МИНИСТЕРСТВО НАУКИ И ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ АГЕНТСВО ПО ОБРАЗОВАНИЮ
БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ
УНИВЕРСИТЕТ ИМ В.Г.ШУХОВА

Математическая логика и теория алгоритмов

Методические указания к выполнению лабораторных работ и РГЗ

Терехов Д.В., Куценко Д.А.

Белгород 2007 (v 1.0.1)

В данном пособии приведен перечень лабораторных работ и РГЗ по математической логике и теории алгоритмов. Каждая работа состоит из теоретической и практической частей. Теоретическая часть направлена на освоение студентом теоретического материала, содержащегося в работе, а также решение задач общего характера. Практическая часть направлена закрепление полученных теоретических знаний посредством решения прикладных задач на соответствующую тематику.

РГЗ №1

Высказывания, логика высказываний

Теоретические сведения

Теоретический материал, необходимый для выполнения РГЗ можно найти в учебном пособии по математической логике (Терехов, Куценко).

Задания к выполнению лабораторной работы

Теоретическая часть:

Решить задачи из лабораторного практикума по математической логике (Хачатрян, Гончарова) согласно своему варианту (см. таблицу).

Варианты заданий:

№	Номера задач (занятия 1 и 2)														
1	2.4	6.1	7.16	9.3	10.21	13.4	16.4	22.3	29.2	34.8	37.4	47.2	60.2	70.3	73.1
2	2.5	6.2	7.17	9.4	11.1	13.5	17	22.4	29.3	34.9	37.5	48.1	60.3	70.4	73.2
3	2.6	6.3	7.18	9.5	11.2	13.6	18.4	22.5	29.4	34.10	37.6	48.2	60.4	70.5	73.3
4	2.7	6.4	7.19	9.6	11.3	13.7	18.2	22.6	29.5	34.11	37.7	48.3	62.1	70.6	73.4
5	2.8	6.5	7.20	9.7	11.4	13.8	18.3	22.7	29.6	34.12	37.8	48.4	62.2	70.7	73.5
6	2.9	6.6	7.21	9.8	11.5	13.9	19.1	22.8	29.7	34.13	38.1	49.2	62.3	70.8	73.6
7	2.11	6.7	7.22	9.9	11.7	13.10	19.2	23.1	29.8	34.14	38.2	49.3	62.5	70.9	73.7
8	2.12	6.8	7.23	9.10	11.9	13.11	19.3	23.2	29.9	34.15	38.3	50.1.а	64.1	70.10	73.8
9	2.13	6.9	8.1	9.11	11.10	13.12	19.4	23.3	30	34.16	38.4	50.1.б	64.2	70.11	73.9
10	2.14	6.11	8.2	9.12	11.12	13.13	19.5	24.1	31.1	35.1	38.5	50.2.а	64.3	70.12	73.10
11	3.а.2	6.12	8.3	9.14	11.13	13.14	19.7	24.2	31.2	35.3	38.6	50.2.б	64.4	70.13	73.11
12	3.а.3	6.13	8.4	10.1	11.14	13.17	19.8	24.3	31.3	35.4	38.7	51.а	67.2	70.14	73.12
13	3.а.4	6.14	8.5	10.2	11.15	13.18	20.1	24.4	31.4	35.5	38.8	51.б	67.3	70.15	73.13
14	3.б.1	6.15	8.6	10.3	11.16	13.19	20.2	25.1	31.5	35.6	38.9	52	67.4	70.16	73.14
15	3.б.2	6.16	8.7	10.5	11.17	14.1	20.4	25.2	31.6	35.7	38.10	53	67.5	70.17	73.15
16	3.б.3	6.17	8.8	10.6	11.18	14.2	20.5	25.3	31.7	35.8	39	54	67.6	70.18	73.16
17	3.б.4	6.18	8.9	10.7	11.19	14.3	20.6	25.4	31.8	35.9	40.а	55	67.7	70.19	73.17
18	3.б.5	7.1	8.10	10.8	11.20	14.4	20.7	25.5	31.9	35.10	40.б	56	67.8	70.20	73.18
19	4.1	7.2	8.11	10.9	12.1	14.6	20.8	26	31.10	36.1	41.а	57	68.1	70.21	73.19
20	4.3	7.3	8.12	10.10	12.2	14.7	21.1	27.1	31.11	36.2	41.б	58.1	68.2	70.22	73.20
21	4.4	7.4	8.13	10.11	12.3	14.8	21.2	27.2	31.12	36.3	42	58.2	68.3	71.1	73.21
22	4.5	7.5	8.15	10.12	12.4	14.9	21.3	28.1	31.13	36.4	43.а	58.3	63.1	71.2	73.22
23	4.6	7.6	8.16	10.13	12.5	15.1	21.4	28.2	32	36.5	43.б	58.5	63.2	71.3	73.23
24	4.7	7.8	8.17	10.14	12.6	15.2	21.5	28.3	33	36.6	43.в	58.6	72.1	71.4	73.24
25	4.8	7.10	8.18	10.15	12.8	15.3	21.6	28.4	34.1	36.7	43.г	58.7	72.2	71.5	73.25
26	5.10	7.11	8.19	10.16	12.9	15.4	21.7	28.5	34.2	36.8	44	58.8	72.3	71.6	73.26
27	5.9	7.12	8.20	10.17	12.10	15.5	21.8	28.6	34.3	36.9	46.1	58.9	72.4	71.7	73.27
28	5.8	7.13	8.21	10.18	13.1	16.1	21.9	28.7.	34.5	36.10	46.2	58.10	72.5	71.8	73.28
29	5.6	7.14	9.1	10.19	13.2	16.2	21.10	28.8	34.6	37.2	46.3	58.11	70.1	71.9	73.29
30	5.7	7.15	9.2	10.20	13.3	16.3	22.1	29.1	34.7	37.3	47.1	58.12	70.2	71.10	73.30

