Лабораторная работа № 4

Задача: Вычислить арифметические выражения с использованием одностековой целочисленной вычислительной машины. Вывести на экран вычисленный результат.

Пример для вычисления:

x = 15;
y = 28 - x;
*a1b = 387 + y * (26 + x / 3) – 6;*

Важно: Лабораторная работа выполняется в три фазы.

Фаза 1 – перевод исходного алфавитно-цифрового текста в массив лексем, соответствующий исходному выражению после символа «=».

Фаза 2 – перевод массива лексем, полученного в фазе 1 в массив лексем в виде ПОЛИЗ с помощью МП-автомата.

Фаза 3 – вычисление выражения в виде ПОЛИЗ, представленного массивом лексем, с помощью МП-автомата.

Фаза 1. Перевод исходного алфавитно-цифрового текста в массив лексем, соответствующий исходному выражению после символа « = ».

Представим процесс перевода в виде конечного автомата с обрамлением.

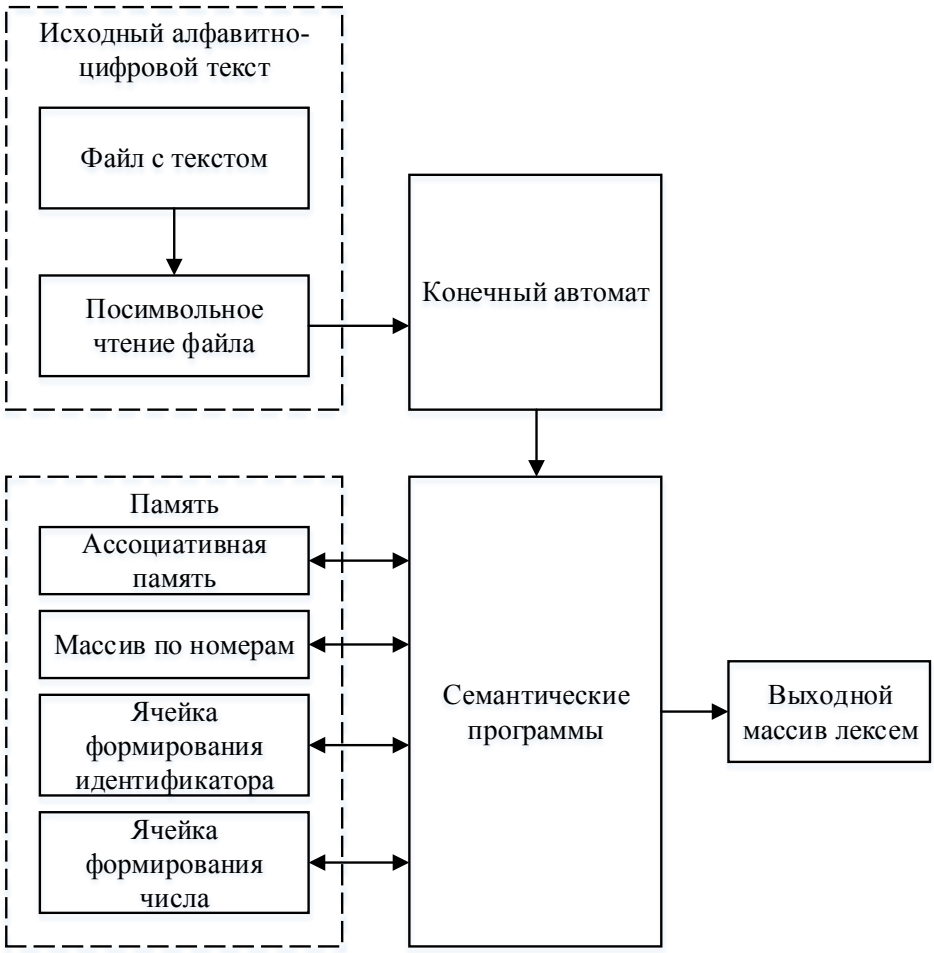

Рис.1 Структурная схема фазы 1.

Лексема (от др.-греч. λέξις – лексис – слово) – машинное слово фиксированной длины. Группы символов исходного текста объединяются в единые синтаксические объекты – лексемы в соответствии с их типом.

