



Автономная некоммерческая организация
дополнительного профессионального образования
(АНО ДПО «Инфосфера»)
Центр профессиональной подготовки
ИНСТИТУТ ПРОГРАММНЫХ СИСТЕМ

**Рабочая программа дисциплины
«Теория языков программирования»**

Разработал:
преподаватель ИПС
АНО ДПО «Инфосфера»
С.В. Шамбир

Йошкар-Ола, 2017

Пояснительная записка

Цель курса ознакомить студентов с теорией и практикой трансляции языков программирования. Темы курса включают в себя устройство компиляторов, лексический анализ, синтаксический анализ, таблицы символов, обработку объявлений, управление памятью, генерацию кода и методы оптимизации.

Планируемые результаты обучения

Осуществляется предварительная самостоятельная или под руководством разработка алгоритмов с использованием графических средств (блок-схемы, UML-диаграммы и др.). Не требуется взаимодействие с другими программистами, системным аналитиком и архитектором программного обеспечения. Осуществляется решение типовых задач. Полученные результаты представляются руководителю разработки программного обеспечения.

Осуществляется предварительная самостоятельная или под руководством разработка алгоритмов с использованием графических средств (блок-схемы, UML-диаграммы и др.). Не требуется взаимодействие с другими программистами, системным аналитиком и архитектором программного обеспечения. Осуществляется решение типовых задач. Полученные результаты представляются руководителю разработки программного обеспечения.

Выполняются самостоятельная разработка процедур сборки модулей и компонент программного обеспечения и верификация выпусков программного продукта. Производится разработка процедур развертывания и обновления программного обеспечения, процедур миграции и преобразования (конвертации) данных и программных интерфейсов с использованием выбранных программных средств, технологий создания открытых систем. Осуществляется решение различных типов задач проектирования программных комплексов различной сложности, выбор способов реализации взаимодействия программных компонент/модулей. Требуется взаимодействие с программистами-разработчиками модулей, архитектором программного обеспечения. Полученные результаты представляются руководителю разработки программного обеспечения.

В процессе интеграции программных модулей и компонент и верификации выпусков программного продукта осуществляется сборка модулей и компонент программного обеспечения, производится интеграция с внешней средой. Обеспечивается согласованное функционирование и требуемый уровень качества.

Проведение интеграции программных модулей и компонент и верификация выпусков программного продукта предполагают определение задач программной интеграции, распределение задач между подчиненными, обеспечение взаимодействия подчиненных сотрудников.

Программист несет ответственность за результат выполнения работ на уровне группы программистов. В процессе интеграции требуется взаимодействие с архитектором программного обеспечения. Полученные результаты представляются руководителю разработки программного обеспечения.

Учебно-тематический план

№	Наименование разделов	Всего часов	В том числе		Форма контроля
			Лекции	Практ. занятия	
1	Обзор языков программирования	4	2	2	Контрольная работа
1.1	История языков программирования				
1.2	Краткий обзор парадигм программирования				
1.3	Роль трансляции в процессе программирования				
2	Основные вопросы проектирования языков программирования	4	2	2	Контрольная работа
2.1	Основные принципы разработки языков программирования				
2.2	Цели разработки				
2.3	Виды эквивалентности типов				
2.4	Модели данных				
2.5	Модели конструкций управления				
2.6	Механизмы абстракции				
3	Виртуальные машины	4	2	2	Контрольная работа
3.1	Понятие виртуальной машины				
3.2	Иерархия виртуальных машин				
3.3	Иерархия виртуальных машин				
4	Введение в теорию трансляции	4	2	2	Контрольная работа
4.1	Сравнение интерпретаторов и компиляторов				
4.2	Стадии трансляции				
4.3	Машинно-зависимая и машинно-независимая части транслятора				
4.4	Трансляция как одна из задач программной инженерии				
5	Лексический анализ	4	2	2	Контрольная работа
5.1	Применение регулярных выражений в программах лексического анализа				
5.2	Ручное кодирование и автоматическая генерация лексических анализаторов				
5.3	Формальное определение лексем; реализация конечного автомата				
6	Синтаксический анализ	8	4	4	Контрольная работа
6.1	Формальное определение грамматик				
6.2	BNF и EBNF				
6.3	Нисходящий и восходящий анализ				
6.4	Табличные синтаксические анализаторы и метод рекурсивного спуска				

