МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования «Южно-Уральский государственный университет»

(национальный исследовательский университет)

Высшая школа электроники и компьютерных наук

Кафедра системного программирования

Списки, операции с ними

(название практической работы)

ОТЧЕТ

по практической работе № 5

по дисциплине «Функциональное и логическое программирование»

Выполнил:

студент группы КЭ–401

/ Д.А. Иванов /

(подпись)

« » 2024 г.

Проверил:

преподаватель кафедры СП

/ М.А. Ческидова /

(подпись)

« » 2024 г.

Оглавление

[1. Задание 1 3](#_Toc179292138)

[2. Задание 2 5](#_Toc179292139)

[3. Задание 3 7](#_Toc179292140)

[4. Задание 4 9](#_Toc179292141)

[5. Задание 5 10](#_Toc179292142)

[6. Задание 6 11](#_Toc179292143)

[7. Задание 7 13](#_Toc179292144)

[8. ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ 14](#_Toc179292145)

# Задание 1

Составить программу, содержащую список из элементов целого типа. Вывести на экран:

* все элементы списка;
* количество элементов списка;
* N-ый элемент списка или сообщение, что такой элемент не существует.

На листинге 1 представлен код для вывода требуемых фильмов полным списком.

Листинг 1 ­– Список чисел

list([2, 3, 4, 5, 6, 10, 8, 9, 7, 1, 0, 44, 33, 22, 11, 66, 55, 44, 33]).

print\_list :-

list(L),

write(L), nl.

count\_elements([], 0).

count\_elements([\_|T], Count) :-

count\_elements(T, Count1),

Count is Count1 + 1.

nth\_element(1, [H|\_], H).

nth\_element(N, [\_|T], Element) :-

N > 1,

N1 is N - 1,

nth\_element(N1, T, Element).

count :-

list(L),

count\_elements(L, Count),

write(Count), nl.

nth(N, Element) :-

list(L),

nth\_element(N, L, Element).

На рисунке 1 представлены результаты выполнения этой программы.

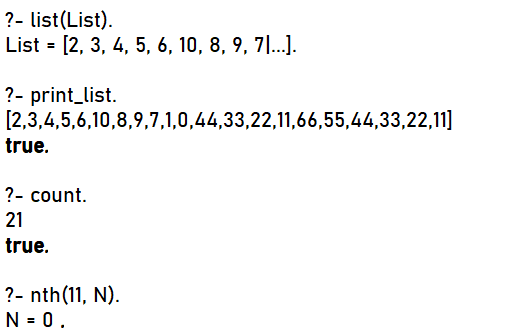


Рисунок 1 – Результат выполнения задания 1

# Задание 2

На основе, созданной ранее базы знаний «Фильмы» выполните следующие задания:

* создайте список названий фильмов;
* создайте список из жанров;
* создайте список из дат (элементами списка должны быть структуры!)
* выведите названия фильмов и в той же строке - соответствующий жанр (работа с двумя списками одновременно).

На листинге 2 представлен код программы удовлетворяющией требованиям.

Листинг 2 – Список фильмов

фильм('Искупление', 'Великобритания', 130, 'мелодрама', date(19, 2, 2008)).

фильм('Консервы', 'Россия', 115, 'триллер', date(22, 2, 2007)).

фильм('Параграф 78', 'Россия', 90, 'боевик', date(22, 2, 2007)).

фильм('Дети шпионов', 'США', 90, 'комедия', date(30, 3, 2001)).

фильм('Скрытая угроза', 'США', 90, 'триллер', date(2, 11, 2001)).

фильм('Дочь моего босса', 'США', 90, 'комедия', date(22, 8, 2003)).

% Создание списка названий фильмов

список\_названий(Список) :-

findall(Название, фильм(Название, \_, \_, \_, \_), Список).

% Создание списка жанров

список\_жанров(Список) :-

findall(Жанр, фильм(\_, \_, \_, Жанр, \_), Список).

% Создание списка дат

список\_дат(Список) :-

findall(Дата, фильм(\_, \_, \_, \_, Дата), Список).

% Вывод названий фильмов и соответствующих жанров

название\_и\_жанр :-

findall((Название, Жанр), фильм(Название, \_, \_, Жанр, \_), Список),

вывести\_список(Список).

вывести\_список([]).

вывести\_список([(Название, Жанр)|Остальные]) :-

write(Название), write(' - '), write(Жанр), nl,

вывести\_список(Остальные).

Результат представлен на рисунке 2.