Обозначим лексемы длиной 32 бита следующих видов для различных групп синтаксических объектов:

- Лексема типа «0» - «Операнд - число».

Для трансляции допускаются только целочисленные операнды.

Пример. При трансляции числа «15» получим следующую лексему:

Тип лексемы 2 разряда	Значение 30 разрядов
0	15

- Лексема типа «1» - «Операнд - идентификатор».

При трансляции идентификатора семантическая программа помещает имя идентификатора в ассоциативный массив (например, в ассоциативный контейнер типа «map») в качестве ключа, которому присваивается номер, начиная с ноля, в качестве значения. Номер каждого следующего добавленного нового идентификатора увеличивается на единицу по отношению к предыдущему. Если добавляемый идентификатор уже находится в ассоциативном массиве, то повторного добавления его в ассоциативный массив не происходит.

Пример. При трансляции впервые встреченного идентификатора «x1» получим следующую лексему:

Тип лексемы 2 разряда	Номер идентификатора в ассоциативном массиве 30 разрядов
1	0

Впоследствии, когда значение идентификатора будет вычислено, оно будет помещено в одномерный массив под индексом, соответствующим его номеру в ассоциативном массиве. Так, если вычисленное значение идентификатора «x1» будет равно «57», то в одномерный массив оно запишется под индексом «0» (mas[0] = 57).

- Лексема типа «2» - «Операция или разделитель».

При трансляции операции или разделителя в лексему запишется 8-разрядный ASCII код соответствующей операции или разделителя.

Пример. При трансляции операции «+» получим следующую лексему:

Тип лексемы 2 разряда	Не используется 22 разряда	ASCII код 8 разрядов
2	-	0x2B

Задача: Используя конечный автомат, перевести исходный алфавитно-цифровой текст в лексемы. Идентификатор левой части выражения до символа «=» необходимо представить в виде лексемы и занести в ассоциативный массив. В дальнейшем, когда значение идентификатора будет вычислено, необходимо занести его значение в массив по номерам. Правую часть выражения необходимо перевести в массив лексем для дальнейших вычислений в фазах 2 и 3.

Важно: Для выполнения работы необходимо использовать язык программирования C++ и принципы ООП. Для представления лексем необходимо описать соответствующий объект типа «Lexema». Для перевода текста в массив лексем необходимо описать заданный конечный автомат в виде объекта, который также потребуется в фазах 2 и 3. Использование таблицы переходов, таблицы выходов и символов исходного алфавитно-символьного текста в качестве входов обязательно.

Для формирования массива лексем из исходного алфавитно-цифрового текста воспользуемся конечным автоматом.

Конечный автомат построим на основе синтаксической диаграммы представленной на рис. 2.

