Министерство науки и высшего образования Российской Федерации
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)

Институт прикладной математики и компьютерных наук

УТВЕРЖДЕНО:
Директор
А. В. Замятин

Рабочая программа дисциплины

Методы компиляции

по направлению подготовки / специальности

09.03.03 Прикладная информатика

Направленность (профиль) подготовки:
Искусственный интеллект и большие данные
Форма обучения
Очная

Квалификация
Бакалавр

Год приема
2025

СОГЛАСОВАНО:
Руководитель ОП
С.П.Сущенко

Председатель УМК
С.П.Сущенко

Томск – 2025

1. Цель и планируемые результаты освоения дисциплины

Целью освоения дисциплины является формирование следующих компетенций:

ОПК-2 Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности.

ОПК-5 Способен инсталлировать программное и аппаратное обеспечение для информационных и автоматизированных систем.

Результатами освоения дисциплины являются следующие индикаторы достижения компетенций:

ИОПК-2.2 Применяет знания, полученные в области информационных технологий и программных средств, при решении задач профессиональной деятельности

ИОПК-2.3 Использует современные информационные технологии, в том числе отечественного производства на всех этапах разработки программных систем

ИОПК-5.3 Выполняет работы по настройке, администрированию и проверке работоспособности программного и аппаратного обеспечения при решении задач профессиональной деятельности

2. Задачи освоения дисциплины

- Изучение формальных грамматик и их классификация.
- Изучение основных методов реализации блоков лексического и синтаксического анализа.
- Изучение методов детерминированного синтаксического анализа на основе восходящей и нисходящей стратегий.
- Изучение методов генерации команд программ, описанных в виде промежуточного представления.
- Получение практических навыков создания трансляторов языков программирования.

3. Место дисциплины в структуре образовательной программы

Дисциплина относится к Блоку 1 «Дисциплины (модули)».

Дисциплина относится к обязательной части образовательной программы. Дисциплина входит в модуль «Модуль «Разработка программного обеспечения»».

4. Семестр(ы) освоения и форма(ы) промежуточной аттестации по дисциплине

Шестой семестр, зачет с оценкой

5. Входные требования для освоения дисциплины

Для успешного освоения дисциплины требуются результаты обучения по следующим дисциплинам: «Информатика», «Языки программирования», «Алгоритмы и структуры данных», «Дискретная математика», «Теория автоматов».

6. Язык реализации

Русский

7. Объем дисциплины

Общая трудоемкость дисциплины составляет 3 з.е., 108 часов, из которых:
- лекции: 32 ч.
- лабораторные: 16 ч.
- самостоятельная практическая подготовка: 16 ч.

Объем самостоятельной работы студента определен учебным планом.

8. Содержание дисциплины, структурированное по темам

Тема 1. Языки и грамматики. Трансляция
1.1. Языки и грамматики
1.2. Грамматический анализ и трансляция
Тема 2. Автоматные грамматики и лексический анализ
2.1. Автоматные грамматики и конечный автомат
2.2. Недетерминированный конечный автомат
2.3. Лексический анализ
2.4. Праволинейные грамматики
2.5. Регулярные множества
Тема 3. КС-грамматики и синтаксический LL-анализ
3.1. Недетерминированный LL-анализ
3.2. Детерминированный LL-анализ
Тема 4. Обратная польская строка
4.1. Обратная польская строка для арифметических выражений
4.2. Грамматика с дополнительными операциями
Тема 5. Обратная польская строка для операторов языка
5.1. Грамматика с условными операторами и циклами
5.2. Распределение памяти и описание переменных
5.3. Грамматика с процедурами
5.4. Обработка ошибок при трансляции и выполнении программы
Тема 6. Генерация команд в компиляторе
6.1. Одноадресная система команд
6.2. Генерация команд выделения памяти и индексирования
Тема 7. Генерация команд сравнения и перехода
7.1. Генерация команд сравнения
7.2. Генерация команд перехода
7.3. Формирование меток (адресов) для команд перехода
Тема 8. Рекурсивный спуск. Анализ снизу-вверх
8.1. Рекурсивный спуск
8.2. Анализ снизу-вверх
Тема 9. Детерминированный LR-анализ
9.1. Отношения простого предшествования
9.2. Отношения операторного предшествования
9.3. LR(k)-анализатор.

9. Текущий контроль по дисциплине

Текущий контроль по дисциплине проводится путем контроля посещаемости, контроля выполнения лабораторных работ, письменных контрольных работ по лекционному материалу и фиксируется в форме контрольной точки не менее одного раза в семестр.