Практическая часть:

Разработать программный модуль, способный находить значение формулы, представленной в нормальной форме на данной интерпретации.

Разработать программу, способную считывать формулу логики высказываний в одной из нормальных форм (по выбору пользователя) и находить значения данной формулы на вводимых пользователем интерпретациях.

Содержание отчета

Название и цель лабораторной работы.
Решение предложенных в теоретической части задач.
Программу на выбранном языке программирования в виде исходных кодов (с поясняющими комментариями) и в электронном варианте для демонстрации на ЭВМ.
Спецификацию программы с указанием основных структур данных и алгоритмов.
Примеры работы программы на тестовых данных.

РГЗ №2

Предикаты, логика предикатов, метод резолюций

Теоретические сведения

Теоретический материал, необходимый для выполнения РГЗ можно найти в учебном пособии по математической логике (Терехов, Куценко)

Задания к выполнению лабораторной работы

Теоретическая часть:

Решить задачи из лабораторного практикума по математической логике (Хачатрян, Гончарова) согласно своему варианту (см. таблицу).

Варианты заданий:

№	Номера задач				
1	17	5.3	10.13	1.1	11
2	18.1	5.4	10.14	1.2	12
3	18.2	5.6	10.15	1.3	13
4	18.3	6.1	11.1	1.4	14
5	19	6.2	11.2	1.5	15
6	20	6.3	11.3	1.6	16
7	21	6.4	12.1	1.7	17
8	22	7	12.2	1.8	18
9	2.2	8.1	12.3	1.9	19
10	2.3	8.2	12.4	1.10	20
11	2.4	8.3	12.5	1.11	21
12	2.5	8.4	13.1	2.1	22
13	2.6	8.5	13.2	2.2	23
14	3.1	8.6	13.3	3	24
15	3.2	9.1	14.1	4.1	25
16	3.3	9.2	14.2	4.2	26
17	3.4	9.3	14.3	4.3	27
18	3.5	9.4	14.4	5	28
19	3.6	10.1	14.5	6	29
20	3.7	10.2	14.6	7.1	30
21	4.1	10.3	14.7	7.2	31
22	4.2	10.4	15.1	7.3	32
23	4.3	10.5	15.2	7.4	33
24	4.4	10.6	15.3	7.5	34
25	4.5	10.7	16.1	8.1	35
26	4.6	10.8	16.2	8.2	36
27	4.7	10.9	16.3	8.3	14
28	4.8	10.10	16.4	8.4	15
29	5.1	10.11	16.5	9	16
30	5.2	10.12	16.6	10	17

