Министерство науки и высшего образования Российской Федерации
ФГБОУ ВО «Тверской государственный университет»

Утверждаю:
Руководитель ООП
_____ / А.В. Язенин /
«13» февраля 2020 года

Рабочая программа дисциплины (с аннотацией)

ЯЗЫКИ ПРОГРАММИРОВАНИЯ И МЕТОДЫ ТРАНСЛЯЦИИ

Направление подготовки
02.03.02 ФУНДАМЕНТАЛЬНАЯ ИНФОРМАТИКА
И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Профиль подготовки
Инженерия программного обеспечения

Для студентов 2-го курса

Форма обучения – очная

Составитель:

к.ф.-м.н., доцент И.С. Солдатенко _____

Тверь, 2020

I. Аннотация

1. Цель и задачи дисциплины

Целью освоения дисциплины является:
Освоение теоретических основ и получение практических навыков построения языков программирования и программных средств трансляции для них.

Задачами освоения дисциплины являются:
Систематическое рассмотрение и практическое освоение основных понятий языков программирования, синтаксиса, семантики, формальных способов описания языков программирования, типов данных, способов и механизмов управления данными, методов и основных этапов трансляции, конструкций распределенного и параллельного программирования.

2. Место дисциплины в структуре ООП

Дисциплина относится к части учебного плана, формируемой участниками образовательных отношений, раздел «Дисциплины профиля подготовки».
Дисциплина объединяет знания в области теории формальных языков, формальных грамматик, теории автоматов и методов трансляции. Позволяет на практике применить полученные на дисциплинах «Теоретические основы информатики», «Дискретная математика», «Методы программирования», «Математическая логика и теория алгоритмов» знания для построения работающего транслятора с языка программирования высокого уровня.

Предварительные знания и навыки:
Для успешного освоения дисциплины необходимы знания и навыки, полученные в ходе изучения следующих предшествующих дисциплин: «Теоретические основы информатики», «Дискретная математика», «Практикум на ЭВМ», «Методы программирования».

Дальнейшее использование:
Позволяет лучше разобраться в материале дисциплин «Теория автоматов и формальных языков», «Архитектура ЭВМ», «Операционные системы».

3. Объем дисциплины:

6 зачетных единиц, 216 академических часов, **в том числе**:

- **контактная аудиторная работа:**
  лекции 62 часа, в т.ч. практическая подготовка 31 час; практические занятия 31 час, в т.ч. практическая подготовка 16 часов;
- **контактная внеаудиторная работа**:
  контроль самостоятельной работы и курсовая работа не предусмотрены учебным планом;
- **самостоятельная работа:**
  123 часа, в том числе контроль 74 часа.

4. Планируемые результаты обучения по дисциплине, соотнесенные с планируемыми результатами освоения образовательной программы

Планируемые результаты освоения образовательной программы (формируемые компетенции)	Планируемые результаты обучения по дисциплине	Семестр
Указывается код и наименование компетенции	*Приводятся индикаторы достижения компетенции в соответствии с учебным планом*	
ПК-3 Способен собирать, обрабатывать и интерпретировать экспериментальные данные, необходимые для проектной и производственно- технологической деятельности; разрабатывать новые алгоритмические, методические и технологические решения в конкретной сфере профессиональной деятельности	ПК-3.1 Знает основы проектирования и элементы архитектурных решений информационных систем	3, 4
	ПК-3.2 Применяет в практической деятельности профессиональные стандарты в области информационных технологий, осуществляет алгоритмизацию методов решения прикладных задач ПК-3.3 Имеет практический опыт составления технического задания на разработку информационной системы	4

5. Форма промежуточной аттестации и семестр прохождения:
экзамен и РГР в 3-м и 4-м семестрах.

6. Язык преподавания русский**.**

II. Содержание дисциплины, структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

Учебная программа – наименование разделов и тем	Всего (час.)	Контактная работа (час.)					Самостоятельная работа, в том числе контроль (час.)
		Лекции		Практические занятия		Контроль самостоятельной работы (в том числе курсовая работа)	
		всего	в т.ч. практическая подготовка	всего	в т.ч. практическая подготовка		
3-й семестр							
1. Введение в трансляцию	34	12	6	4	4	0	18
2. Лексический анализ	32	10	5	6	3	0	16
3. Синтаксический анализ	42	8	4	5	2	0	29
ИТОГО за 3-й семестр	108	30	15	15	9	0	63
4-й семестр							
3. Синтаксический анализ	28	10	5	5	2	0	13
4. Генерация кода	48	16	8	5	2	0	27
5. Оптимизация и параллелизм	32	6	3	6	3	0	20
ИТОГО за 4-й семестр	108	32	16	16	7	0	60
ИТОГО	**216**	**62**	**31**	**31**	**16**	**0**	**123**

Подробное тематическое наполнение разделов дисциплины дано в п. VI.

III. Образовательные технологии

Учебная программа – наименование разделов и тем	Вид занятия	Образовательные технологии
1. Введение в трансляцию 2. Лексический анализ 3. Синтаксический анализ 4. Генерация кода 5. Оптимизация и параллелизм	• лекция • практическое занятие	• традиционные (фронтальная лекция, решение упражнений), • цифровые (показ презентаций, выполнение компьютерных лабораторных работ, расчетно-графической работы), • технология проблемного обучения, • групповая работа

IV. Оценочные материалы для проведения текущей и промежуточной аттестации

3-й семестр

№	Результат (индикатор)	Примерная формулировка заданий	Вид/способ	Критерии оценивания
МАТЕРИАЛЫ ДЛЯ ПРОВЕДЕНИЯ ТЕКУЩЕЙ АТТЕСТАЦИИ				
1	ПК-3.1	**Расчетно-графическая работа «Ручная трансляция с MiniC на язык машины 8080»** Самостоятельная лабораторная работа, направленная на изучение исходного языка MiniC, целевого языка ассемблера машины 8080 и основных принципов трансляции. Условие задания приведено в разделе VI.	**вид**: самостоятельная лабораторная работа **способ:** письменно **результаты**: отчет.	**Максимум – 10 б.** **Критерии оценки:** • корректно выполненная и оформленная работа – макс. балл, • отсутствуют комментарии к блокам трансляции – минус 2 балла, • за ошибку трансляции одного оператора исходной программы снимается по 1 баллу.
2	ПК-3.1	**Домашнее задание №1 «Применение регулярных выражений»** Состоит из трех подзаданий – «Моделирование НКА», «Построение ДКА по НКА» и «Распознавание строки по шаблону с помощью ДКА». Условия приведены в разделе VI.	**вид**: самостоятельная лабораторная работа **способ:** на компьютере **результаты**: компьютерная программа.	**Максимум – 8 б.,** из них • «Моделирование НКА» – 3 балла, • «Построение ДКА по НКА» – 3 балла, • «Распознавание строки по шаблону с помощью ДКА» – 2 балла.

				Критерии оценки: максимальный балл ставится за корректно компилируемую программу, успешно проходящую проверочные тесты.
3	ПК-3.1	**Домашнее задание №2 «Лексический анализатор для MiniC»** Первая из трех самостоятельных работ по написанию курсового транслятора. Условие задания приведено в разделе VI.	**вид**: самостоятельная лабораторная работа **способ:** на компьютере **результаты**: компьютерная программа.	**Максимум – 7 б.** **Критерии оценки:** • максимальный балл ставится за корректно компилируемую программу, успешно проходящую проверочные тесты; • отсутствие вывода лексических ошибок – минус 1 балл, • отсутствие вывода таблицы символов – минус 1 балл.
4	ПК-3.1	**Домашнее задание №3 «Преобразование грамматик»** Состоит из двух подзаданий – «Избавление от прямой левой рекурсии в грамматике» и «Левая факторизация грамматики». Условия приведены в разделе VI.	**вид**: самостоятельная лабораторная работа **способ:** на компьютере **результаты**: компьютерная программа.	**Максимум – 8 б.,** из них • «Избавление от прямой левой рекурсии в грамматике» – 4 б., • «Левая факторизация грамматик» – 4 б. **Критерии оценки:** • максимальный балл ставится за корректно компилируемую программу, успешно проходящую проверочные тесты; • отступление от рекомендованной декомпозиции задания на функции – минус 1 б.
5	ПК-3.1	**Домашнее задание №4 «Калькулятор арифметических выражений»** Условие задания приведено в разделе VI.	**вид**: самостоятельная лабораторная работа **способ:** на компьютере **результаты**: компьютерная программа.	**Максимум – 7 б.,** из них • лексический анализатор – 2 балла, • синтаксический анализатор с генерацией ОПЗ – 2 балла, • вычисление ОПЗ – 2 балла, • программа, объеди-