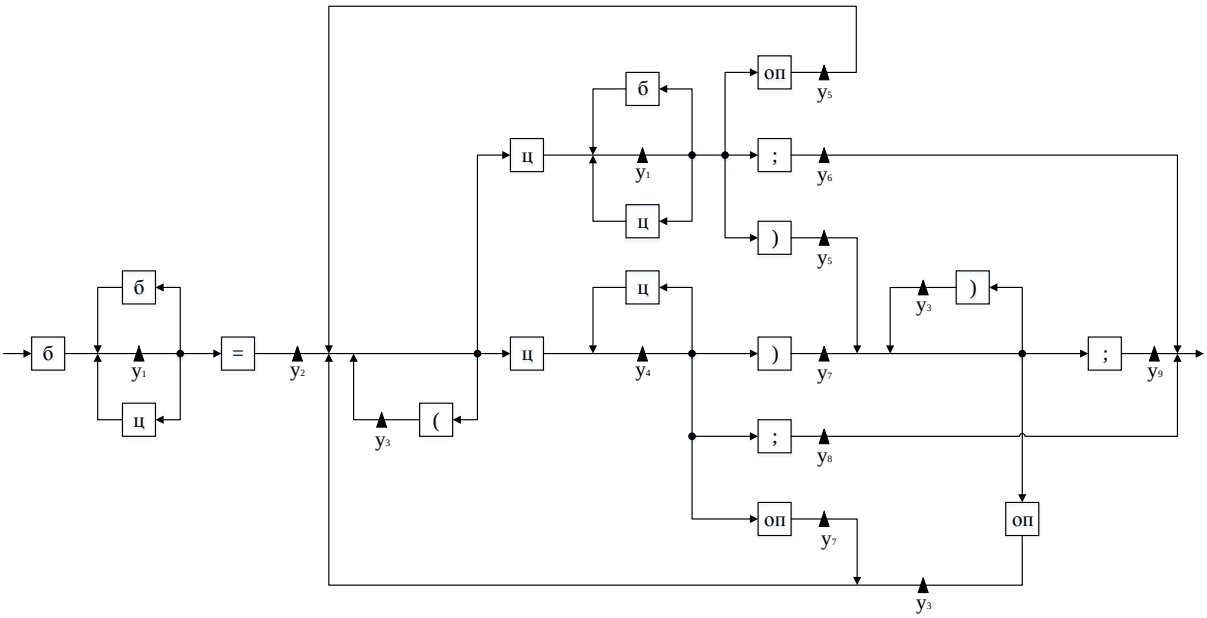

Рис. 2. Синтаксическая диаграмма для преобразования строки алфавитно-цифрового текста в массив лексем.

На рис. 2 применяется условное обозначение «оп», обозначающие символы арифметических операций « + », « - », « * », « / ».

Конечный автомат для преобразования строки алфавитно-цифрового текста в массив лексем, соответствующий синтаксической диаграмме – Рис . 2, представлен таблицами fs и fy.

Таблица переходов fs:

	S0	**S1**	**S2**	**S3**	**S4**	**S5**
б	S1	S1	S3	S3	-	-
ц	-	S1	S4	S3	S4	-
=	-	S2	-	-	-	-
(	-	-	S2	-	-	-
оп	-	-	-	S2	S2	S2
)	-	-	-	S5	S5	S5
;	-	-	-	S0	S0	S0

Таблица выходов fy:

	S0	**S1**	**S2**	**S3**	**S4**	**S5**
б	y1	y1	y1	y1	-	-
ц	-	y1	y4	y1	y4	-
=	-	y2	-	-	-	-
(	-	-	y3	-	-	-
оп	-	-	-	y5	y7	y3
)	-	-	-	y5	y7	y3
;	-	-	-	y6	y8	y9

Описание семантических действий для выходов Y:

y1	Добавляем символы к имени идентификатора в соответствующую ячейку.
y2	Идентификатор прочитан, заносим его в ассоциативный массив и присваиваем номер. Очищаем ячейку для формирования идентификатора.
y3	Создаем лексему типа 2 считанной операции или разделителя, заносим полученную лексему в массив лексем.
y4	Добавляем символы к значению операнда-числа.
y5	Создаем лексему типа 1 считанного операнда-идентификатора, заносим полученную лексему в массив лексем. Создаем лексему типа 2 считанной операции или разделителя, заносим полученную лексему в массив лексем.
y6	Создаем лексему типа 1 считанного операнда-идентификатора, заносим полученную лексему в массив лексем. Создаем лексему типа 2 для символа разделителя « ; », заносим полученную лексему в массив лексем. Символьная строка преобразована в массив лексем, переходим к преобразованию полученного массива в ПОЛИЗ (Фаза 2).
y7	Создаем лексему типа 0 считанного операнда-числа, заносим полученную лексему в массив лексем. Создаем лексему типа 2 для считанной операции или разделителя, заносим полученную лексему в массив лексем.
y8	Создаем лексему типа 0 считанного операнда-числа, заносим полученную лексему в массив лексем. Создаем лексему типа 2 для символа разделителя « ; », заносим полученную лексему в массив лексем. Символьная строка преобразована в массив лексем, переходим к преобразованию полученного массива в ПОЛИЗ (Фаза 2).

y9	Создаем лексему типа 2 для символа разделителя « ; », заносим полученную лексему в массив лексем. Символьная строка преобразована в массив лексем, переходим к преобразованию полученного массива в ПОЛИЗ (Фаза 2).

Фаза 2. Перевод массива лексем, полученного в фазе 1, в массив лексем ПОЛИЗ.

Для перевода массива лексем в ПОЛИЗ воспользуемся МП-автоматом, представленном на рис. 3, который реализует алгоритм Замельсона-Бауэра.