Практическая подготовка оценивается по результатам выполненных лабораторных работ.

Оценочные материалы текущего контроля размещены на сайте ТГУ в разделе «Информация об образовательной программе» - https://www.tsu.ru/sveden/education/eduop/.

10. Порядок проведения и критерии оценивания промежуточной аттестации

Зачет с оценкой в шестом семестре проводится путём выполнения в течение семестра четырёх контрольных работ в письменной форме по билетам, а также самостоятельной разработки синтаксиса учебного языка программирования и реализации

для него транслятора. При этом рекомендуется разработку синтаксиса языка и реализацию транслятора выполнять бригадным методам, в бригаде может быть 2 или студента.

Билет содержит одно теоретические и одно практическое задание. Продолжительность выполнения контрольных работ – 1,5 часа. Разработка синтаксиса учебного языка программирования и этапов реализации транслятора выполняется самостоятельно и на лабораторных занятиях.

Примерный перечень теоретических заданий на контрольных работах.

1.Построение недетерминированного конечного автомата по недетерминированной автоматной грамматике и его функционирование. Трудоемкость работы автомата. Привести пример грамматики, автомата и его работы!

2.Преобразование недетерминированной автоматной грамматики в детерминированную. Трудоемкость преобразования. Привести пример грамматики и её преобразования!

3.Преобразование праволинейной грамматики в автоматную. Трудоемкость преобразования. Привести пример!

4.Преобразование автоматной грамматики с удалением бесполезных нетерминалов. Трудоемкость преобразования. Привести пример!

5.Преобразование регулярного выражения в праволинейную грамматику. Трудоемкость преобразования. Привести пример!

6. Недетерминированный алгоритм анализа сверху-вниз (магазинный автомат). Его построение по заданной КС-грамматике, функционирование. Доказательство корректности. Пример!

7. Порождающие правила для грамматики, включающей присваивания, выражения, переменные с индексами, последовательность операторов присваивания. Генерация ОПС для всех этих конструкций языка при работе анализатора LL(1). Пример!

8.Порождающие правила для грамматики, включающей вложенные условные операторы IF-THEN-ELSE, выражения с операциями сравнения. Генерация ОПС для всех этих конструкций языка при работе анализатора LL(1). Пример!

9.Порождающие правила для грамматики, включающей описания переменных и массивов, операторы ввода-вывода и операторы выделения памяти для массивов. Генерация ОПС для всех этих конструкций языка при работе анализатора LL(1). Пример!

10.Архитектура одноадресных команд. Принципы генерации команд на основе ОПС. Правила генерации команд, реализующих арифметические операции и присваивание. Пример!

11.Принципы генерации команд на основе ОПС, реализующих операции индексирования элементов массива. Пример!

12. Правила генерации команд на основе ОПС, реализующих операции сравнения. Пример!

13.Общий алгоритм вычисления адресов для команд перехода при генерации команд на основе ОПС. Пример!

14.Принципы генерации команд на основе ОПС, реализующих операции условного и безусловного перехода. Пример!

15.Недетерминированный алгоритм анализа снизу-вверх (магазинный автомат). Его построение по заданной произвольной КС-грамматике, функционирование. Доказательство корректности. Трудоемкость работы алгоритма. Пример.

16.Отношения простого предшествования, определение, вычисление из элементарных отношений first и last. Построение анализатора и его функционирование. Условия однозначности его работы. Трудоемкость работы анализатора. Пример.

17.Отношения операторного предшествования, определение, вычисление из элементарных отношений first, last, firstterm и lastterm. Построение анализатора и его

функционирование. Условия однозначности его работы. Трудоемкость работы анализатора. Пример.

18.Грамматика нестрогого предшествования, определение. Преобразование такой грамматики к грамматике простого предшествования. Пример.

19.Построение анализатора методом рекурсивного спуска, преобразование грамматики для этого. Условия однозначности его работы. Трудоемкость работы анализатора. Пример.

20.LR(k)-анализатор. Принципы построения LR(1)-анализатора, его функционирование. Условия однозначности его работы. Трудоемкость работы анализатора. Пример.

Примерный перечень практических заданий на контрольных работах.

1.Построить грамматику, конечный автомат и семантические программы для анализа следующих входных данных: Строка символов, состоящая из слов русского языка, в конце – символ точка. Слова разделены пробелами (одним или более). На выходе автомата: 1-е слово, число символов в слове, 2-е слово, число символов в слове, и т.д. В конце вывод – общее количество слов. При ошибке на входе – выдать сообщение.