				няющая все три модуля – 1 балл. **Критерии оценки:** максимальный балл ставится за корректно компилируемую программу, успешно проходящую проверочные тесты.
6	ПК-3.1	**Модульная контрольная работа 1** Пример задания приведен в разделе VI.	**вид**: контрольная работа **способ:** письменно **результаты**: выполненные задания	**Максимум – 10 б.** **Критерии оценки:** Контрольная состоит из 9 заданий. Баллы присваиваются за полностью корректно выполненные задания.
7	ПК-3.1	**Модульная контрольная работа 2** Пример задания приведен в разделе VI.		**Максимум – 10 б.** **Критерии оценки:** Контрольная состоит из 10 заданий. Баллы присваиваются за полностью корректно выполненные задания.
МАТЕРИАЛЫ ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ				
8	ПК-3.1	Программа экзамена приведена в разделе VI. **Пример устного вопроса:** Конечный автомат. Задание с помощью таблицы, графа. Регулярные выражения. Детерминированный и недетерминированный автоматы. Преобразование НКА в ДКА. Моделирование НКА. Построение НКА по регулярному выражению. **Пример письменного упражнения:** Построить таблицу АМП для простой грамматики и промоделировать его работу на конкретном слове.	**вид**: традиционный экзамен **способ:** устно/письменно **результаты**: устные ответы и выполненные упражнения	**Максимум – 40 б.** 4 вопроса по 5 баллов каждый и 4 задачи по 5 баллов каждая. **Критерии оценки:** - ответ дан полностью и корректно – максимальный балл; - ответ дан полностью, допущены 1-2 ошибки – 4 балла; - ответ дан частично, но корректно – 2 б.

4-й семестр

№	Результат (индикатор)	Примерная формулировка заданий	Вид/способ	Критерии оценивания
МАТЕРИАЛЫ ДЛЯ ПРОВЕДЕНИЯ ТЕКУЩЕЙ АТТЕСТАЦИИ				

1	ПК-3.1 ПК-3.2 ПК-3.3	**Расчетно-графическая работа «Модификация транслятора»** Самостоятельная лабораторная работа, направленная на более глубокое понимание методов трансляции. Условие задания приведено в разделе VI.	**вид**: самостоятельная лабораторная работа **способ:** на компьютере **результаты**: компьютерная программа.	**Максимум – 5 б.** **Критерии оценки:** корректно выполненная работа – макс. балл.
2	ПК-3.1 ПК-3.2	**Домашнее задание №5 «Построение селективных множеств»** Состоит из двух подзаданий – «Функция FIRST» и «Функция FOLLOW». Условия приведены в разделе VI.	**вид**: самостоятельная лабораторная работа **способ:** на компьютере **результаты**: компьютерная программа.	**Максимум – 6 б.,** из них • «Функция FIRST» – 3 балла, • «Функция FOLLOW» – 3 балла. **Критерии оценки:** • максимальный балл ставится за корректно компилируемую программу, успешно проходящую проверочные тесты; • отступление от рекомендованной декомпозиции задания на функции – минус 1 б.
3	ПК-3.1 ПК-3.2	**Домашнее задание №6 «Автомат с магазинной памятью для LL(1) грамматик»** Состоит из двух подзаданий – «Построение таблицы АМП синтаксического анализатора для LL(1) грамматики» и «Драйвер табличного синтаксического анализатора». Условия приведены в разделе VI.	**вид**: самостоятельная лабораторная работа **способ:** на компьютере **результаты**: компьютерная программа.	**Максимум – 6 б.** из них • «Построение таблицы АМП…» – 3 б., • «Драйвер…» – 3 б. **Критерии оценки:** • максимальный балл ставится за корректно компилируемую программу, успешно проходящую проверочные тесты; • отступление от рекомендованной декомпозиции задания на функции – минус 1 б.
4	ПК-3.1 ПК-3.2	**Домашнее задание №7 «Транслятор с языка MiniC на язык атомов»** Вторая из трех самостоятельных работ по написанию курсового транслятора. Состоит из двух подзаданий – «Построение транслятора выражений на язык атомов» и «Построение полного транслятора с языка MiniC на язык атомов».	**вид**: самостоятельная лабораторная работа **способ:** на компьютере **результаты**: компьютерная программа.	**Максимум – 18 б.,** из них • «Построение транслятора выражений на язык атомов» – 8 б., • «Построение полного транслятора с языка MiniC на язык атомов» – 10 б.

		Условия приведены в разделе VI.		**Критерии оценки:** • максимальный балл ставится за корректно компилируемую программу, успешно проходящую проверочные тесты; • отступление от рекомендованной декомпозиции задания на функции – минус 2 б. • отсутствие вывода лексических ошибок – минус 1 балл, • отсутствие вывода синтаксических ошибок – минус 2 балла, • отсутствие реализации области видимости переменных (второе подзадание) – минус 2 балла.
5	ПК-3.1 ПК-3.2	**Домашнее задание №8 «Генератор кода для машины 8080»** Третья из трех самостоятельных работ по написанию курсового транслятора. Условие задания приведено в разделе VI.	**вид**: самостоятельная лабораторная работа **способ:** на компьютере **результаты**: компьютерная программа.	**Максимум – 9 б.** **Критерии оценки:** • максимальный балл ставится за корректно компилируемую программу, успешно проходящую проверочные тесты; • отступление от рекомендованной декомпозиции задания на функции – минус 2 б. • отсутствие поддержки генерации кода для одного атома – минус 1 балл.
6	ПК-3.1	**Модульная контрольная работа 1** Пример задания приведен в разделе VI.	**вид**: контрольная работа **способ:** письменно **результаты**: выполненные задания	**Максимум – 8 б.** **Критерии оценки:** Контрольная состоит из 6 заданий. Баллы присваиваются за полностью корректно выполненные задания.
7	ПК-3.1	**Модульная контрольная работа 2** Пример задания приведен в разделе VI.		**Максимум – 8 б.** **Критерии оценки:** Контрольная состоит

				из 5 заданий. Баллы присваиваются за полностью корректно выполненные задания.
МАТЕРИАЛЫ ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ				

8	ПК-3.1	Программа экзамена приведена в разделе VI.
		
		Пример устного вопроса:
		Предиктивный синтаксический анализатор.
		
		Пример письменного упражнения:
		Для заданной грамматики
		`S -> Fli | mh`
		`O -> epsilon | vFkj`
		`F -> i | Of`
		постройте LR(0)-автомат.
			вид:
			традиционный экзамен
			способ:
			устно/письменно
			результаты:
			устные ответы и выполненные упражнения
				Максимум – 40 б.
				
				1 вопрос на 5 баллов, 2 задачи по 5 баллов и 5 задач по 6 баллов.
				
				Критерии оценки:
				- ответ дан полностью и корректно – максимальный балл;
				- ответ дан полностью, допущены 1-2 ошибки – 4 балла;
				- ответ дан частично, но корректно – 2 б.

Шкала оценивания выполнения индикаторов:

Индикатор считается выполненным, если либо во время текущей, либо промежуточной аттестации студент набрал как минимум пороговое количество баллов за те виды активности, которые отвечают за данный индикатор. Типовые оценочные материалы с привязкой к отдельным индикаторам приведены в таблице выше.

3-й семестр

№	Индикатор	Текущая аттестация		Промежуточная аттестация (экзамен)	
		Порог	Максимум	Порог	Максимум
1	ПК-3.1	24	60	16	40

4-й семестр

№	Индикатор	Текущая аттестация		Промежуточная аттестация (экзамен)	
		Порог	Максимум	Порог	Максимум
1	ПК-3.1	24	60	16	40
2	ПК-3.2	18	44	–	–
3	ПК-3.3	2	5	–	–

Шкала и критерии выставления оценок за дисциплину:

Шкала и критерии выставления оценок «отлично», «хорошо», «удовлетворительно» и «неудовлетворительно» описаны в локальной нормативной документации Тверского государственного университета (Положение о рейтинговой системе обучения студентов ТвГУ). Положительная оценка может быть выставлена только в том случае, если выполнены все индикаторы.