Каждой операции и разделителю присваивается определенный ранг в соответствии со следующей таблицей:

	(	)	+-	*/	;	@
Рл	3	0	1	2	-1	
Рм	-1		1	2		-2

Здесь Рл – ранг символа в исходной строке (записанной на ленту МП-автомата), а Рм – ранг символа, записанного в магазинную память, @ - символ, обозначающий конец данных в магазинной памяти.

Задача: Используя конечный автомат, перевести исходный массив лексем в массив лексем ПОЛИЗ. Массив ПОЛИЗ будет использоваться для дальнейших вычислений в фазе 3.

Важно: Для выполнения работы необходимо использовать язык программирования C++ и принципы ООП. Для перевода исходного массива лексем в массив ПОЛИЗ необходимо описать заданный конечный автомат в виде объекта, необходимо использовать объект автомата из фазы 1. Использование таблицы переходов, таблицы выходов и лексем из массива лексем в качестве входов обязательно. В качестве магазинной памяти необходимо использовать контейнерный класс «стек», в который при его инициализации необходимо записать лексему типа 2 с разделителем « @ ».

Процесс перевода исходного массива лексем (фаза 1) в ПОЛИЗ будем осуществлять с помощью МП-автомата, представленного на Рис. 3. Управляющий конечный автомат для МП-автомата – Рис. 4.

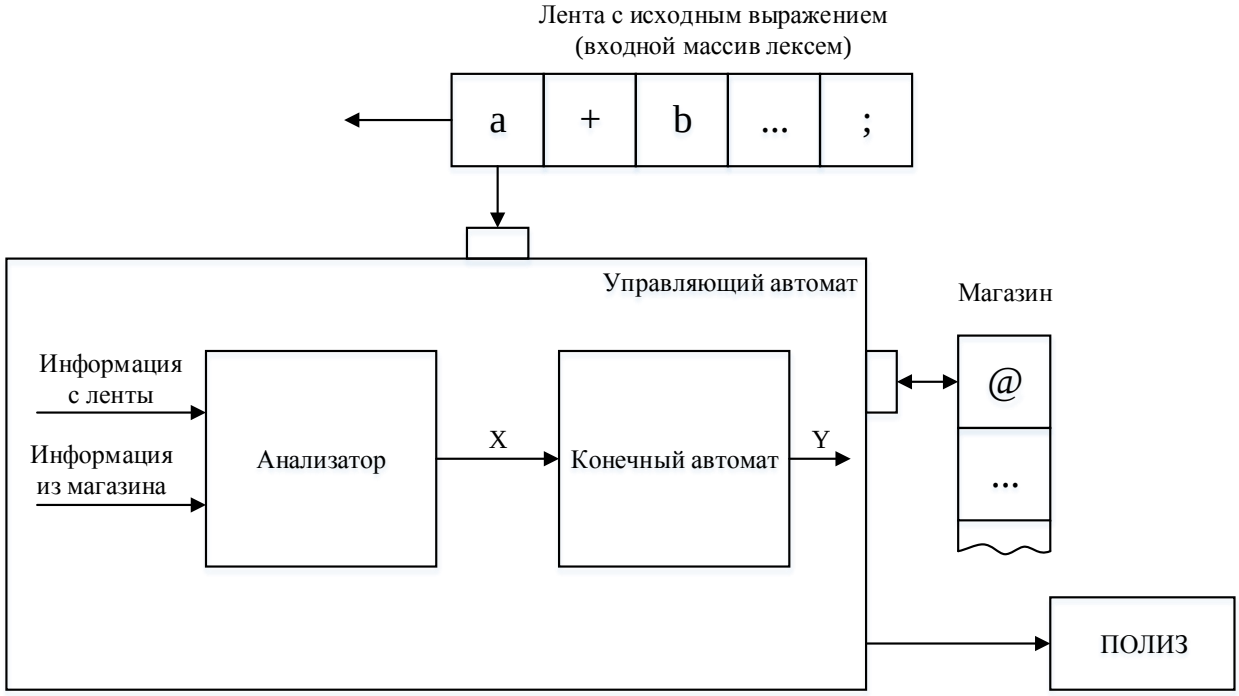

Рис. 3. МП-автомат для перевода выражения в ПОЛИЗ.