Практическая часть:

Разработать программу, способную считывать формулу логики высказываний в одной из нормальных форм (в соответствии с вариантом), решающую задачу согласно своему варианту из предложенного списка

заданий. При решении задачи воспользоваться разработками из предыдущей лабораторной работы. Алгоритмы, выполняющие решение задачи, должны содержаться в отдельном модуле.

Варианты заданий к практической части:

1. Программа должна строить полную таблицу истинности введенной пользователем формулы логики высказываний.
2. Программа должна доказывать общезначимость введенной пользователем формулы логики высказываний.
3. Программа должна доказывать противоречивость введенной пользователем формулы логики высказываний.
4. Программа должна отыскивать интерпретации, на которых формула логики высказываний, введенная пользователем, принимает истинное значение.
5. Программа должна отыскивать интерпретации, на которых формула логики высказываний, введенная пользователем, принимает ложное значение.

Если деление второй цифры варианта на 2 дает нулевой остаток, то при решении задачи необходимо работать с КНФ, иначе с ДНФ. К примеру, вариант студента 7, значит вторая цифра варианта 0 (07), следовательно, остаток от деления – 0 и программная реализация должна выполнятся на КНФ (вариант при этом необходимо взять 2й, поскольку 7 (вариант) по модулю 5 (кол-во предложенных заданий) равно 2м).

Дополнительное задание повышенной сложности (по желанию студента): написать программу, отыскивающую совершенную нормальную форму введенной пользователем формулы логики высказываний.

Содержание отчета

Название и цель лабораторной работы.
Решение предложенных в теоретической части задач.
Программу на выбранном языке программирования в виде исходных кодов (с поясняющими комментариями) и в электронном варианте для демонстрации на ЭВМ.
Спецификацию программы с указанием основных структур данных и алгоритмов.
Примеры работы программы на тестовых данных.

Лабораторная работа №1

Машина Тьюринга-Поста, язык BrainFuck

Теоретические сведения

Теоретический материал, необходимый для выполнения лабораторной работы можно найти в учебном пособии по математической логике (Терехов, Куценко).

Brainfuck — один из известнейших эзотерических языков программирования, придуман Урбаном Мюллером (Urban Müller) в 1993 году для забавы. Язык имеет восемь команд, каждая из которых записывается одним символом. Исходный код программы на Brainfuck представляет собой последовательность этих символов без какого-либо дополнительного синтаксиса.

Машина, которой управляют команды Brainfuck, состоит из упорядоченного набора ячеек и указателя текущей ячейки, напоминая ленту и головку машины Тьюринга. Кроме того, подразумевается устройство общения с внешним миром (см. команды . и ,) через поток ввода и поток вывода.8 команд языка Brainfuck:

> перейти к следующей ячейке
< перейти к предыдущей ячейке
\+ увеличить значение в текущей ячейке на 1
\- уменьшить значение в текущей ячейке на 1
. напечатать значение из текущей ячейки
, ввести извне значение и сохранить в текущей ячейке
[если значение текущей ячейки нуль, перейти вперёд по тексту программы на ячейку, следующую за соответствующей] (с учётом вложенности)
] если значение текущей ячейки не нуль, перейти назад по тексту программы на ячейку, следующую за соответствующей [(с учётом вложенности)

Язык Brainfuck можно описать с помощью эквивалентов языка Си (предполагается, что переменная ptr объявлена как указатель на байт):

команда Brainfuck	Си эквивалент
>	++ptr
<	--ptr
+	++*ptr
-	--*ptr
.	putchar(*ptr)
,	*ptr=getchar()