2.Построить грамматику, конечный автомат и семантические программы для анализа следующих входных данных: Строка символов, состоящая из целых десятичных чисел со знаком (+ или – обязательно!), в конце – символ '#'. Числа разделены пробелами (одним или более). На выходе автомата: 1-е число, число цифр в нем, 2-е число, число цифр в нем, и т.д. В конце вывод – общее количество чисел. При ошибке на входе – выдать сообщение.

3.Построить грамматику, конечный автомат и семантические программы для анализа следующих входных данных: Строка символов, состоящая из слов английского языка, в конце – символ точка. Слова разделены запятыми (после запятой – один или более пробелов) или пробелами (одним или более). На выходе автомата: 1-е слово, число символов в слове, 2-е слово, число символов в слове, и т.д. В конце вывод – общее количество слов. При ошибке на входе – выдать сообщение.

4.Рассмотреть по шагам пример анализа, генерации ОПС для примера: c=a+e;

5.Рассмотреть по шагам вычисление ОПС для примера: b1=1; a[i+1]=b1–5.4*c[i];

6.Рассмотреть по шагам пример анализа, генерации ОПС для примера:
begin c=a+e end

7.Рассмотреть по шагам вычисление ОПС для примера: begin b=1; a=(b–5)*c end

8.Рассмотреть по шагам пример анализа, генерации ОПС для примера: {c=a*5+e;}

9.Рассмотреть по шагам вычисление ОПС для примера: {b=1; a[i+1]=(b–4)*c[i];}

10.Рассмотреть по шагам пример анализа, генерации ОПС для примера:
begin c=a*7 end

11.Рассмотреть по шагам вычисление ОПС для примера:
begin b=9; a=(b+5)/(c+4) end

12.Пример генерации команд для операторов: a:=10; b:=M[a]; if a>b then b:=1 else b:=2. Записать по шагам все действия.

13. Пример генерации команд для выражения x:=(b*c-d)*a. Записать по шагам все действия с магазином при генерации.

14.Пример генерации команд для выражения x:=(a+b)*(c-d). Записать по шагам все действия с магазином при генерации.

15. Пример генерации команд для выражения x[i+1]:=y[i,j-a]*2. Записать по шагам все действия с магазином при генерации.

16. Пример генерации команд для операторов: a:=L[1]; b:=2; while a>b do begin b:=b*3; a:=a*2 end. Записать по шагам все действия с магазином при генерации.

Требования к разработке синтаксиса учебного языка программирования и реализации для него транслятор-интерпретатора. Работа выполняется бригадой из двух или трёх студентов.

Общие требования к языку для бригады из двух студентов:

1) типы данных – целые или вещественные числа и переменные (можно без явных описаний типа), одномерные массивы, причём массивы могут быть статическими;

2) операторы присваивания и формулы со скобками и операциями с двумя приоритетами, (+, –) – низшего, (*, /) – высшего;

3) условные операторы и циклы с условиями, включая операции сравнения;

4) операторы ввода и вывода.

Если в бригаде три человека, то в языке должны быть некоторые расширения, например, 2 типа данных (целые и вещественные), или стандартные математические функции в математических выражениях или др.

Замечание: на минимальную положительную оценку достаточно, чтобы в языке были только простые переменные без явных описаний типа, а из операторов - присваивания и формулы со скобками с операциями +, –, *, /, а также операторы ввода и вывода.

Должны быть разработаны следующие описания к транслятору-интерпретатору:

1) список лексем с номерами лексем, таблица переходов автомата;

2) КС-грамматика языка, в которой лексемы суть терминалы;

3) КС-грамматика языка, преобразованная в нестрогую форму Грейбах;

4) семантические действия для генерации ОПС;

5) список операций ОПС;

6) формат ОПС.

Тесты для проверки функционирования транслятора-интерпретатора:

1) проверка сложных формул с вводом и выводом;

2) тест с действиями: ввод n, ввод n элементов массива, упорядочение массива, вывод массива.

Компоненты транслятора-интерпретатора:

1) лексический анализатор в виде функции;

2) синтаксический анализатор – генератор ОПС;

3) интерпретатор ОПС.