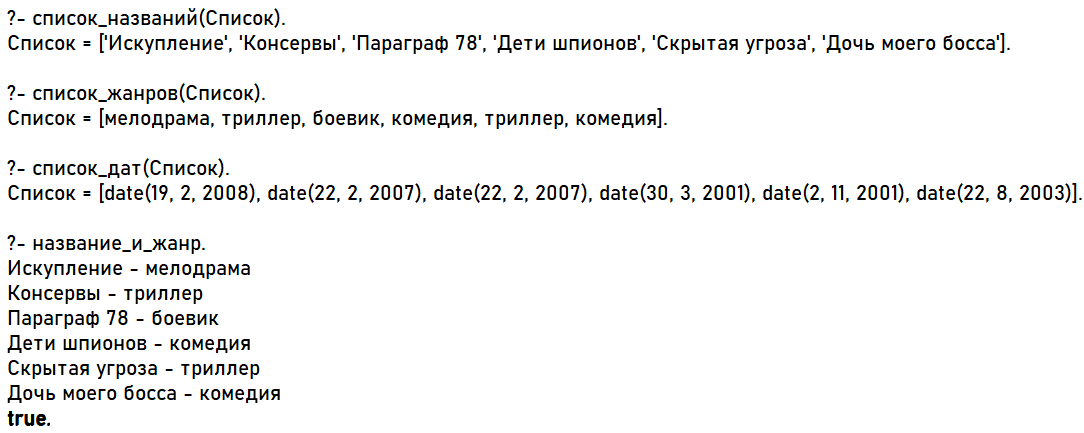


Рисунок 2 – Результат выполнения задания 2

# Задание 3

На языке prolog cоставить программу поиска элемента в списке из целых чисел. Представить порядок вычисления цели для поиска элемента в списке, когда элемент найден, и для случая, когда элемент не найден.

На листинге 3 представлен код программы для поиска элемента в списке.

Листинг 3 – Поиск числа

member(X, [X|\_]).

member(X, [\_|Tail]) :-

member(X, Tail).

На рисунке 3 представлен результат выполнения задания 3.

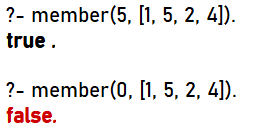


Рисунок 3 – Результат выполнения задания 3

Порядок вычисления цели для случая, когда элемент найден (например, 3 в списке [1, 2, 3, 4]):

* программа начинает с первой цели: member(3, [1, 2, 3, 4]);
* проверяется первый элемент: 3 = 1 (не соответствует);
* переход к хвосту: member(3, [2, 3, 4]);
* проверяется первый элемент: 3 = 2 (не соответствует);
* переход к хвосту: member(3, [3, 4]);
* проверяется первый элемент: 3 = 3 (соответствует). Цель выполнена.

Порядок вычисления цели для случая, когда элемент найден элемент не найден (например, 5 в списке [1, 2, 3, 4]):

* программа начинает с первой цели: member(5, [1, 2, 3, 4]);
* проверяется первый элемент: 5 = 1 (не соответствует);
* переход к хвосту: member(5, [2, 3, 4]);
* проверяется первый элемент: 5 = 2 (не соответствует);
* переход к хвосту: member(5, [3, 4]);
* проверяется первый элемент: 5 = 3 (не соответствует);
* переход к хвосту: member(5, [4]);
* проверяется первый элемент: 5 = 4 (не соответствует);
* хвост пустой, и дальнейших элементов нет. Цель не выполнена.

# Задание 4

Даны два списка [5,4] и [3,2,1]. Слить их в один.

Для слияния двух списков в Prolog используется предикат append, который объединяет два списка в один.

На листинге 4 представлен код предиката append.

Листинг 4 ­– Предикат append

append([], L, L).

append([H|T], L, [H|R]) :-

append(T, L, R).

На рисунке 4 представлен результат выполнения этой программы.



Рисунок 4 – Результат выполнения задания 4

Порядок вычисления представляет из себя следующий алгоритм:

* если первый список пуст, результатом является второй список;
* если первый список непустой, добавляем его головной элемент к результату;
* рекурсивно объединяем хвост с вторым списком.

# Задание 5

Дан список [23,1,34,2,45,5,4]. Разделить этот список на два.

На листинге 5 представлен код для разделения списка пополам.

Листинг 5 ­– Разделение списка

split\_list(List, Left, Right) :-

length(List, N),

Half is N // 2,

length(Left, Half),

append(Left, Right, List).

На рисунке 5 представлен результат выполнения этой программы.

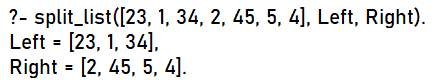


Рисунок 5 – Результат выполнения задания 5

Порядок вычисления представляет из себя следующий алгоритм:

* определяем длину списка N;
* задаем длину первой половины (N/2);
* определяем Right такой, чтобы получался исходный список.

# Задание 6

На основе, базы знаний «Фильмы» выполните следующие задания:

* разъедините списки из длительностей фильмов и их названий на 2 списка по значению длительности, вводимому пользователем (например, в первые списки поместить длительности и названия фильмов, которые идут менее 120 минут, а во вторые- более 120 минут);
* создайте список из годов дат и найдите год самой последней премьеры (максимальный элемент списка)
* создайте список из стран и удалите из него повторяющиеся элементы так, чтобы наименование страны осталось в списке только один раз.