[	while (*ptr) {
]	}

Программа на языке Brainfuck, печатающая «Hello World!»:

```
;Подготовка в памяти (с ячейки 1) массива значений близких к ASCII-кодам символов, которые необходимо вывести (70, 100, 30, 10), через повторение 10 раз приращения ячеек на 7, 10, 3 и 1, соответственно
++++++++++ ;присваивание ячейке 0 (счетчику) значения 10
[ ;повторять, пока значение текущей ячейки (ячейки 0) больше нуля
     >+++++++ ;приращение ячейки 1 на 7
     >++++++++++ ;приращение ячейки 2 на 10
     >+++ ;приращение ячейки 3 на 3
     >+ ;приращение ячейки 4 на 1
     <<<<- ;возврат к ячейке 0 (счетчику), и его уменьшение на 1
] ;вернуться к началу цикла

;Получение кодов букв и их вывод
>++. ;Вывод «H». Получение кода «H» (72) из 70 в ячейке 1 и вывод
>+.  ;Вывод «e». Получение кода «e» (101) из 100 в ячейке 2 и вывод
+++++++.. ;Вывод «ll». Получение кода «l» (108) из 101 в ячейке 2 и вывод дважды
+++. ;Вывод «o». Получение кода «o» (111) из 108 в ячейке 2 и вывод
>++. ;Вывод пробела. Получение кода пробела (32) из 30 в ячейке 3 и вывод
<<+++++++++++++++. ;Вывод «W». Получение кода «W» (87) из 72 в ячейке 1 и вывод
>.   ;Вывод «o». Код «o» (111) уже находится в ячейке 2, просто его выводим
+++. ;Вывод «r». Получение кода «r» (114) из 111 в ячейке 2 и вывод
------.   ;Вывод «l». Получение кода «l» (108) из 114 в ячейке 2 и вывод
--------. ;Вывод «d». Получение кода «d» (100) из 108 в ячейке 2 и вывод
>+.  ;Вывод «!». Получение кода «!» (33) из 32 в ячейке 3 и вывод
>.   ;Вывод кода перевода строки (10) из ячейки 4
```

Те же действия выполняет и следующая программа, но она будет в три с лишним раза больше, чем приведенный выше оптимизированный вариант:

```
++++++++++++++++++++++++++++++++++++++++++++++++
 ++++++++++++++++.
++++++++++++++++++++++++++++.
+++++++..
+++.
```

```
-------------------------------------------------------
 ----------------------.
++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 .
++++++++++++++++++++++.
+++.
------.
--------.
-------------------------------------------------------
 ------------.
----------------------.
```

Несмотря на внешнюю примитивность, Brainfuck с бесконечным набором ячеек имеет тьюринговскую полноту, а, следовательно, по потенциальным возможностям не уступает «настоящим» языкам, подобным Си, Паскалю или Java. Brainfuck подходит для экспериментов по генетическому программированию из-за простоты синтаксиса, и, соответственно, генерации исходного кода.

В «классическом» Brainfuck, описанном Мюллером, размер ячейки — один байт, количество ячеек 30000. В начальном состоянии указатель находится в крайней левой позиции, а все ячейки заполнены нулями. Увеличение/уменьшение значений ячеек происходит по модулю 256. Ввод/вывод также происходит побайтно, с учётом кодировки ASCII (то есть в результате операции ввода (,) символ 1 будет записан в текущую ячейку как число 0x31 (49), а операция вывода (.), совершённая над ячейкой, содержащей 0x41 (65), напечатает латинскую A). В других вариантах языка размер и количество ячеек может быть другим (большим). Есть версии, где значение ячеек не целочислено (с плавающей точкой).