V. Учебно-методическое и информационное обеспечение дисциплины

1) Рекомендуемая литература

а) Основная литература:

1. Малявко А.А. Формальные языки и компиляторы [Электронный ресурс] / Малявко А.А. - Новосиб.: НГТУ, 2014. - 431 с.: ISBN 978-5-7782-2318-9 — Режим доступа: http://www.iprbookshop.ru/47725.html
2. Назаров, М.В. Введение в программирование больших вычислительных задач на современном Фортране с использованием компиляторов Intel / М.В. Назаров, И.Л. Артемов. - 2-е изд., испр. - М.: Национальный Открытый Университет «ИНТУИТ», 2016. - 260 с.: ил.; То же [Электронный ресурс]. — Режим доступа: http://biblioclub.ru/index.php?page=book&id=428932

б) Дополнительная литература:

1. Гагарина Л.Г. Введение в теорию алгоритмических языков и компиляторов: учеб, пособие / Л.Г. Гагарина, Е.В. Кокорева. — М.: ИД ФОРУМ, 2011. — 176 с.: ил.; 60х90 1/16. — (Ввганее образование), (переплет) ISBN 978-5-8199-0404-6, 1000 экз. — Режим доступа: http://znanium.com/go.php?id=265617
2. Окулов, С.М. Программирование в алгоритмах [Электронный ресурс] — Электрон. дан. — Москва: Издательство "Лаборатория знаний", 2017. — 386 с. — Режим доступа: https://e.lanbook.com/book/94140
3. Гуриков С.Р. Введение в программирование на языке Visual C# : учеб. пособие / С.Р. Гуриков. — М.: ФОРУМ: ИНФРА-М, 2017. — 447 с. — (Высшее образование: Бакалавриат) — Режим доступа: http://znanium.com/go.php?id=752394

2) Программное обеспечение

а) Лицензионное программное обеспечение:

- Microsoft Office профессиональный плюс 2013
- Microsoft Windows 10 Enterprise
- MS Visual Studio Ultimate 2013

б) Свободно распространяемое программное обеспечение

- Google Chrome
- Pyton 3

3) Современные профессиональные базы данных и информационные справочные системы

- **ЭБС «ZNANIUM.COM»** www.znanium.com;
- **ЭБС «IPRBooks»** http://www.iprbookshop.ru;
- **ЭБС «Университетская библиотека онлайн»** https://biblioclub.ru;
- **ЭБС «Лань»** http://e.lanbook.com

4) Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины

- Электронная образовательная среда ТвГУ http://lms.tversu.ru
- Научная библиотека ТвГУ http://library.tversu.ru

VI. Методические материалы для обучающихся по освоению дисциплины

1. Структура рейтинговых баллов

3-й семестр

Название работы	**Кол-во баллов**
ТЕКУЩАЯ АТТЕСТАЦИЯ	
Первый модуль	
Домашнее задание №1.1 «Моделирование НКА»	3
Домашнее задание №1.2 «Построение ДКА по НКА»	3
Домашнее задание №1.3 «Распознавание строки по шаблону с помощью ДКА»	2
Домашнее задание №2 «Лексический анализатор для MiniC»	7
Модульная контрольная 1	10
ИТОГО за первый модуль	**25**
Второй модуль	
Домашнее задание №3.1 «Избавление от прямой левой рекурсии в грамматике»	4
Домашнее задание №3.2 «Левая факторизация грамматики»	4
Домашнее задание №4 «Калькулятор арифметических выражений»	7
Расчетно-графическая работа «Ручная трансляция с MiniC на язык машины 8080»	10
Модульная контрольная 2	10

ИТОГО за второй модуль	**35**
ПРОМЕЖУТОЧНАЯ АТТЕСТАЦИЯ	
Экзамен	40

4-й семестр

Название работы	**Кол-во баллов**
ТЕКУЩАЯ АТТЕСТАЦИЯ	
Первый модуль	
Домашнее задание №5.1 «Функция FIRST»	3
Домашнее задание №5.2 «Функция FOLLOW»	3
Домашнее задание №6.1 «Построение таблицы АМП синтаксического анализатора для LL(1) грамматики»	3
Домашнее задание №6.2 «Драйвер табличного синтаксического анализатора»	3
Модульная контрольная 1	8
ИТОГО за первый модуль	**20**
Второй модуль	
Домашнее задание №7.1 «Построение транслятора выражений на язык атомов»	8
Домашнее задание №7.2 «Построение полного транслятора с языка MiniC на язык атомов»	10
Домашнее задание №8 «Генератор кода для машины 8080»	9
Расчетно-графическая работа «Модификация транслятора»	5
Модульная контрольная 2	8
ИТОГО за второй модуль	**40**
ПРОМЕЖУТОЧНАЯ АТТЕСТАЦИЯ	
Экзамен	40

2. Тематический план разделов учебной программы

Раздел 1. Введение в трансляцию

- Работа с компилятором с точки зрения программиста. IDE. Средства отладки. Знакомство с Visual Studio.

- Препроцессор. Раздельная компиляция. Виды файлов (объектные/ исполняемые модули, библиотеки). Статические/ динамические библиотеки.
- Описание подмножества языка C++ для построения учебного транслятора. Введение в целевой язык ассемблера.
- Введение в трансляцию. Интерпретация и трансляция. Уровни языков. Фазы компиляции. Пример трансляции выражений.

Раздел 2. Лексический анализ

- Задачи лексического анализа, токены, лексемы, идентификаторы, таблица символов. Методы написания ЛА. Формальные языки. Конечные автоматы: определение, способы задания через графы, таблицы. Регулярные выражения. Драйвер КА.
- Описание лексической структуры учебного ЯП. Построение ЛА для учебного ЯП. Генератор лексических анализаторов Lex. Написание собственного лексического анализатора средствами Lex.
- Недетерминированные конечные автоматы. Преобразование НКА в ДКА. Построение НКА из регулярного выражения.

Раздел 3. Синтаксический анализ

- Задачи синтаксического анализа и трансляции. Деревья разбора, атомы. Грамматики, языки, иерархия Хомского. Вывод. Деревья вывода.
- Построение грамматик. Неоднозначные грамматики. Устранение левой рекурсии. Левая факторизация грамматики.
- Типы синтаксического разбора: восходящий и нисходящий. Простая грамматика. Нисходящий анализ методом рекурсивного спуска. Грамматика для выражений.
- Автомат с магазинной памятью, детерминированный и недетерминированный. «NP-полное» моделирование НАМП.
- LL(1) грамматики. Алгоритмы построения функций FIRST и FOLLOW. Алгоритм нахождения селективных множеств.
- Автоматическое построение таблицы АМП для LL(1) грамматики. Драйвер АМП.
- Построение грамматики для учебного ЯП. Преобразование в LL(1) вид. Построение синтаксического анализатора методом рекурсивного спуска для проверки синтаксической корректности.
- Восходящий анализ. Метод «сдвиг-свертка». LR грамматики. Простой LR (SLR). Построение таблицы для SLR.

Раздел 4. Генерация кода

- Транслирующие грамматики. Синтаксически управляемые определения и трансляции. Атрибуты: синтезируемые и наследуемые. S-атрибутные и L-атрибутные грамматики. Пример генерации промежуточного кода по аннотированным деревьям разбора для выражений.

- Генерация промежуточного представления: синтаксические деревья и атомы. Трехадресный код. SSA. Построение транслирующей грамматики для учебного ЯП. Построение транслятора в промежуточное представление для учебного ЯП.
- Генерация кода. Задача распределения регистров.
- Генератор трансляторов Yacc. Написание транслятора выражений средствами Yacc.
- Управление памятью: статическая область кода, данных, стек, куча. Функциональные фреймы. Трансляция вызовов функций. Сборка мусора.

Раздел 5. Оптимизация и параллелизм
- Глобальная оптимизация кода.
- Локальная оптимизация кода.
- Параллелизм на уровне команд. Конвейерная обработка. Программная конвейеризация.

3. Самостоятельная работа

3.1 Домашнее задание №1 «Применение регулярных выражений»

Задание 1.1. Моделирование НКА

Напишите программу, которая принимает на вход регулярное выражение и набор слов. Для каждого слова программа должна выдать один из двух ответов – принадлежит или не принадлежит это слово заданному регулярному языку. Определение принадлежности слова регулярному языку должно происходить путем моделирования работы **недетерминированного** конечного автомата, построенного из регулярного выражения.

Вход/выход: Входные данные вводятся с клавиатуры. Первой запрашивается строка с регулярным выражением. Затем программа спрашивает слово, выводит результат работы НКА на этом слове, спрашивает следующее слово и т.д., пока не будет введена кодовая комбинация, обозначающая конец ввода.