6.5	Управление таблицами символов				
6.6	Использование средств поддержки процесса трансляции				
7	Модели управления выполнением				
7.1	Порядок вычисления подвыражений	8	4	4	Контрольная работа
7.2	Исключения и их обработка				
7.3	Системы динамической поддержки				
8	Работа с объявлениями, модульностью и управление размещением в памяти				
8.1	Виды объявлений	8	4	4	Контрольная работа
8.2	Механизмы параметризации				
8.3	Параметризация типов				
8.4	Механизмы разделения и ограничения областей видимости				
8.5	Сборка мусора				
9	Системы типов				
9.1	Тип данных как набор значений с операциями над ними	8	4	4	Контрольная работа
9.2	Типы данных				
9.3	Модели проверки типов				
9.4	Семантические модели типов				
9.5	Определяемых пользователем				
9.6	Параметрический полиморфизм				
9.7	Полиморфизм подтипа				
9.8	Алгоритмы проверки типов				
10	Интерпретация				
10.1	Итеративная и рекурсивная интерпретации	8	4	4	Контрольная работа
10.2	Итеративная интерпретация промежуточного представления				
10.3	Рекурсивная интерпретация дерева разбора программы				
11	Генерация кода				
11.1	Промежуточное представление и объектный код	8	4	4	Контрольная работа
11.2	Промежуточные представления; реализация генераторов кода				
11.3	Генерация кода путем обхода дерева				
11.4	Контекстно-зависимая трансляция				
11.5	Использование регистров				
12	Оптимизация				
12.1	Машинно-независимая оптимизация	4	2	2	Контрольная работа
12.2	Анализ потоков данных				
12.3	Оптимизации циклов				

12.4	Машинно-зависимая оптимизация				
	Итого	72	36	36	

Содержание курса

Требования к слушателям

SE102 Основы программирования

CS220 Архитектура ЭВМ и программирование микропроцессоров

SE201 Объектно-ориентированное программирование

CS103 Структуры данных и алгоритмы

Учебные задачи курса

Освоив дисциплину «Теория языков программирования и методы трансляции», студент должен:

Иметь представление

о структуре транслятора, методах и алгоритмах построения лингвистических программных средств.

Знать

- этапы трансляции программы
- алгоритмы реализации лексического анализа
- эффективные алгоритмы нисходящих и восходящих методов синтаксического анализа
- основные задачи и подходы при реализации семантического анализа.

Уметь

- разрабатывать грамматику простого языка программирования
- разрабатывать код лексического, синтаксического и семантического анализа кода

Иметь опыт

- построения простого транслятора языка программирования по выбранной грамматике.

Содержание курса «Теория языков программирования»

Тема 1. Обзор языков программирования

История языков программирования; краткий обзор парадигм программирования; роль трансляции в процессе программирования

Тема 2. Основные вопросы проектирования языков программирования.

Основные принципы разработки языков программирования; цели разработки; виды эквивалентности типов; модели данных; модели конструкций управления; механизмы абстракции

Тема 3. Виртуальные машины

Понятие виртуальной машины; иерархия виртуальных машин;

Тема 4. Введение в теорию трансляции

Сравнение интерпретаторов и компиляторов; стадии трансляции; машинно-зависимая и машинно-независимая части транслятора; трансляция как одна из задач программной инженерии

Тема 5. Лексический анализ

Применение регулярных выражений в программах лексического анализа; ручное кодирование и автоматическая генерация лексических анализаторов; формальное определение лексем; реализация конечного автомата

Тема 6. Синтаксический анализ

Формальное определение грамматик; BNF и EBNF; нисходящий и восходящий анализ; табличные синтаксические анализаторы и метод рекурсивного спуска; управление таблицами символов; использование средств поддержки процесса трансляции

Тема 7. Модели управления выполнением

Порядок вычисления подвыражений; исключения и их обработка; системы динамической поддержки

Тема 8. Работа с объявлениями, модульностью и управление размещением в памяти

Виды объявлений; механизмы параметризации; параметризация типов; механизмы разделения и ограничения областей видимости; сборка мусора

Тема 9. Системы типов

Тип данных как набор значений с операциями над ними; типы данных; модели проверки типов; семантические модели типов, определяемых пользователем; параметрический полиморфизм; полиморфизм подтипа; алгоритмы проверки типов

Тема 10. Интерпретация

Итеративная и рекурсивная интерпретации; итеративная интерпретация промежуточного представления; рекурсивная интерпретация дерева разбора программы

Тема 11. Генерация кода

Промежуточное представление и объектный код; промежуточные представления; реализация генераторов кода; генерация кода путем обхода дерева; контекстно-зависимая трансляция; использование регистров

Тема 12. Оптимизация

Машинно-независимая оптимизация; анализ потоков данных; оптимизации циклов; машинно-зависимая оптимизация

Методические рекомендации.