Существует расширение классического языка BrainFuck, чтобы можно было использовать процедуры. Для поддержки этого расширения дополнительно вводятся 3 новых команды:

(начало декларации процедуры; идентификатором процедуры служит числовое значение, находящееся в текущей ячейке
) конец декларации процедуры
: - вызов процедуры с идентификатором, равным числу, которое находится в текущей ячейке

Пример программы с процедурами:

```
;процедура 1, очищающая (в 0) 3  следующих ячейки от
 текущей
+( ;увеличиваем значение текущей ячейки на 1 (было 0).
 Это будет идентификатор процедуры
     >[-] ;очищаем 1ю ячейку справа
     >[-] ;очищаем 2ю ячейку справа
     >[-] ;очищаем 3ю ячейку справа
     <<< ;возвращаемся на ячейку, с которой начали
) ;конец процедуры
```

```
;процедура 2, увеличивающая значение следующей ячейки
 на 3
+( ;увеличиваем значение текущей ячейки на 1 (было 1).
 Это будет идентификатор процедуры
     > ;сдвигаемся на следующую ячейку
     +++ ;увеличиваем значение ячейки
     < ;возвращаемся назад
) ;конец процедуры

;тело программы
[-] ;очищаем текущую ячейку
++: ;устанавливаем в ячейке 2 и вызываем процедуру с
 этим идентификатором – увеличиваем значение следующей
 ячейки на 3
[-] ;очищаем текущую ячейку
+: ;устанавливаем в ячейке 1 и вызываем процедуру с
 этим идентификатором – очищаем следующие 3 ячейки
```

Задания к выполнению лабораторной работы

Теоретическая часть:

Построить машины Тьюринга, решающие задачи согласно варианту из прилагаемого перечня - одна задача и первой части заданий и одна задача из второй. Для построения машины Тьюринга воспользоваться программным эмулятором машины Тьюринга-Поста.

Варианты заданий – часть первая:

1. На информационной ленте машины Тьюринга содержится массив символов +. Необходимо разработать функциональную схему машины Тьюринга, которая каждый второй символ + заменит на –. Каретка в состоянии q0 находится где-то над указанным массивом.
2. На информационной ленте машины Тьюринга в трех секциях в произвольном порядке записаны 3 цифры 1, 2, 3. Каретка обозревает крайнюю левую цифру. Необходимо составить функциональную схему машины Тьюринга, которая расположит эти цифры в порядке возрастания.
3. Даны два натуральных числа n и m, представленные в унарной системе счисления. Между этими числами стоит знак « ? ». Выяснить отношение m и n, т.е. знак « ? » заменить на один из подходящих знаков « > », « < », « = ».
4. Определить по номеру года: високосный он или нет
5. На информационной ленте машины Тьюринга находится массив, состоящий только из символов A и B. Сжать массив, удалив из него все элементы B.
6. На ленте машины Тьюринга находится десятичное число. Определить делится это число на 5 без остатка. Если делится, то записать справа от числа слово «да», если нет — «нет». Каретка находится где-то над числом.
7. На информационной ленте машины Тьюринга в трех секциях в произвольном порядке записаны 3 различные буквы: A, B и C. Каретка в состоянии q0 обозревает букву, расположенную справа. Необходимо составить функциональную схему машины Тьюринга, которая сумеет поменять местами крайние буквы.
8. Дана строка из букв « a » и « b » . Разработать машину Тьюринга, которая переместит все буквы « a » в левую, а буквы « b » — в правую части строки. Каретка находится над крайним левым символом строки.
9. На ленте машины Тьюринга записано число в десятичной системе счисления. Каретка находится над крайней правой цифрой. Записать цифры этого числа в обратном порядке.

10. Дано число n в восьмеричной системе счисления. Разработать машину Тьюринга, которая увеличивала бы заданное число n на 1.
11. На ленте машины Тьюринга находится число, записанное в десятичной системе счисления. Умножить это число на 2, если каретка находится над крайней левой цифрой числа.
12. Дана десятичная запись натурального числа n>1. Разработать машину Тьюринга, которая уменьшала бы заданное число n на 1. При этом запись числа n–1 не должна содержать левый нуль, например, 100–1=99, а не 099. Начальное положение головки — правое.
13. У каждого слова длины > 3 в алфавите A стереть три последних символа, а слова меньшей длины переработать в пустое слово.
14. Распознать, является ли слово на информационной ленте машины Тьюринга симметричным.
15. Найти для натурального n (в двоичном представлении) значение n+1 (также двоичное).
16. Проверить, дает ли длина слова при делении на 3 остаток 2.
17. Реализовать функцию f(x) = x', где x' — обращение слова x.
18. Реализовать функцию f(x) = xx', где x' — обращение слова x.