Пример работы программы

```
> 011(0+1)*
> Regexp parsed OK. Enter words:
> 011
> OK
> 0101
> FAIL
> 0
> FAIL
> 011011010110
> OK
> EOF
> Bye!
```

Задание 1.2. Построение ДКА по НКА

Напишите программу, которая строит детерминированный конечный автомат по недетерминированному автомату, построенному из регулярного выражения. Программа должна

принимать на вход регулярное выражение и печатать на экране построенный ДКА в табличном виде.

Вход: Входные данные вводятся с клавиатуры и состоят из одного слова – регулярного выражения.

Выход: Таблица переходов ДКА, построенного по НКА. Формат таблицы может быть любой – главное, чтобы он был читаемым. В примере ниже представлен один из возможных вариантов.

Пример входа

```
(a+b)*abb
```

Пример выхода

```
S    a    b
---------
0    1    2
1    1    3
2    1    2
3    1    4
4    1    2  <-
```

Задание 1.3. Распознавание строки по шаблону с помощью ДКА

Доработайте программу из упр. 1.2 так, чтобы она принимала на вход регулярное выражение и набор слов. Для каждого слова программа должна выдать один из двух ответов – принадлежит ли это слово регулярному языку или нет. Определение принадлежности строки регулярному языку должно происходить путем моделирования работы **детерминированного** конечного автомата, построенного из НКА, который, в свою очередь, построен по регулярному выражению.

Вход/выход: Входные данные вводятся с клавиатуры. Первой запрашивается строка с регулярным выражением. Затем программа спрашивает слово, выводит результат работы ДКА на этом слове, спрашивает следующее слово и т.д., пока не будет введена кодовая комбинация, обозначающая конец ввода.

Пример работы программы

```
> (a+b)*abb
> Regexp parsed OK. Enter words:
> abb
> OK
> abab
> FAIL
> bbb
> FAIL
> abbababababaaaabb
> OK
> EOF

> Bye!
```

3.2 Домашнее задание №2 «Лексический анализатор для MiniC»

Напишите класс лексического анализатора Scanner для языка MiniC на основе конечного автомата, построенного на лекциях. Лексический анализатор должен читать программу на языке MiniC из потока ввода. Для демонстрации работы анализатора напишите программу, которая напечатает список лексем для тестовой программы, а также таблицу символов и строковых констант.

Вход/выход: Входные данные берутся из потока ввода, который передается в конструктор ЛА. Если лексический анализатор встретил лексическую ошибку, необходимо вывести сообщение об этом. Возможные типы лексических ошибок:

- неподдерживаемый языком символ (за исключением случаев, когда этот символ является частью строковой или символьной константы);
- отсутствие разделителя между символами операций;
- одиночный символ | или одиночный символ &;
- пустая символьная константа;
- символьная константа, содержащая более одного символа.

Пример работы программы

Содержимое сканируемого файла	Вывод лексического анализатора

`char str[] = "Hello, world!";`
`int a = 5;`
`char c = 'c';`
`a = a + c;`
	`[kwchar]`
	`[id, 0]`
	`[lbracket]`
	`[rbracket]`
	`[opassign]`
	`[str, 0]`
	`[semicolon]`
	`[kwint]`
	`[id, 1]`
	`[opassign]`
	`[num, 5]`
	`[semicolon]`
	`[kwchar]`
	`[id, 2]`
	`[opassign]`
	`[chr, 'c']`
	`[semicolon]`
	`[id, 1]`
	`[opassing]`
	`[id, 1]`
	`[opplus]`
	`[id, 2]`
	`[semicolon]`
	
	`SYMBOL TABLE:`
	`0 str`
	`1 a`
	`2 c`
	
	`STRING TABLE:`
	`0 Hello, world!`

3.3 Домашнее задание №3 «Преобразование грамматик»

Задание 3.1 Избавление от прямой левой рекурсии в грамматике

Напишите программу, которая принимает на вход КС-грамматику без циклов и ε-правил, избавляется от прямой левой рекурсии в ней и печатает результат на экран.

Вход/выход: Входные данные могут браться из файла или быть заданными вручную. Порядок вывода на экран правил новой грамматики значения не имеет.

Пример работы программы

Содержимое файла	Вывод на экран

```
E -> E + T
E -> E - T
E -> T
T -> T * F
T -> T / F
T -> F
F -> ( E )
F -> id
F -> num
```

или содержимое входной переменной

```
G = {"E": [["E","+","T"], ["E","-","T"], ["T"]],
     "T": [["T","*","F"], ["T","/","F"], ["F"]],
     "F": [["(","E",")"], ["id"], ["num"]]}
```

```
E -> T E'
E' -> + T E'
E' -> - T E'
E' -> e
T -> F T'
T' -> * F T'
T' -> / F T'
T' -> e
F -> ( E )
F -> id
F -> num
```

Задание 3.2 Левая факторизация грамматики

Напишите программу, которая принимает на вход КС-грамматику, проводит левую факторизацию (избавляется от общих префиксов в правилах для одного и того же нетерминала) и печатает результат на экран.

Вход/выход: Входные данные могут браться из файла или быть заданными вручную. Порядок вывода на экран правил новой грамматики значения не имеет.

Пример работы программы

Содержимое файла

```
E -> E + T
E -> E - T
E -> T
T -> T * F
T -> T / F
T -> F
F -> ( E )
F -> id
F -> num

stmt -> if expr then stmt else stmt
stmt -> if expr then stmt
```

или содержимое входных переменных

```
G = {"E": [["E","+","T"], ["E","-","T"], ["T"]],
     "T": [["T","*","F"], ["T","/","F"], ["F"]],
     "F": [["(","E",")"], ["id"], ["num"]]}

G2 = {"stmt": [["if", "expr", "then", "stmt", "else", "stmt"], ["if", "expr", "then", "stmt"]]
```

Вывод на экран

```
E -> T
E -> E E'
E' -> + T
E' -> - T
T -> F
T -> T T'
T' -> * F
T' -> / F
F -> ( E )
F -> id
F -> num

stmt' -> else stmt
stmt' -> e
stmt -> if expr then stmt stmt'
```


3.4 Домашнее задание №4 «Калькулятор арифметических выражений»

Напишите программу, которая принимает на вход арифметическое выражение, состоящее из числовых констант, операций сложения, вычитания, умножения, деления и скобок, и выводит результат его вычисления, либо сообщение об ошибке, если выражение имеет некорректный синтаксис.

Вход/выход: Выражение для вычисления вводится с клавиатуры, результат печатается на экран.

Пример работы программы

Ввод	Вывод на экран
`(23 + 5)/2 - 45 * (12 + (6*3 - 2)/2)`	`-886`
`(12 + 7) / 2 - 13 * 2)`	`Error!`
`26++1`	`Error!`
`(((2 + 3) * 2) - 10)`	`0`

3.5 Домашнее задание №5 «Построение селективных множеств»

Задание 5.1 Функция FIRST

Пусть дана грамматика G. Напишите функцию FIRST(α), где α – любая последовательность терминалов и нетерминалов грамматики G, которая возвращает множество терминалов, с которых начинаются строки, выводимые из α. Если из α выводимо ε, то ε тоже входит в FIRST(α).

Вход/выход: Грамматика считывается из файла, выражения для функции FIRST вводятся с клавиатуры.

Пример работы программы

Грамматика в файле grammar.txt	Вывод на экран

`E -> T E'`
`E' -> + T E'`
`E' -> - T E'`
`E' -> e`
`T -> F T'`
`T' -> * F T'`
`T' -> / F T'`
`T' -> e`
`F -> ( E )`
`F -> id`
	`[(, id]`
	`[+, -, e]`
	`[(, id]`
	`[*, /, e]`
	`[(, id]`
	`[)]`
	`[id]`

Вызовы функции FIRST	

`G.FIRST("E");`
`G.FIRST("E'");`
`G.FIRST("T");`
`G.FIRST("T'");`
`G.FIRST("F");`
`G.FIRST(")");`
`G.FIRST("id");`	

Задание 5.2 Функция FOLLOW

Пусть дана грамматика G. Напишите функцию FOLLOW(A), где A – произвольный нетерминал грамматики G, возвращающую множество терминалов, которые могут располагаться непосредственно после (справа от) A в некоторой сентенциальной форме. Если A окажется крайним справа символом в некоторой сентенциальной форме, то в FOLLOW(A) также входит $ (маркер конца входного потока).