Для реализации заявленных учебных целей используются методологические технологии, реализующие деятельностный, личностно-ориентированный, практико-ориентированный подходы. В ходе изучения курса используются технологии проблемного, эвристического, модульно-рейтингового, опережающего, знаково-контекстного, проектного, дифференцированного, группового, личностно-ориентированного обучения, информационные и дистанционные технологии.

Основными видами учебной работы (стратегическими технологиями) являются лекции. Также делается упор на практические занятия. Лекция классическая предусматривает (ЛК) сообщение темы, плана лекции, списка литературы, изложение информации под запись слушателям (монолог, диалог в проблемном изложении), фронтальная проверка знаний, подведение итогов, выводы, ответы на вопросы слушателей, возникшие в ходе лекции или по итогам самостоятельной работы над теоретическим материалом). Практические работы предусматривают последовательное выполнение заданий слушателями в классе, направленных на упорядочивание, систематизацию теоретических знаний; выполнение упражнений на запоминание, осмысление и оперирование языковой терминологией; перевод знаний на уровень практических умений и навыков. Самостоятельная работа включает процедуры самообучения слушателей курса, инициируемые и управляемые со стороны преподавателя в режиме их подготовки к лекциям, семинарам, практическим работам, сдаче экзаменов и зачетов; процедуры самообучения в условиях свободы выбора заданий для самостоятельного освоения новых знаний, овладения умениями, используя весь арсенал современных источников информации (учебники, учебные пособия, ресурсы интернета, собственный опыт); консультирование преподавателя.

При организации учебных занятий используются активные методы обучения (работа в группах, взаимообучение, самоконтроль, индивидуальные задания дифференцированной сложности).

В процессе обучения возможно использование следующих тактических технологий: лекция классическая, лекция проблемная, лекция-визуализация, лекция-диалог, аудиторно-практическое занятие классическое, практикум-лабораторная работа, самообучение.

Литература

1. Компиляторы. Принципы, технологии и инструментарий. Альфред В. Ахо, Моника С. Лам, Рави Сети, Джеффри Д. Ульман, 2008 - 1184 стр.
2. Modern Compiler Implementation in Java. Andrew W. Appel, - 512 стр.
3. Optimizing Compilers for Modern Architectures. Randy Allen, 2010 - 790 стр.
4. Engineering a Compiler. Keith Cooper, 2010 - 801 стр.
5. Writing Compilers and Interpreters: A Software Engineering Approach. Ronald Mak, 2011 – 864 p.
6. Engineering a Compiler. Keith Cooper, Linda Torczon, 2011 - 824 p.
7. Optimizing Compilers for Modern Architectures: A Dependence-based Approach. Randy Allen, Ken Kennedy, 2001 – 790 p.

Контрольные задания

Контрольные вопросы для самостоятельной оценки качества освоения дисциплины.

1. История языков программирования; краткий обзор парадигм программирования; роль трансляции в процессе программирования.
2. Основные вопросы проектирования языков программирования.
3. Понятие виртуальной машины; иерархия виртуальных машин.
4. Сравнение интерпретаторов и компиляторов; стадии трансляции; машинно-зависимая и машинно-независимая части транслятора.
5. Лексический анализ.
6. Синтаксический анализ.
7. Порядок вычисления подвыражений; исключения и их обработка; системы динамической поддержки.
8. Работа с объявлениями, модульностью и управление размещением в памяти
9. Системы типов
10. Итеративная и рекурсивная интерпретации
11. Рекурсивная интерпретация дерева разбора программы
12. Промежуточное представление и объектный код.
13. Генерация кода путем обхода дерева; контекстно-зависимая трансляция;
14. Использование регистров
15. Машинно-независимая оптимизация.
16. Анализ потоков данных.
17. Оптимизации циклов.
18. Машинно-зависимая оптимизация