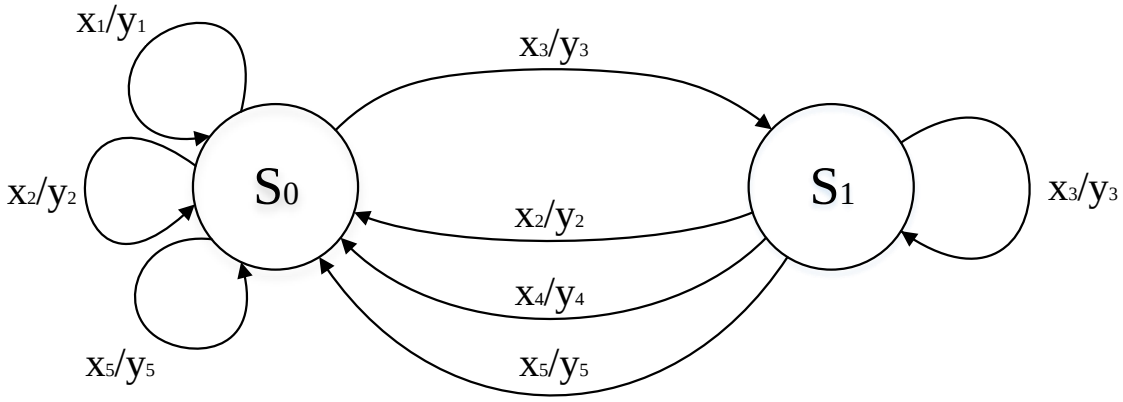

Рис. 4. Управляющий конечный автомат.

Описание входов и выходов:

x1/y1	Из входного массива лексем прочитан операнд – лексема типа 0 или типа 1.	Переписываем текущую лексему из входного массива лексем в массив ПОЛИЗ, переходим к следующей лексеме во входном массиве (сдвиг ленты).
x2/y2	Из входного массива лексем прочитана операция или разделитель – лексема типа 2, кроме «) », « ; », при этом $Р_л > Р_м$.	Переписываем текущую лексему из входного массива лексем в магазинную память, переходим к следующей лексеме во входном массиве (сдвиг ленты).

x3/y3	Из входного массива лексем прочитана операция или разделитель – лексема типа 2, при этом $Р_л \leq Р_м$.	Переписываем текущую лексему из магазинной памяти в массив ПОЛИЗ, удаляем текущую лексему из магазинной памяти.
x4/y4	Из входного массива лексем прочитана операция или разделитель – лексема типа 2 «) », при этом $Р_л > Р_м$.	Удаляем текущую лексему из магазинной памяти. Переходим к следующей лексеме во входном массиве лексем (сдвиг ленты).
x5/y5	Из входного массива лексем прочитана операция или разделитель – лексема типа 2 « ; », при этом $Р_л > Р_м$.	Переписываем текущую лексему из входного массива лексем в массив ПОЛИЗ. Конец преобразования – входной массив лексем преобразован в ПОЛИЗ.

Фаза 3. Вычисление выражения в виде ПОЛИЗ, представленного массивом лексем, с помощью МП-автомата.

Для вычисления воспользуемся МП-автоматом, представленным на рис. 5, который содержит управляющего автомат без памяти (комбинационная схема) с семантическими действиями – рис. 6.