Вход/выход: Грамматика считывается из файла, выражения для функции FOLLOW вводятся с клавиатуры.

Пример работы программы

Грамматика в файле grammar.txt

```
E -> T E'
E' -> + T E'
E' -> - T E'
E' -> e
T -> F T'
T' -> * F T'
T' -> / F T'
T' -> e
F -> ( E )
F -> id
```

Вызовы функции FOLLOW

```
G.FOLLOW("E");
G.FOLLOW("E'");
G.FOLLOW("T");
G.FOLLOW("T'");
G.FOLLOW("F");
```

Вывод на экран

```
[')', '$']
[')', '$']
['+', '-', ')', '$']
['+', '-', ')', '$']
['+', '-', '*', '/', ')', '$']
```

3.6 Домашнее задание №6 «Автомат с магазинной памятью для LL(1) грамматик»

Задание 6.1 Построение таблицы АМП синтаксического анализатора

Пусть дана грамматика G. Напишите функцию, которая строит таблицу автомата с магазинной памятью нисходящего предиктивного синтаксического анализатора.

Вход/выход: Грамматика считывается из файла.

Пример работы программы

Грамматика в файле grammar.txt

```
E -> T E'
E' -> + T E'
E' -> e
T -> F T'
T' -> * F T'
T' -> e
F -> ( E )
F -> id
```

Вызовы функции FIRST

```
G.buildSATable();
G.printSATable();
```

Вывод на экран (пример)

```
1. E -> T E'
2. E' -> + T E'
3. E' -> e
4. T -> F T'
5. T' -> * F T'
6. T' -> e
7. F -> ( E )
8. F -> id

      id    +     *     (     )     $
-------------------------------------
E     R1,R                R1,R
E'          R2,R                R3,R  R3,R
```


```
T     R4,R                R4,R
T'          R6,R  R5,R          R6,R  R6,R
F     R8,R                R7,R
id    P,A
+           P,A
*                 P,A
(                       P,A
)                             P,A
$                                   Accept
```

Задание 6.2 Драйвер табличного синтаксического анализатора

Напишите драйвер нисходящего синтаксического анализатора, который для заданной грамматики по построенной таблице и данной последовательности лексем вернет истину, если последовательность лексем принадлежит языку, задаваемому грамматикой, и ложь в противном случае.

Вход/выход: Грамматика считывается из файла, последовательность лексем (терминалов) вводится с клавиатуры.

Пример работы программы

Грамматика в файле grammar.txt	Вывод на экран
E -> T E' E' -> + T E' E' -> - T E' E' -> e T -> F T' T' -> * F T' T' -> / F T' T' -> e F -> (E) F -> id	True False False True
Ввод с клавиатуры	
(id + id) * id (id + id id * * id id * id + id	

3.7 Домашнее задание №7 «Транслятор с языка MiniC на язык атомов»

Задание 7.1 Построение транслятора выражений на язык атомов

Для транслирующей грамматики выражений языка MiniC построить транслятор на язык атомов методом рекурсивного спуска. После списка атомов ваша программа должна печатать таблицу символов с вашими и временными переменными. Транслятор должен использовать лексический анализатор языка MiniC.

В этом упражнении **не требуется** реализовывать вызовы функций, поэтому атомы CALL, RET и PARAM, а также соответствующие правила грамматики (№30, 32-35) делать пока не нужно. Помимо этого, есть небольшие расхождения с грамматикой, которые необходимо учитывать, а именно: функция alloc() пока не принимает никаких параметров, а вместо функции checkVar() вы будете использовать упрощенную функцию add.

Вход/выход: Входные данные считываются из файла, список атомов записывается в результирующий файл.

Пример работы программы

Выражение в файле prog.minic	Вывод в файл prog.atom

b*b - 4*a*c	(MUL, 0, 0, 3) (MUL, '4', 1, 4) (MUL, 4, 2, 5) (SUB, 3, 5, 6) SYMBOL TABLE 0 b 1 a 2 c 3 temp1 4 temp2 5 temp3 6 temp4

Задание 7.2 Построение полного транслятора с языка MiniC на язык атомов

Для транслирующей грамматики языка MiniC построить транслятор на язык атомов методом рекурсивного спуска. После списка атомов ваша программа должна печатать таблицу символов с вашими и временными переменными, а также таблицу строковых констант. Транслятор должен использовать лексический анализатор языка MiniC.

Вход/выход: Входные данные считываются из файла, список атомов записывается в результирующий файл.

Пример работы программы

Программа в файле prog.minic

```
int sqRoots(int x, int y, int z){
     int result;
     result = y*y - 4*x*z;
     if (result < 0){
       out "No real roots\n";
     } else {
       if (result == 0)
          out "One root\n";
       else
          out "Two roots\n";
     }
     return result;
}

int main(){
  int a, b, c, d;
  in a;
  in b;
  in c;
  d = sqRoots(a, b, c);
  return 0;
}
```

Программа в файле prog.atom

```
Syntax OK
0       (MUL, 2, 2, 5)
0       (MUL, '4', 1, 7)
0       (MUL, 7, 3, 8)
0       (SUB, 5, 8, 6)
0       (MOV, 6, , 4)
```

```
0       (MOV, '1', , 9)
0       (LT, 4, '0', L2)
0       (MOV, '0', , 9)
0       (LBL, , , L2)
0       (EQ, 9, '0', L0)
0       (OUT, , , S0)
0       (JMP, , , L1)
0       (LBL, , , L0)
0       (MOV, '1', , 10)
0       (EQ, 4, '0', L5)
0       (MOV, '0', , 10)
0       (LBL, , , L5)
0       (EQ, 10, '0', L3)
0       (OUT, , , S1)
0       (JMP, , , L4)
0       (LBL, , , L3)
0       (OUT, , , S2)
0       (LBL, , , L4)
0       (LBL, , , L1)
0       (RET, , , 4)
0       (RET, , , '0')
11      (IN, , , 12)
11      (IN, , , 13)
11      (IN, , , 14)
11      (PARAM, , , 12)
11      (PARAM, , , 13)
11      (PARAM, , , 14)
11      (CALL, 0, , 16)
11      (MOV, 16, , 15)
11      (RET, , , '0')
11      (RET, , , '0')
SYMBOL TABLE
-----------------------------------------------------
code    name    kind    type    len     init    scope   offset
0       sqRoots func    int     3       0       -1      -1
1       x       var     int     None    0       0       -1
2       y       var     int     None    0       0       -1
3       z       var     int     None    0       0       -1
4       result  var     int     None    0       0       -1
5       [tmp1]  var     int     None    0       0       -1
6       [tmp2]  var     int     None    0       0       -1
7       [tmp3]  var     int     None    0       0       -1
8       [tmp4]  var     int     None    0       0       -1
9       [tmp5]  var     int     None    0       0       -1
10      [tmp6]  var     int     None    0       0       -1
11      main    func    int     0       0       -1      -1
12      a       var     int     None    0       11      -1
13      b       var     int     None    0       11      -1
14      c       var     int     None    0       11      -1
15      d       var     int     None    0       11      -1
16      [tmp7]  var     int     None    0       11      -1
STRING TABLE
------------
```

```
0       No real roots\n
1       One root\n
2       Two roots\n
```

3.8 Домашнее задание №8 «Генератор кода для машины 8080»

Дополните ваш транслятор заключительным фрагментом – генератором кода, который по списку атомов выдает код для машины 8080.

Вход/выход: Входные данные (программа на языке MiniC) считываются из файла, ассемблерный код на языке машины 8080 записывается в результирующий файл. Ваш транслятор должен вывести: список атомов, таблицу символов, таблицу строк и результирующий код на языке ассемблера 8080.

Пример работы программы

Обратите внимание: в примере вывода ниже ассемблерный код записан в три колонки. Это сделано для экономии места. Ваш транслятор может так не делать.

