\*/

fibonacci(X, Y) :-

X < 2, Y is 1, !.

fibonacci(X, Y) :-

X2 is X - 2,

X1 is X - 1,

fibonacci(X2, Y2),

fibonacci(X1, Y1),

Y is Y2 + Y1.

fibonacci() :-

repeat,

writeln('Введите номер:'),

read(X),

(X < 0, !; (fibonacci(X, RES), write('Результат: '), writeln(RES), nl, nl, fail)).

print\_fibonacci() :-

writeln('Предикат, который находит числа Фибоначчи по их номерам, которые в цикле вводятся'),

writeln('с клавиатуры (осуществляется до тех пор, пока не будет введено отрицательное число.'),

writeln('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'),

fibonacci().

t2() :- print\_fibonacci().

/\* 3. Написать предикат, который разбивает числовой список по двум числам, вводимым с

клавиатуры на три списка: меньше меньшего введенного числа, от меньшего введенного

числа до большего введенного числа, больше большего введенного числа. Список и два

числа вводятся с клавиатуры в процессе работы предиката.

Например: [3,7,1,-3,5,8,0,9,2], 8, 3 –> [1,-3,0,2], [3,7,5,8], [9]\*/

split(N1, N2, [H | T], [H | T1], L2, L3) :-

H < min(N1, N2), !,

split(N1, N2, T, T1, L2, L3).

split(N1, N2, [H | T], L1, [H | T2], L3) :-

H =< max(N1, N2), !,

split(N1, N2, T, L1, T2, L3).

split(N1, N2, [H | T], L1, L2, [H | T3]) :-

split(N1, N2, T, L1, L2, T3).

split(\_, \_, [], [], [], []).

print\_split() :-

writeln('Написать предикат, который разбивает числовой список по двум числам, вводимым с'),

writeln('клавиатуры на три списка: меньше меньшего введенного числа, от меньшего введенного'),

writeln('числа до большего введенного числа, больше большего введенного числа. Список и два'),

writeln('числа вводятся с клавиатуры в процессе работы предиката.'),

writeln('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'),

writeln('Введите список: '), read(L),

writeln('Введите первое число: '), read(N1),

writeln('Введите второе число: '), read(N2),

split(N1, N2, L, L1, L2, L3),

writeln('Результат: '), writeln(L1), writeln(L2), writeln(L3).

t3 :- print\_split().

/\* 4. Написать предикат, который формирует список из наиболее часто встречающихся

элементов списка. Список вводится с клавиатуры в процессе работы предиката.

Встроенные предикаты поиска максимума и сортировки не использовать!

Например: [0,3,5,7,1,5,3,0,3,3,5,7,0,5,0] –> [0,3,5].

\*/

mostСommon([], [], 0) :-

!.

mostСommon([H | T], L\_res, Max) :-

delete(T, H, L\_temp),

length([H | T], LenL),

length(L\_temp, LenL\_temp),

Max\_temp is LenL - LenL\_temp,

mostСommon(L\_temp, L\_new, Max\_maybe),

(Max\_temp > Max\_maybe ->

(L\_res = [H], Max is Max\_temp);

Max is Max\_maybe,

(Max\_temp =:= Max\_maybe ->

L\_res = [H | L\_new];

L\_res = L\_new)

).

mostСommon(L, L\_new) :-

mostСommon(L, L\_new, \_).

print\_mostСommon() :-

writeln('Написать предикат, который формирует список из наиболее часто встречающихся'),

writeln('элементов списка. Список вводится с клавиатуры в процессе работы предиката.'),

writeln('Встроенные предикаты поиска максимума и сортировки не использовать!'),

writeln('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'),

writeln('Введите список: '), read(L),

mostСommon(L, L\_new),

writeln('Результат: '), writeln(L\_new).

t4 :- print\_mostСommon().