



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

---

ФАКУЛЬТЕТ «Информатика и системы управления»

---

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

---

## Лабораторная работа № 19

|                |   |
|----------------|---|
| Дисциплина     | Функциональное и логическое программирование. |
| Тема           | Обработка списков на Prolog.                  |
| Студент        | Степанов А. О.                                |
| Группа         | ИУ7-63Б                                       |
| Оценка (баллы) |   |
| Преподаватель  | Толпинская Н.Б.                               |

Москва, 2020 г.

# ЗАДАНИЕ

Используя хвостовую рекурсию, разработать эффективную программу, (комментируя назначение аргументов), позволяющую:

1. Найти длину списка (по верхнему уровню);
2. Найти сумму элементов числового списка;
3. Найти сумму элементов числового списка, стоящих на нечетных позициях исходного списка (нумерация от 0).

Листинг 1: Текст программы

```
1 domains
2     list = integer*.
3     index = integer.
4
5 predicates
6     length(list, integer).
7     sum(list, integer).
8     sumOdd(list, integer).
9     sumOdd(list, integer, index).
10
11 clauses
12     length([], 0) :- !.
13     length([_|Tail], Length) :-
14         length(Tail, TailLength),
15         Length = TailLength + 1.
16
17     sum([], 0) :- !.
18     sum([Head|Tail], Sum) :-
19         sum(Tail, TailSum),
20         Sum = TailSum + Head.
21
22     sumOdd(List, Sum) :- sumOdd(List, Sum, 0).
23     sumOdd([], 0, _) :- !.
24     sumOdd([_|Tail], Sum, Index) :-
25         Index mod 2 = 0,
26         NextIndex = Index + 1,
27         sumOdd(Tail, Sum, NextIndex).
28     sumOdd([Head|Tail], Sum, Index) :-
```

```

29      Index mod 2 = 1,
30      NextIndex = Index + 1,
31      sumOdd(Tail, TailSum, NextIndex),
32      Sum = TailSum + Head.
33
34 goal
35      length([1, 2, 3, 4, 5], Length);
36      sum([1, 2, 3, 4, 5, 6], Sum);
37      sumOdd([1, 2, 1, 2, 1, 2], SumOdd).

```

## РЕЗУЛЬТАТ РАБОТЫ

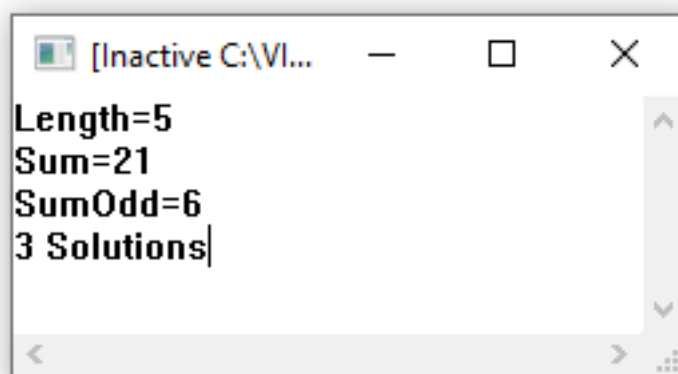


Рис. 1: Результат работы программы

## ФОРМИРОВАНИЕ ОТВЕТА

Для одного из вариантов ВОПРОСА и одного из заданий составить таблицу, отражающую конкретный порядок работы системы: Т.к. резольвента хранится в виде стека, то состояние резольвенты требуется отображать в столбик: вершина – сверху! Новый шаг надо начинать с нового состояния резольвенты! Для каждого запуска алгоритма унификации, требуется указать № выбранного правила и дальнейшие действия – и почему.

Таблица 1:  $\text{length}([1, 2, 3, 4, 5], \text{Length})$

| № ша-<br>га | Состояние резольвенты, и вы-<br>вод: дальнейшие действия (по-<br>чему?)   | Для каких термов запускается<br>алгоритм унификации: T1=T2<br>и каков результат (и подстанов-<br>ка)  | дальнейшие дей-<br>ствия: прямой ход<br>или откат (почему и<br>к чему приводит?) |
|-------------|---|---|--|
| 1           | $\text{length}([1, 2, 3, 4, 5], \text{Length})$   | Подстановка: $\text{Tail} = [2, 3, 4, 5]$ ,<br>$\text{Length} = \text{Length}$<br>$\text{length}([1, 2, 3, 4, 5], \text{Length})$<br>$\text{length}([\_ \text{Tail}], \text{Length})$ | Прямой ход   |
| 2           | $\text{length}(\text{Tail}, \text{TailLength})$<br><br>$\text{Length} = \text{TailLength} + 1$  | Подстановка: $\text{Tail} = [3, 4, 5]$ ,<br>$\text{Length} = \text{TailLength}$<br>$\text{length}([2, 3, 4, 5], \text{TailLength})$   | Прямой ход   |
| 3           | $\text{length}(\text{Tail}, \text{TailLength})$<br><br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$   | Подстановка: $\text{Tail} = [4, 5]$ ,<br>$\text{Length} = \text{TailLength}$<br>$\text{length}([3, 4, 5], \text{TailLength})$<br>$\text{length}([\_ \text{Tail}], \text{TailLength})$ | Прямой ход   |
| 4           | $\text{length}(\text{Tail}, \text{TailLength})$<br><br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$  | Подстановка: $\text{Tail} = [5]$ , $\text{Length} = \text{TailLength}$<br>$\text{length}([4, 5], \text{TailLength})$<br>$\text{length}([\_ \text{Tail}], \text{Length})$              | Прямой ход   |
| 5           | $\text{length}(\text{Tail}, \text{TailLength})$<br><br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$ | Подстановка: $\text{Tail} = []$ , $\text{Length} = \text{TailLength}$<br>$\text{length}([5], \text{TailLength})$<br>$\text{length}([\_ \text{Tail}], \text{Length}], 0)$              | Прямой ход   |
| 6           | $\text{length}(\text{Tail}, \text{TailLength})$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$     | Сравнение: $[]$ и $[]$<br>$\text{length}([], \text{TailLength})$<br>$\text{length}([], 0)$  | Прямой ход   |