Программа в файле prog.minic

```
int sqRoots(int x, int y, int z){
      int result;
      result = y*y - 4*x*z;
      if (result < 0){
        out "No real roots";
      } else {
        if (result == 0)
            out "One root";
        else
            out "Two roots";
      }
      return result;
}

int main(){
      int a, b, c, d;
      in a;
      in b;
      in c;
      d = sqRoots(a, b, c);
      return 0;
}
```

Вывод в файл prog.asm

```
Syntax OK

ATOMS
-------------------------------------------------------
0       (MUL, 2, 2, 5)
0       (MUL, '4', 1, 7)
0       (MUL, 7, 3, 8)
0       (SUB, 5, 8, 6)
0       (MOV, 6, , 4)
0       (MOV, '1', , 9)
0       (LT, 4, '0', L2)
0       (MOV, '0', , 9)
0       (LBL, , , L2)
0       (EQ, 9, '0', L0)
0       (OUT, , , S0)
0       (JMP, , , L1)
0       (LBL, , , L0)
0       (MOV, '1', , 10)
0       (EQ, 4, '0', L5)
0       (MOV, '0', , 10)
0       (LBL, , , L5)
0       (EQ, 10, '0', L3)
```

```
0       (OUT, , , S1)
0       (JMP, , , L4)
0       (LBL, , , L3)
0       (OUT, , , S2)
0       (LBL, , , L4)
0       (LBL, , , L1)
0       (RET, , , 4)
0       (RET, , , '0')
11      (IN, , , 12)
11      (IN, , , 13)
11      (IN, , , 14)
11      (PARAM, , , 12)
11      (PARAM, , , 13)
11      (PARAM, , , 14)
11      (CALL, 0, , 16)
11      (MOV, 16, , 15)
11      (RET, , , '0')
11      (RET, , , '0')

SYMBOL TABLE
-----------------------------------------------------
code    name    kind    type    len     init    scope   offset
0       sqRoots func    int     3       0       -1      -1
1       x       var     int     None    0       0       20
2       y       var     int     None    0       0       18
3       z       var     int     None    0       0       16
4       result  var     int     None    0       0       12
5       [tmp1]  var     int     None    0       0       10
6       [tmp2]  var     int     None    0       0       8
7       [tmp3]  var     int     None    0       0       6
8       [tmp4]  var     int     None    0       0       4
9       [tmp5]  var     int     None    0       0       2
10      [tmp6]  var     int     None    0       0       0
11      main    func    int     0       0       -1      -1
12      a       var     int     None    0       11      8
13      b       var     int     None    0       11      6
14      c       var     int     None    0       11      4
15      d       var     int     None    0       11      2
16      [tmp7]  var     int     None    0       11      0

STRING TABLE
-----------------------------------------------------
0       No real roots\n
1       One root\n
2       Two roots\n

ASM 8080 code
-----------------------------------------------------
```

```
ORG 8000H
str0: DB 'No real roots', 0
str1: DB 'One root', 0
str2: DB 'Two roots', 0
ORG 0
LXI H, 0
SPHL
CALL main
END
@MULT:
; Code for MULT library function
@PRINT:
; Code for PRINT library func-
tion

sqRoots:
```

```
      ; (MOV, '0', , 9)
MVI A, 0
LXI H, 2
DAD SP
MOV M, A
      ; (LBL, , , L2)
LBL2:
      ; (EQ, 9, '0', L0)
MVI A, 0
MOV B, A
LXI H, 2
DAD SP
MOV A, M
CMP B
JZ LBL0
```

```
main:
LXI B, 0
PUSH B
PUSH B
PUSH B
PUSH B
PUSH B
      ; (IN, , , 12)
IN 0
LXI H, 8
DAD SP
MOV M, A
      ; (IN, , , 13)
IN 0
LXI H, 6
```

```
LXI B, 0
PUSH B
PUSH B
PUSH B
PUSH B
PUSH B
PUSH B
PUSH B
      ; (MUL, 2, 2, 5)
LXI H, 18
DAD SP
MOV A, M
MOV D, A
LXI H, 18
DAD SP
MOV A, M
CALL @MULT
MOV A, C
LXI H, 10
DAD SP
MOV M, A
      ; (MUL, '4', 1, 7)
LXI H, 20
DAD SP
MOV A, M
MOV D, A
MVI A, 4
CALL @MULT
MOV A, C
LXI H, 6
DAD SP
MOV M, A
      ; (MUL, 7, 3, 8)
LXI H, 16
DAD SP
MOV A, M
MOV D, A
LXI H, 6
DAD SP
MOV A, M
CALL @MULT
MOV A, C
LXI H, 4
DAD SP
MOV M, A
      ; (SUB, 5, 8, 6)
LXI H, 4
DAD SP
MOV A, M
MOV B, A
LXI H, 10
DAD SP
MOV A, M
SUB B
LXI H, 8
DAD SP
MOV M, A
      ; (MOV, 6, , 4)
LXI H, 8
DAD SP
MOV A, M
LXI H, 12
DAD SP
MOV M, A
      ; (OUT, , , S0)
LXI A, str0
CALL @print
      ; (JMP, , , L1)
JMP LBL1
      ; (LBL, , , L0)
LBL0:
      ; (MOV, '1', , 10)
MVI A, 1
LXI H, 0
DAD SP
MOV M, A
      ; (EQ, 4, '0', L5)
MVI A, 0
MOV B, A
LXI H, 12
DAD SP
MOV A, M
CMP B
JZ LBL5
      ; (MOV, '0', , 10)
MVI A, 0
LXI H, 0
DAD SP
MOV M, A
      ; (LBL, , , L5)
LBL5:
      ; (EQ, 10, '0', L3)
MVI A, 0
MOV B, A
LXI H, 0
DAD SP
MOV A, M
CMP B
JZ LBL3
      ; (OUT, , , S1)
LXI A, str1
CALL @print
      ; (JMP, , , L4)
JMP LBL4
      ; (LBL, , , L3)
LBL3:
      ; (OUT, , , S2)
LXI A, str2
CALL @print
      ; (LBL, , , L4)
LBL4:
      ; (LBL, , , L1)
LBL1:
      ; (RET, , , 4)
LXI H, 12
DAD SP
MOV A, M
LXI H, 22
DAD SP
MOV M, A
POP B
POP B
POP B
POP B
POP B
POP B
POP B
RET
DAD SP
MOV M, A
      ; (IN, , , 14)
IN 0
LXI H, 4
DAD SP
MOV M, A
      ; (CALL, 0, ,
16)
PUSH B
PUSH D
PUSH H
PUSH PSW
LXI B, 0
PUSH B
LXI B, 0
LXI H, 4
DAD SP
MOV A, M
MOV C, A
PUSH B
LXI B, 0
LXI H, 6
DAD SP
MOV A, M
MOV C, A
PUSH B
LXI B, 0
LXI H, 8
DAD SP
MOV A, M
MOV C, A
PUSH B
CALL sqRoots
POP B
POP B
POP B
POP B
MOV A, B
LXI H, 0
DAD SP
MOV M, A
POP PSW
POP H
POP D
POP B
      ; (MOV, 16, ,
15)
LXI H, 0
DAD SP
MOV A, M
LXI H, 2
DAD SP
MOV M, A
      ; (RET, , ,
'0')
MVI A, 0
LXI H, 12
DAD SP
MOV M, A
POP B
POP B
POP B
POP B
```

```
      ; (MOV, '1', , 9)
MVI A, 1
LXI H, 2
DAD SP
MOV M, A
      ; (LT, 4, '0', L2)
MVI A, 0
MOV B, A
LXI H, 12
DAD SP
MOV A, M
CMP B
JM LBL2
```

```
      ; (RET, , , '0')
MVI A, 0
LXI H, 22
DAD SP
MOV M, A
POP B
POP B
POP B
POP B
POP B
POP B
POP B
RET
```

```
POP B
RET
      ;   (RET,   ,   ,
'0')
MVI A, 0
LXI H, 12
DAD SP
MOV M, A
POP B
POP B
POP B
POP B
POP B
RET
```

4. Расчетно-графическая работа

4.1 Пример задания для РГР на 3-й семестр

Выполните ручную трансляцию программы, решающей указанную ниже задачу, с языка MiniC на язык машины 8080.

Дан набор ненулевых целых чисел; признак его завершения — число 0. Вывести сумму всех положительных четных чисел из данного набора. Если требуемые числа в наборе отсутствуют, то вывести 0.

Необходимые элементы отчета к РГР:

- Титульный лист
- Условие задачи
- Листинг программы на **языке MiniC**, решающей поставленную задачу
- Листинг результата трансляции на язык машины 8080 **с комментариями**

Код задачи оформить в виде **функции**, вызываемой из main. Все входные параметры задачи **вводятся** с клавиатуры в **функции main**, а затем передаются в функцию в аргументах.

Код машины 8080 должен быть трансляцией кода на языке MiniC.