Каждая контрольная работа, а также реализация транслятора оценивается оценками «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Критерии оценивания контрольной работы:

- «отлично», задания выполнены полностью и без ошибок, приведён правильный пример;

- «хорошо», задания выполнены полностью, но с несущественными ошибками, приведён в целом правильный пример, возможно с несущественными ошибками;

- «удовлетворительно», задания выполнены не полностью, с существенными ошибками, но приведён в целом правильный пример, или задания выполнены с несущественными ошибками, но примеры отсутствуют;

- «неудовлетворительно», оба задания отсутствуют, или выполнено только одно задание или выполнены оба задания но оба с грубейшими ошибками.

Критерии оценивания реализации транслятора:

- «отлично», описания транслятора выполнены полностью и без существенных ошибок, транслятор реализует все необходимые конструкции языка и правильно исполняет все тесты;

- «хорошо», описания транслятора выполнены почти полностью и без существенных ошибок, транслятор реализует все необходимые конструкции языка, возможно с замечаниями, правильно исполняет почти все тесты;

- «удовлетворительно», описания транслятора соответствуют языку программирования, но в языке реализованы не все конструкции, указанные в задании, сами описания выполнены почти полностью и без существенных ошибок, транслятор реализует некоторые конструкции языка, возможно с замечаниями, и правильно исполняет соответствующие тесты;

- «неудовлетворительно», описания и реализация не соответствует требованиям оценки «удовлетворительно».

Транслятор можно реализовать на любом удобном языке программирования, например, на Си.

Так как реализация транслятора осуществляется бригадой, то в описании транслятора и его реализации в программном коде должны разбираться все члены бригады. Если студент, входящий в бригаду, разбирается с трудом и с ошибками, то его личная оценка снижается по сравнению с общей оценкой реализации, а если совсем не разбирается, то его личная оценка = «неудовлетворительно».

Для получения положительной оценки по зачёту требуется выполнить каждую контрольную работу и реализацию транслятора не хуже, чем на «удовлетворительно». Общая оценки по зачёту вычисляется путём усреднения всех этих оценок, причём вес оценки за реализацию транслятора удваивается.

Оценочные материалы для проведения промежуточной аттестации размещены на сайте ТГУ в разделе «Информация об образовательной программе» - https://www.tsu.ru/sveden/education/eduop/.

11. Учебно-методическое обеспечение

а) Электронный учебный курс по дисциплине в электронном университете «LMS IDO»

б) Оценочные материалы текущего контроля и промежуточной аттестации по дисциплине.

12. Перечень учебной литературы и ресурсов сети Интернет

а) основная литература:

1. А. Ахо, Р.Сети, Дж. Ульман, Компиляторы: принципы, технологии, инструменты, Вильямс, 2003 г., 768 с.
2. Лебедев В.Н., Введение в системы программирования, Москва: Статистика, 1975 г., 312 с.
3. Грис, Д., Конструирование компиляторов для цифровых вычислительных машин, Мир, 1975 г., 544 с.
4. Ахо А., Ульман Дж., Теория синтаксического анализа, перевода и компиляции, Мир, 1978 г., 1104 с.

б) дополнительная литература:

5. Кнут Д., Искусство программирования, Мир, 1976 г., 736 с.
6. Пахомов Б.И., C/C++ и MS Visual C++ 2008 для начинающих, БХВ-Петербург, 2009 г., 642 с.

13. Перечень информационных технологий

а) лицензионное и свободно распространяемое программное обеспечение:
MS Visual Studio C++, Borland C++ Builder.

б) информационные справочные системы:

– Электронный каталог Научной библиотеки ТГУ – http://chamo.lib.tsu.ru/search/query?locale=ru&theme=system

– Электронная библиотека (репозиторий) ТГУ – http://vital.lib.tsu.ru/vital/access/manager/Index

– ЭБС Лань – http://e.lanbook.com/

– ЭБС Консультант студента – http://www.studentlibrary.ru/

– Образовательная платформа Юрайт – https://urait.ru/

– ЭБС ZNANIUM.com – https://znanium.com/

– ЭБС IPRbooks – http://www.iprbookshop.ru/

14. Материально-техническое обеспечение

Аудитории для проведения занятий лекционного типа и контрольных работ.

Аудитории для проведения лабораторных занятий, оснащенные компьютерной техникой и доступом к сети Интернет, в электронную информационно-образовательную среду и к информационным справочным системам..

Помещения для самостоятельной работы, оснащенные компьютерной техникой и доступом к сети Интернет, в электронную информационно-образовательную среду и к информационным справочным системам.

15. Информация о разработчиках

Костюк Юрий Леонидович, д.т.н., профессор кафедры теоретических основ информатики.

Провкин Виктор Алексеевич, ассистент кафедры компьютерной безопасности.