|    |   |                                       |              |
|----|---|---------------------------------------|--------------|
|    | $\text{Length} = \text{TailLength} + 1$   |                                       |              |
| 7  | $\text{length}([], 0)$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$ | Подстановка: $\text{TailLength} = 0$  | Прямой ход   |
| 8  | $\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$                           | Подстановка: $\text{Length} = 1$      | Прямой ход   |
| 9  | $\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$  | Подстановка: $\text{Length} = 2$      | Прямой ход   |
| 10 | $\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$   | Подстановка: $\text{Length} = 3$      | Прямой ход   |
| 11 | $\text{Length} = \text{TailLength} + 1$<br>$\text{Length} = \text{TailLength} + 1$  | Подстановка: $\text{Length} = 4$      | Прямой ход   |
| 12 | $\text{Length} = \text{TailLength} + 1$   | Подстановка: $\text{Length} = 5$      | Прямой ход   |
| 13 | Пусто   | <b>Результат:</b> $\text{Length} = 5$ | Обратный ход |

## ВЫВОДЫ

Эффективность достигается за счет использования отсечения (!), которое останавливает поиски следующих фактов и правил.

## ВОПРОСЫ

1. **Что такое рекурсия? Как организуется хвостовая рекурсия в Prolog? Как можно организовать выход из рекурсии в Prolog?**

Рекурсия — это ссылка на определяемый объект во время его определения. В языке Prolog рекурсия организуется при помощи правила, в котором есть обращение к тому же правилу. Выход из рекурсии в Prolog организуется при помощи отсечения.

2. **Какое первое состояние резольвенты?**

Первое состояние резольвенты – вопрос.

3. **В каких пределах программы переменные уникальны?**

Областью действия переменной в Prolog является одно предложение. В разных предложениях может использоваться одно имя переменной для обозначения разных объектов. Исключением является анонимная переменная. Каждая анонимная переменная – это отдельный объект.

4. **В какой момент, и каким способом системе удастся получить доступ к голове списка?**

В момент подстановки переменной, с помощью конструкции `[Head|Tail]`, где `Head` – голова списка, а `Tail` – хвост.

5. **Каково назначение использования алгоритма унификации?**

Чтобы связать переменные из вопроса со значениями, находящимся в правилах и фактах.

6. **Каков результат работы алгоритма унификации?**

```
1 goal
2     P1 = birthday(person("Ivan", "Petrov"), date("August", 2, 1980)),
3     P1 = birthday(Name, date(_, _, 1980)), write(Name).
```

При согласовании первой подцели переменная `P1` получит значение, указанное справа от знака “=”. При согласовании второй подцели `P1` уже связана. Так как термы, находящиеся по обе стороны знака “=” сопоставимы, то переменная `Name` будет связана со значением `person(“Ivan”, “Petrov”)`. При согласовании третьей подцели, стандартного предиката `write`, будет напечатано значение связанной переменной `Name`.

## **7. Как формируется новое состояние резольвенты?**

Преобразование резольвенты выполняется с помощью редукции.

Редукция – замена цели телом того правила, заголовок которого унифицируется с целью. Новая резольвента получается в два этапа:

- (a) В текущей резольвенте выбирается одна из целей и для неё выполняется редукция  $\Rightarrow$  получаем новую конъюнкцию целей (новую резольвенту)
- (b) К полученной новой резольвенте применяется подстановка, как наибольший общий унификатор цели и заголовка правила, сопоставимого с этой целью.

## **8. Как применяется подстановка, полученная с помощью алгоритма унификации – как глубоко?**

Если унификация прошла успешно, то применяется подстановка. Переменные связываются со значениями.

## **9. В каких случаях запускается механизм отката?**

В том месте программы, где возможен выбор нескольких вариантов, Пролог сохраняет в специальный стек точку возврата для последующего возвращения в эту позицию. Точка возврата содержит информацию, необходимую для возобновления процедуры при откате. Выбирается один из возможных вариантов, после чего продолжается выполнение программы.

Во всех точках программы, где существуют альтернативы, в стек заносятся указатели. Если впоследствии окажется, что выбранный вариант не приводит к успеху, то осуществляется откат к последней из имеющихся в стеке точек программы, где был выбран один из альтернативных вариантов. Выбирается очередной вариант, программа продолжает свою работу. Если все варианты в точке уже были использованы, то регистрируется неудачное завершение и осуществляется переход на предыдущую точку возврата, если такая есть. При откате все связанные переменные, которые были означены после этой точки, опять освобождаются.

## **10. Когда останавливается работа системы? Как это определяется на формальном уровне?**

Работа системы останавливается в двух случаях:

- Когда встретился символ отсечения (!);
- Когда резольвента осталась пустой (формально не осталось подходящих фактов и правил).