4.2 Пример задания для РГР на 4-й семестр

Добавить в разработанный вами транслятор одну из указанных модификаций. Результат оформить в виде отчета, одним из элементов которого должно быть техническое задание на внесение изменений в транслятор с указанием того, какие изменения в какие части транслятора будут внесены.

Перечень изменений

1. Добавление однострочных комментариев //
2. Добавление многострочных комментариев /* */
3. Добавление директив препроцессора #define, #undef, #ifdef, #else, #endif
4. Добавление оператора взятия остатка от деления %

5. Добавление операции сравнения >=
6. Добавление оператора break в цикл for
7. Добавление оператора break в цикл while
8. Добавление оператора continue в цикл for
9. Добавление оператора continue в цикл while
10. Добавить операторы побитовых сдвигов влево << и вправо >>
11. Добавить оператор префиксного декремента --a
12. Добавить оператор постфиксного декремента a--
13. Добавление цикла **do** [*тело цикла*] **while**(*условие*)
14. Добавление диагностических предупреждений во время компиляции при неявном преобразовании типа int в тип char и операции явного преобразования типов static_cast<тип> (имя)
15. Добавление операции унарного минуса
16. Реализовать ветку default в операторе switch
17. Добавление сокращенного присваивания a += b
18. Реализовать поддержку ключевого слова const и символьных констант
19. Реализовать операцию деления
20. Добавление двумерных массивов
21. Добавление меток и оператора goto

5. Примеры модульных контрольных работ

5.1 Пример первой модульной контрольной 3-го семестра

1. Выпишите лексемы для следующего фрагмента кода на MiniC (0.8 балла):

```
int a, b; in(); int(b); a = a*a + -2*a*b + b*b; out(a;
```

2. Постройте граф переходов конечного автомата для следующего языка из алфавита {0, 1} (0.8 балла):

```
Строки, содержащие три подряд идущие единицы или два подряд идущих нуля. Общее количество нулей и единиц в этих строках может быть любым.
```

3. Постройте регулярное выражение для следующего языка из алфавита {0, 1} (1.2 балла):

```
Строки, начинающиеся и заканчивающиеся единицей, между которыми следует произвольное количество нулей.
```

4. Постройте регулярное выражение по следующему автомату (1.2 балла):

	a	b
*A	A	B
*B	C	B
C	C	C

5. Опишите в словесной форме язык, заданный КА (1.2 балла):

	a	b
*A	A	B
*B	C	B
C	C	C

6. Вычеркните слова, не входящие в регулярный язык `(0+11)*0` (0.4 балла):
   `0, 1, 111, 0010, 01010, 1010, 111000, 1111000, 11110, 0000, 1111`

7. Постройте НКА по регулярному выражению (0.8 балла): `(0+11)*0`

8. Детерминизируйте НКА, построенный в упражнении 7 (2 балла).

9. Проэмулируйте работу НКА, построенного в упражнении 7, на следующем слове (1.6 балла): `011110`

5.2 Пример второй модульной контрольной 3-го семестра

1. Классифицируйте каждое из правил по Хомскому **(0.8 балла)**:
   - A B -> B A
   - S a b c d e -> S
   - a S b -> a A c B b
   - a b c -> A B C
   - A B C -> A d C
   - A -> A
   - A -> epsilon
   - A -> a
   - a w B g -> a B w g
   - A b C -> A a C
2. Классифицируйте каждую из приведенных ниже грамматик по Хомскому **(0.8 балла)**
   - A -> b A
   - a -> A B C
   - A B C -> A d C

   - A x B -> A x S E d B
   - b -> b A
   - A x B -> A S E d B
   - a S A -> a S b b
3. Для указанной ниже грамматики и строки `aa+(a**)` приведите: левый вывод, правый вывод, дерево вывода **(0.8 балла)**

   - S -> S + S
   - S -> S S
   - S -> (S)
   - S -> S*
   - S -> a
4. Для строки `fafbcfbfd` постройте дерево вывода в следующей грамматике **(0.8 балла)**

   - S -> S a A
   - S -> A
   - A -> A b B
   - A -> B
   - B -> c S d
   - B -> e
   - B -> f
5. Постройте грамматику для регулярного выражения `(a+bb)(a+c)*a*` **(0.8 балла)**

6. Постройте грамматику для вывода строк из нулей и единиц, содержащих нечетное число нулей и четное число единиц **(0.8 балла)**
7. Для указанной ниже грамматики: избавьтесь от левой рекурсии, а затем выполните левую факторизацию **(1.6 балла)**:

    - S -> A c | B b | S C a b
    - A -> a | b B | C
    - B -> B b | d
    - C -> a S | C d B d | d
8. Постройте автомат с магазинной памятью по представленной ниже грамматике **(1.6 балла)**:

    - S -> a A b B | b B S
    - A -> b A | a
    - B -> b B | c d
9. Покажите последовательность стеков во время работы АМП из предыдущего упражнения на слове `bcdbbcdaabcd` **(0.8 балла)**
10. Постройте синтаксический анализатор методом рекурсивного спуска для грамматики из упр. 8 **(1.2 балла)**

5.3 Пример первой модульной контрольной 4-го семестра

*Замечание: для заданий 1-3 никаких преобразований с грамматикой делать нельзя. Грамматика **может** быть леворекурсивной, пустой и содержать неопределенные нетерминалы* – не отчаивайтесь.

Задание №1. Для заданной грамматики постройте множества FIRST **(1 балл)** и FOLLOW **(1 балл)**.

Задание №2. Для заданной грамматики постройте селективные множества[1] **(0.6 балла)**.

Задание №3. Для заданной грамматики укажите, какое количество конфликтных ячеек будет содержать его таблица АМП. Отметьте в столбце «X» те правила, выбор которых не детерминирован **(0.6 балла)**.

```
O -> hshk | gdg | nO
I -> Kb | AUf | ArAt
J -> UrO | uJv
A -> iA | In
K -> epsilon | ms
U -> Kpn | sIn
```

Нетерминал	FIRST	FOLLOW

№	Правило	Селективное множество	X	№	Правило	Селективное множество	X
1				10			
2				11			
3				12			
4				13			
5				14			
6				15			
7				16			
8				17			
9				18			

[1] Напоминание: для правила вида A -> α селективное множество состоит из всех терминалов FIRST(α), а если ε входит в FIRST(α), то в селективное множество добавляется еще FOLLOW(A).

Количество конфликтных ячеек АМП:

Задание №4. Постройте аннотированное дерево разбора и граф зависимостей атрибутов для заданной грамматики:

```
Expr_p → Term_q Elist_q,p
Elist_p,q → + Term_r {ADD}_p,r,s Elist_s,q      s ← Alloc()
Elist_p,q → ε                                   q ← p
Term_p → Factor_q Tlist_q,p
Tlist_p,q → * Factor_r {MULT}_p,r,s Tlist_s,q   s ← Alloc()
Tlist_p,q → ε                                   q ← p
Factor_p → ( Expr_p )
Factor_p → ident_p
```

и выражения **(1.6 баллов)**: `(a + b) * (c + d)`

Задание №5. В соответствии с грамматикой языка MiniC выполните ручную трансляцию на язык атомов следующего кода **(1.6 баллов)**:

```
int A, B, i;
in A;
in B;
for(i = B - 1; i > A; --i){
      out i;
}
out "\nVsego chisel = ";
out B - A - 1;
```

Задание №6. В соответствии с грамматикой выражений языка MiniC постройте аннотированное дерево разбора и выпишите атомы с вычисленными атрибутами для выражения **(1.6 баллов)**:

```
(A % 2) == (B % 2)
```

5.4 Пример второй модульной контрольной 4-го семестра

Задание №1. Для заданной программы на языке MiniC заполните таблицу символов **(1 балл)** и список атомов **(1.6 балла)**.

Задание №2. Для списка атомов, полученного в Задании №1, сгенерируйте код для машины 8080.
При написании кода можно использовать процедуры loadOp(), saveOp(), loadRegs(), saveRegs() **(1.2 балла)**.

```
int A, B, i;
int main(){
  in A;
  in B;
  for(i = B - 1; i > A; --i){
      out i;
  }
  out "\nVsego chisel = ";
  out B - A - 1;
  return 0;
}
```

ID	Code	Kind	Type	Length	Default	Scope	Offset

Задание №3. Для заданной грамматики постройте LR(0)-автомат **(1.6 балла)**.

```
L -> Fe | epsilon
F -> f | On
O -> kL | xkO
```

Задание №4. Для LR(0)-автомата из Задания №3 постройте таблицы SLR-анализатора **(1.6 балла)**.

