Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ

по лабораторной работе № 18

по курсу «Функциональное и логическое программирование»

на тему: «Формирование и модификация списков на Prolog»

Студент ИУ7-61Б _____ Мицевич М. Д.
(Группа) (Подпись, дата) (И. О. Фамилия)

Преподаватель _____ Толпинская Н. Б.
(Подпись, дата) (И. О. Фамилия)

2022 г.

Постановка задачи

Задание: используя хвостовую рекурсию, разработать, комментируя аргументы, эффективную программу, позволяющую:

1. Сформировать список из элементов числового списка, больших заданного значения;
2. Сформировать список из элементов, стоящих на нечетных позициях исходного списка (нумерация от 0):
3. Удалить заданный элемент из списка (один или все вхождения);
4. Преобразовать список в множество (можно использовать ранее разработанные процедуры).

Листинг 1 – Решение задания №1

```
domains
        list = integer*

predicates
        bigger(list, integer, list)
        odd_list(list, list)
        set(list, list)
        rm_all(list, integer, list)
        rm_one(list, integer, list)

clauses
        odd_list([_, Head | Tail], [Head | ResTail]) :- odd_list(
           Tail, ResTail).
        odd_list([Head], []) :- !.
        odd_list([], []) :- !.

        bigger([Head | Tail], N, [Head | ResTail]) :- Head > N,
           !, bigger(Tail, N, ResTail).
        bigger([_ | Tail], N, Result) :- bigger(Tail, N, Result).
        bigger([], _, []).

        rm_one([Head | Tail], N, Tail) :- Head = N, !.
        rm_one([Head | Tail], N, [Head | ResList]) :- rm_one(Tail
           , N, ResList).
```

```
        rm_one([], _, []) :- !.

        rm_all([Head | Tail], N, [Head | ResList]) :- Head <> N,
           rm_all(Tail, N, ResList).
        rm_all([_ | Tail], N, ResList) :- rm_all(Tail, N, ResList
           ), !.
        rm_all([], _, []) :- !.

        set([Head | Tail], [Head | Result]) :- rm_all(Tail, Head,
            Nt), !, set(Nt, Result).
        set([], []).

goal
        %odd_list([1, 2, 3, 4, 5], Result).
        %bigger([1, 7, 3, 4, 5, 6, 2], 3, Result).

        %rm_one([1, 2, 3, 1, 2, 3, 1, 2, 3], 3, Result).
        %rm_all([1, 2, 3, 1, 2, 3, 1, 2, 3], 3, Result).

        set([1, 2, 3, 1, 2, 3, 1, 2, 3], Result).
```

Рисунок 1 – Таблица к заданию.

No шага	Состояние резольвенты, и вывод: дальнейшие действия (почему?)	Для каких термов запускается алгоритм унификации: T1=T2 и каков результат (и подстановка)	Дальнейшие действия: прямой ход или откат (почему и к чему приводит?)
1	odd_list([1, 2, 3, 4], Result)	odd_list([_, Head | Tail], [Head | ResTail]) и odd_list([1, 2, 3, 4], Result) **Результат:** успех + подстановка {_ = 1, Head = 2, Tail = [3, 4], Result = [2 | Restail]}	Заголовок правила заменяется его телом с учетом подстановки
2	odd_list([3, 4], ResTail)	odd_list([_, Head | Tail], [Head | ResTail]) и odd_list([3, 4], Result) **Результат:** успех + подстановка {_ = 3, Head = 4, Tail = [], Result = [4 | Restail]}	Заголовок правила заменяется его телом с учетом подстановки
3	odd_list([], ResTail)	odd_list([_, Head | Tail], [Head | ResTail]) и odd_list([], Result) **Результат:** неудача, термы не унифицируемы	Переход к следующему терму
4	/--/	/--/	/--/
5	odd_list([], ResTail)	odd_list([], []) и odd_list([], ResTail) **Результат:** успех + подстановка {Restail = []}	Заголовок правила заменяется его телом с учетом подстановки
6	!	успех	Переход к следующему терму
7			Решение найдено, возврат результатата Restail = [2, 4l]