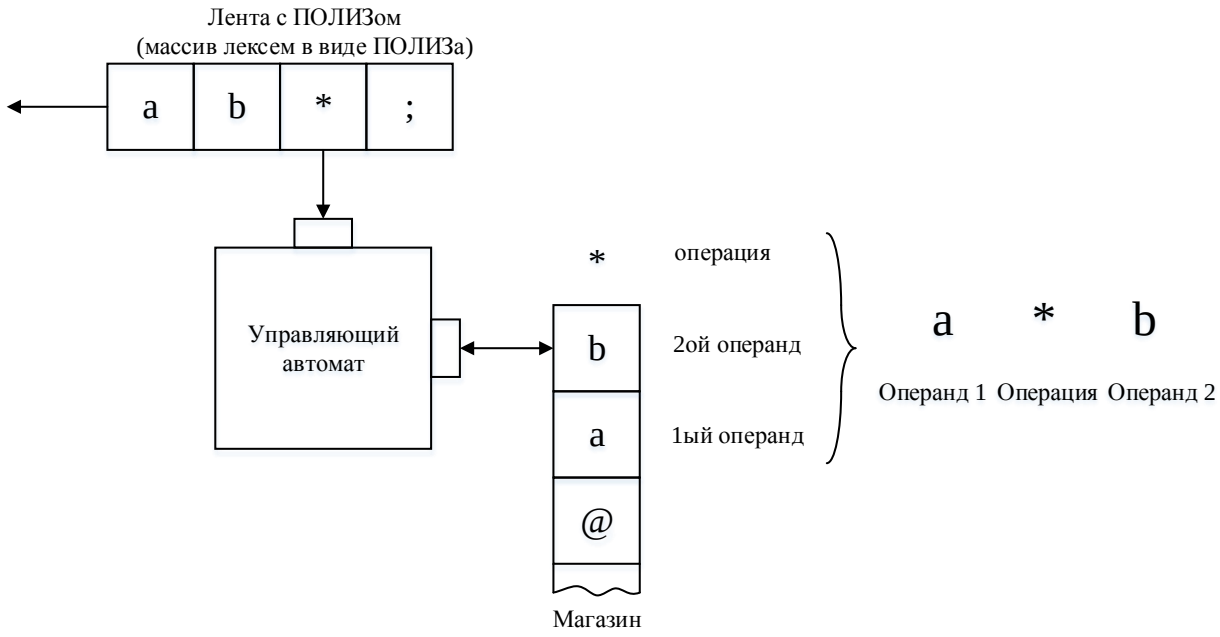

Рис. 5. МП-автомат для вычисления ПОЛИЗ.

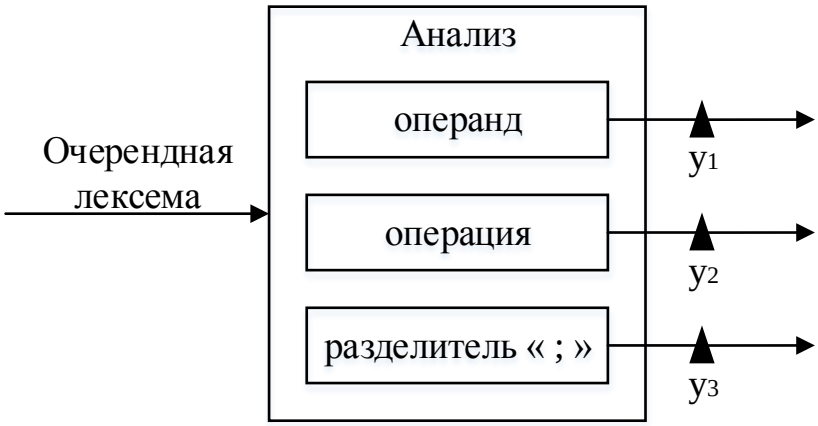

Рис. 6. Управляющий автомат без памяти (комбинационная схема).

Задача: Используя МП-автомат, вычислить выражение, представленное массивом лексем в виде ПОЛИЗ. Результат вычисления присвоить операнду-идентификатору, полученному в фазе 1 до символа «=», и записать в соответствующую ячейку массива по номерам.

Важно: Для выполнения работы необходимо использовать язык программирования C++ и принципы ООП. Необходимо описать автомат без памяти (комбинационную схему) и его семантические действия. В качестве магазинной памяти необходимо использовать контейнерный класс «стек», в который при его инициализации необходимо записать лексему типа 2 с разделителем « @ ».

Описание семантических действий для выходов Y:

y1	Загружаем лексему операнда в магазин.
y2	1. Выгружаем из магазина верхнюю лексему в ячейку «Операнд 2»; 2. Выгружаем из магазина следующую лексему в ячейку «Операнд 1»; 3. Вычисляем <Операнд 1> <Операция> <Операнд 2>, формируем лексему типа 0 операнда-числа, загружаем её в магазин.
y3	Выгружаем из вершины магазина результат вычисления выражения и присваиваем его операнду-идентификатору из фазы 1, стоящему до символа « = ».