Задание №5. Для SLR-анализатора из Задания №4 придумайте слово из не менее, чем 5 символов, и проэмулируйте работу SLR-анализатора на этом слове **(1 балл)**.

6. Программа экзамена 3-го семестра

6.1 Определения *(5 баллов)*

- Препроцессор
- Трансляция, компиляция, интерпретация
- Компоновка
- Объектный файл, исполняемый модуль, динамическая и статическая библиотеки
- Лексический анализ, лексема, токен
- Синтаксический анализ, промежуточное представление, атом, дерево разбора
- Глобальная и локальная оптимизации
- Генерация кода
- Язык, естественный и формальный язык
- Алфавит, строка, распознаватель языка
- Конечный автомат, ДКА и НКА
- Регулярное выражение
- Формальная грамматика, порождающая и распознающая грамматики
- Сентенциальная форма, предложение, вывод в грамматике (левый и правый вывод)
- Язык, порождаемый грамматикой; эквивалентность языков
- Иерархия Хомского грамматик
- Дерево вывода
- Неоднозначная грамматика
- Генератор лексических анализаторов
- Левая рекурсия
- Восходящий и нисходящий синтаксический анализ, LL и LR анализаторы
- Метод рекурсивного спуска
- Автомат с магазинной памятью

6.2 Вопросы *(3 вопроса по 5 баллов)*

1. Трансляция, компиляция, интерпретация. Фазы работы транслятора. Лексемы, токены, промежуточные представления кода, генерация и оптимизация кода. Показать на примере трансляции выражения в обратную польскую запись.
2. Конечный автомат. Задание с помощью таблицы, графа. Регулярные выражения. Детерминированный и недетерминированный автоматы.

Преобразование НКА в ДКА. Моделирование НКА. Построение НКА по регулярному выражению.

3. Порождающая грамматика. Язык, порождаемый грамматикой. Иерархия Хомского. Примеры. Построение грамматики по регулярному выражению. Деревья вывода в КС-грамматиках. Левосторонний и правосторонний выводы. Неоднозначные грамматики. Алгоритм устранения левой рекурсии. Алгоритм левой факторизации.
4. Нисходящий и восходящий синтаксические анализаторы. Пример анализа строки обоими методами. LL и LR анализаторы. Пример построения нисходящего анализатора методом рекурсивного спуска.
5. Автомат с магазинной памятью. Пример работы. Соответствие между типами языков по иерархии Хомского и распознавателями. Алгоритм построения АМП для простой грамматики.

6.3 Упражнения

1) Построить НКА по регулярному выражению. Построить эквивалентный ему ДКА. Промоделировать работу НКА на конкретном слове. (5 баллов)
2) Показать на примере работу алгоритмов избавления от левой рекурсии и левой факторизации грамматики. (5 баллов)
3) Построить рекурсивный спуск по простой грамматике. (5 баллов)
4) Построить таблицу АМП для простой грамматики и промоделировать его работу на конкретном слове. (5 баллов)

7. Программа экзамена 4-го семестра

7.1 Определения *(5 баллов)*

- FIRST(α)
- FOLLOW(A)
- Селективные множества
- LL(1)-грамматика
- LR(1)-грамматика
- Предиктивный синтаксический анализатор
- Синтаксически управляемая трансляция
- Схема трансляции, семантическое действие
- Атрибут, синтезируемые и наследуемые атрибуты
- Аннотированное дерево разбора, граф зависимостей атрибутов
- Синтаксически управляемые определения (СУО)
- Семантическое правило
- S-атрибутное СУО, L-атрибутное СУО
- Атрибутная грамматика, побочные действия в СУО
- Трехадресный код (атом)

- Таблица символов
- Функциональный фрейм
- ПС-анализ, операции переноса и свертки
- Конфликты «сдвиг/свертка», «свертка/свертка»
- Табличный ПС-анализатор, таблицы действий и переходов
- LR(0)-ситуация
- Канонический набор множеств LR(0)-ситуаций
- LR(0)-автомат

7.2 Вопросы и упражнения

Для заданной грамматики выполните следующие упражнения, а также объясните работу всех использованных алгоритмов.

```
S -> Fli | mh
O -> epsilon | vFkj
F -> i | Of
```

1. Постройте множества FIRST (5 баллов), FOLLOW (5 баллов).
2. Постройте таблицу предиктивного (нисходящего) синтаксического анализатора (6 баллов).
3. Напишите на любом языке программирования код предиктивного (нисходящего) синтаксического анализатора (6 баллов).
4. Постройте LR(0)-автомат (6 баллов).
5. Постройте таблицы SLR-анализатора (6 баллов)
6. Придумайте слово из 5 символов и:
    a. промоделируйте работу нисходящего анализатора (3 балла)
    b. промоделируйте работу восходящего анализатора (3 балла)

8. Указания для обучающихся

Организуя свою учебную работу, студенты должны, во-первых, выявить рекомендуемый режим и характер учебной работы по изучению теоретического курса, практическому применению изученного материала, по выполнению заданий для самостоятельной работы, по использованию информационных технологий и т.д. Во-вторых, ознакомиться с указанным в методическом материале по дисциплине перечнем учебно-методических изданий, рекомендуемых студентам для подготовки к занятиям и выполнения самостоятельной работы, а также с методическими материалами на бумажных и/или электронных носителях, выпущенных кафедрой своими силами и предоставляемые студентам во время занятий.

Самостоятельная работа студентов, предусмотренная учебным планом, должна соответствовать более глубокому усвоению изучаемого курса, фор-

мировать навыки исследовательской работы и ориентировать студентов на умение применять теоретические знания на практике.

1. Работа с учебными пособиями.

Для полноценного усвоения курса студент должен, прежде всего, овладеть основными понятиями этой дисциплины. Необходимо усвоить определения и понятия, уметь приводить их точные формулировки, приводить примеры объектов, удовлетворяющих этому определению. Кроме того, необходимо знать круг фактов, связанных с данным понятием. Требуется также знать связи между понятиями, уметь устанавливать соотношения между классами объектов, описываемых различными понятиями.

2. Самостоятельное изучение тем.

Самостоятельная работа студента является важным видом деятельности, позволяющим хорошо усвоить изучаемый предмет и одним из условий достижения необходимого качества подготовки и профессиональной переподготовки специалистов. Она предполагает самостоятельное изучение студентом рекомендованной учебно-методической литературы, различных справочных материалов, написание рефератов, выступление с докладом, подготовку к лекционным и практическим занятиям, подготовку к зачёту и экзамену.

3. Подготовка к практическим занятиям.

При подготовке к практическим занятиям студентам рекомендуется следовать методическим рекомендациям по работе с учебными пособиями, приведенным выше.

4. Составление конспектов.

В конспекте отражены основные понятия темы. Для наглядности и удобства запоминания используются схемы и таблицы.

VII. Материально-техническое обеспечение

Для аудиторной работы

Учебная аудитория № 212 (170002, Тверская обл., г.Тверь, Садовый переулок, д.35)	Набор учебной мебели, меловая доска, мультимедийный комплекс "I - Lerner .ru" в составе: проектор Epson EB -575 Wi, маркерная доска, панель управления Epson ELPCBO2, запасная лампа, запасной фильтр для проектора.

Для самостоятельной работы

Помещение для самостоятельной работы обучающихся: Компьютерный класс факультета прикладной математики и кибернетики № 4б (170002, Тверская обл.,	Персональные ЭВМ (компьютер RAMEC STORM C2D 4600/160Gb/DVD-RW+Монитор LG TFT 17" L1753S-SF silver – 24 шт.), мультимедийный проектор BenQ MP 724 с потолочным креплением и экран 1105, кондиционер General Climate – 2 шт., коммутатор D-Link 10/100/1000mbps 16-potr DGS-1016D, коммутатор D-Link 10/100/1000mbps 16-potr DGS-1016D- 2 шт.

г.Тверь, Садовый переулок, д.35)	

VIII. Сведения об обновлении рабочей программы дисциплины

№ п.п.	Обновленный раздел рабочей программы дисциплины	Описание внесенных изменений	Реквизиты документа, утвердившего изменения
1.	3. Объем дисциплины	Выделение часов на практическую подготовку	От 29.10.2020 года, протокол № 3 ученого совета факультета
2.	II. Содержание дисциплины, структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий	Выделение часов на практическую подготовку по темам	От 29.10.2020 года, протокол № 3 ученого совета факультета