Варианты заданий часть вторая:

1. Найти произведение двух натуральных чисел m и n, заданных в унарной системе счисления. Соответствующие наборы символов « | » разделены знаком « * », а справа от последнего символа правого члена стоит знак « = ». Поместить результат умножения этих чисел вслед за знаком « = ».
2. На ленте машины Тьюринга находится массив 2*N меток. Уменьшить этот массив в 2 раза.
3. На информационной ленте машины Тьюринга находится десятичное число. Найти результат целочисленного деления этого числа на 2.
4. На ленте машины Тьюринга находится слово, состоящее из букв латинского алфавита. Подсчитать число букв «a» в данном слове и полученное значение записать на ленту левее исходного слова через пробел. Каретка обозревает крайнюю левую букву.
5. На ленте машины Тьюринга находится целое положительное число, записанное в десятичной системе счисления. Найти произведение этого числа на число 11. Каретка обозревает крайнюю правую цифру числа.
6. Даны два натуральных числа n и m, заданных в унарной системе счисления. Числа n и m представлены наборами символов « | », разделенных « \ ». В конце набора стоит «=». Разработать машину Тьюринга, которая будет производить деление нацело двух натуральных чисел n и m и находить остаток от деления. При этом результат должен быть записан следующим образом: после «=» должен находиться набор символов « | » частного (он может быть и пустым), после чего ставится знак «(», за которым следует набор символов « | » остатка от деления n на m.
7. Дан массив из открывающихся и закрывающихся скобок. Построить машину Тьюринга, которая удаляла бы пары взаимных скобок. Например, дано : «) (( ) (( ) », надо получить : «) . . . ((. ».
8. Даны два целых положительных числа в десятичной системе счисления. Сконструировать машину Тьюринга, которая будет находить разность этих чисел, если известно, что первое число больше второго, а между ними стоит знак –. Каретка находится над левой крайней цифрой левого числа.
9. Даны два целых положительных числа в различных системах счисления, одно — в троичной системе, другое — в десятичной. Разработать машину Тьюринга, которая будет находить сумму этих чисел в десятичной системе счисления.
10. Сконструировать машину Тьюринга, которая выступит в качестве двоично-восьмиричного дешифратора.
11. Дана конечная последовательность меток, записанных в клетки ленты подряд, без пропусков. Необходимо разработать машину Тьюринга, которая будет записывать в десятичной системе счисления число этих меток.
12. Даны два натуральных числа m и n, представленных в унарной системе счисления. Соответствующие наборы символов « | » разделены « – » , вслед за последним символом набора n стоит знак «=». Разработать машину Тьюринга, которая будет находить разность чисел m и n .При этом результат должен быть записан следующим образом : если m>n , то справа от «=» должны стоять знак «+» и набор символов « | » в количестве m–n; если m=n, то

справа от знака «=» должна стоять пустая клетка; если m<n, то справа от «=» должны стоять знак «–» и набор символов « | » в количестве n–m.

13. Построить машину Тьюринга, которая распознает, является ли слово x началом слова y.
14. Построить машину Тьюринга, которая находит наибольшее общее начало слов x и y.

Содержание отчета

Название и цель лабораторной работы

Описание построенной при решений задач теоретической части машины Тьюринга в виде таблицы состояний (либо набора правил) и графа переходов, а также в электронном виде для демонстрации работы на программной эмуляторе. Представить словесное описание алгоритма работы построенной машины. Граф и таблица (набор правил) должны сопровождаться поясняющими комментариями.