На листинге 6 представлен код программы, удовлетворяющей требованиям.

Листинг 6 ­– Работа со списками фильмов

фильм('Искупление', 'Великобритания', 130, 'мелодрама', date(19, 2, 2008)).

фильм('Консервы', 'Россия', 115, 'триллер', date(22, 2, 2007)).

фильм('Параграф 78', 'Россия', 90, 'боевик', date(22, 2, 2007)).

фильм('Дети шпионов', 'США', 90, 'комедия', date(30, 3, 2001)).

фильм('Скрытая угроза', 'США', 90, 'триллер', date(2, 11, 2001)).

фильм('Дочь моего босса', 'США', 90, 'комедия', date(22, 8, 2003)).

% Разделение списков по длительности фильма

разделить\_по\_длительности(Длительность) :-

findall(Название, (фильм(Название, \_, Длина, \_, \_), Длина < Длительность), Короткие),

findall(Название, (фильм(Название, \_, Длина, \_, \_), Длина >= Длительность), Длинные),

writeln('Фильмы длительностью менее введенного значения:'),

writeln(Короткие),

writeln('Фильмы длительностью больше или равно введенному значению:'),

writeln(Длинные).

% Создание списка годов дат премьер

список\_годов(Список) :-

findall(Год, фильм(\_, \_, \_, \_, date(\_, \_, Год)), Список).

% Нахождение самой поздней премьеры

максимальный\_год :-

список\_годов(Годы),

max\_list(Годы, Максимум),

writeln('Самая поздняя премьера:'),

writeln(Максимум).

% Создание списка стран без повторений

список\_стран(Список) :-

findall(Страна, фильм(\_, Страна, \_, \_, \_), Все\_страны),

sort(Все\_страны, Список).

На рисунке 6 представлен результат выполнения этой программы.

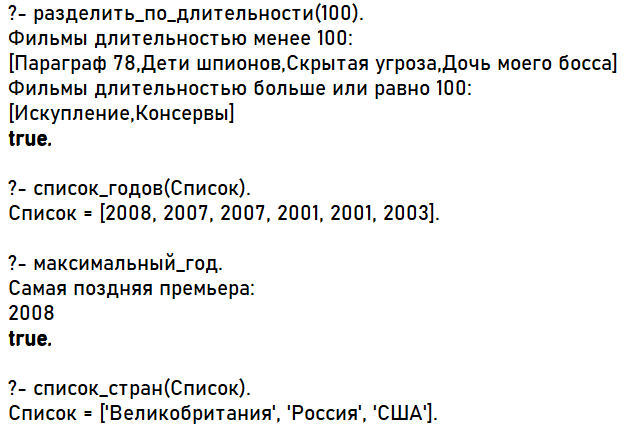


Рисунок 6 – Результат выполнения задания 6

# Задание 7

Сформируйте список из элементов, вводимых с клавиатуры, признаком окончания ввода должен быть ввод пустой строки.

На листинге 7 представлен код для формирования списка из элементов, вводимых с клавиатуры.

Листинг 7 ­– Формирования списка с клавиатуры

read\_list(List) :-

read\_items([], List).

read\_items(Acc, List) :-

write('Введите элемент (или пустую строку для завершения): '),

read\_line\_to\_string(user\_input, Input),

(

Input = "" ->

List = Acc

;

append(Acc, [Input], NewAcc),

read\_items(NewAcc, List)

).

На рисунке 7 представлен результат выполнения этой программы.

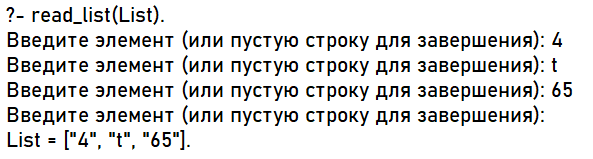


Рисунок 7 – Результат выполнения задания 4

# ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

Найти подсписок с n-го элемента длиной m.

На листинге 8 представлен код программы для поиска нужного подсписка.

Листинг 8 ­– Подсписок

sublist(List, N, M, Sublist) :-

skip\_elements(List, N, Rest),

take\_elements(Rest, M, Sublist).

skip\_elements(List, 1, List).

skip\_elements([\_|Tail], N, Rest) :-

N > 1,

N1 is N - 1,

skip\_elements(Tail, N1, Rest).

take\_elements(\_, 0, []).

take\_elements([H|T], M, [H|Rest]) :-

M > 0,

M1 is M - 1,

take\_elements(T, M1, Rest).

На рисунке 8 представлен результат выполнения этой программы.



Рисунок 8 – Результат выполнения задания 8