Лабораторная работа №2

Формальные теории, продукционные экспертные системы

Теоретические сведения

Теоретический материал, необходимый для выполнения лабораторной работы можно найти в учебном пособии по математической логике (Терехов, Куценко)

Задания к выполнению лабораторной работы

Теоретическая часть:

Решить задачи из лабораторного практикума по математической логике (Хачатрян, Гончарова) согласно своему варианту (см. таблицу).

Варианты заданий:

№	Номера задач			
Занятие	3			4
1	60.2	70.3	73.1	23.1
2	60.3	70.4	73.2	23.2
3	60.4	70.5	73.3	23.3
4	62.1	70.6	73.4	23.4
5	62.2	70.7	73.5	23.5
6	62.3	70.8	73.6	23.6
7	62.5	70.9	73.7	23.7
8	64.1	70.10	73.8	23.8
9	64.2	70.11	73.9	23.9
10	64.3	70.12	73.10	23.10
11	64.4	70.13	73.11	23.11
12	67.2	70.14	73.12	23.12
13	67.3	70.15	73.13	23.13
14	67.4	70.16	73.14	23.14
15	67.5	70.17	73.15	23.15
16	67.6	70.18	73.16	23.16
17	67.7	70.19	73.17	23.17
18	67.8	70.20	73.18	23.18
19	68.1	70.21	73.19	23.19
20	68.2	70.22	73.20	23.20
21	68.3	71.1	73.21	23.22
22	63.1	71.2	73.22	23.23
23	63.2	71.3	73.23	23.24
24	72.1	71.4	73.24	23.25
25	72.2	71.5	73.25	23.26
26	72.3	71.6	73.26	23.27
27	72.4	71.7	73.27	23.1
28	72.5	71.8	73.28	23.6
29	70.1	71.9	73.29	23.7
30	70.2	71.10	73.30	23.8

Практическая часть:

Реализовать на одном из распространенных языков программирования (Pascal, C, C++, можно без графического интерфейса в консольном диалоговом режиме или с использованием файлового ввода/вывода) продукционную систему, позволяющую выполнять прямой вывод — доказывать возможность вывода заданных фактов (цели) на основе правил ЕСЛИ-ТО из базы правил и имеющихся в рабочей памяти фактов.

Если цель не выводима (или не задана), то вывести все возможные выводимые факты.

Если в процессе вывода есть возможность применить сразу несколько правил — применить одно (любое) из них и начать следующую итерацию. Условие остановки итерационного процесса вывода — после появления всех необходимых фактов (цель выводима) или когда ни одно из правил нельзя применить (цель не выводима, получены все возможные выводимые факты).

Факты в рабочей памяти не должны повторяться.

Должна быть предусмотрена возможность просмотра рабочей памяти до и после итеративного процесса вывода новых фактов, а также получение протокола работы системы, описывающего, какие правила в какой последовательности сработали.

Для тестирования системы создать небольшую базу правил (5–7 шт.) на любую неабстрактную тему.

Дополнительные задания повышенной трудности (по выбору студента):

- Осуществить автоматическое расширение базы правил следующим методом: если есть правила $A \Rightarrow B$ и $B \Rightarrow C$, то добавить правило $A \Rightarrow C$, если его нет в базе.
- Добавить следующий режим — если цель не выводится, то необходимо указать, каких фактов в рабочей памяти для этого не хватает (перечислить все альтернативы).
- Реализовать возможность использования правил со сложными условными частями (включающими связки И, ИЛИ, модификатор НЕ, скобки и т.п.).
- Использовать обратный вывод вместо прямого.
- Реализовать систему, осуществляющую доказательство (опровержение) методом резолюций.

Содержание отчета

Название и цель лабораторной работы.
Решение предложенных в теоретической части задач.
Программу на выбранном языке программирования в виде исходных кодов (с поясняющими комментариями) и в электронном варианте для демонстрации на ЭВМ.

Содержимое базы правил.
Содержимое рабочей памяти до и после процесса вывода.
Протокол работы системы