**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Кафедра Вычислительной техники

**утверждена**

на заседании кафедры

Протокол № 10 от 07 июня 2018 г.

**Рабочая программа дисциплины**

|  |
| --- |
| «Вычислительные системы» |

|  |
| --- |
| Направление: 09.04.01 Информатика и вычислительная техника |

|  |
| --- |
| Программа: Сети ЭВМ и телекоммуникации |

|  |
| --- |
| Квалификация: Магистр |

|  |
| --- |
| Форма обучения: очная |

**Составитель программы:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Черкашин Евгений Александрович / 7 июня 2018 г

**Заведующий кафедрой:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Дорофеев Андрей Сергеевич / 7 июня 2018 г

Год набора – 2017, 2018

Иркутск, 2018 г.

**1. Перечень планируемых результатов обучения по дисциплине, соотнесённых с планируемыми результатами освоения образовательной программы**

**1.1. Дисциплина «Вычислительные системы» обеспечивает формирование следующих компетенций с учётом этапа**

|  |  |
| --- | --- |
| **Код и наименование компетенции** | **Код этапа освоения компетенции** |
| ОК-2 – способность понимать роль науки в развитии цивилизации, соотношение науки и техники, иметь представление о связанных с ними современных социальных и этических проблемах, понимать ценность научной рациональности и ее исторических типов | ОК-2.1 |
| ОК-8 – способность к профессиональной эксплуатации современного оборудования и приборов (в соответствии с целями магистерской программы) | ОК-8.1 |

**1.2. В результате освоения дисциплины у обучающихся должны быть сформированы**

|  |  |  |
| --- | --- | --- |
| **Обобщенные трудовые функции / Трудовые функции** | **Код этапа освоения компетенции** | **Результат обучения** |
| ведение научных исследований с использованием вычислительных систем | ОК-2.1 | ***Знать*** методологию проведения научных исследований в области создания высокопроизводительных вычислительных систем, современные структуры данных, позволяющие управлять вычислительным процессом, повышать его вычислительную эффективность; методологию проведения научных исследований с использованием высокопроизводительных вычислительных систем. ***Уметь*** разрабатывать вычислительно-эффективное программное обеспечение; проводить процесс совершенствования структур данных существующего исходного кода программных систем; создавать программы для вычислительных кластеров; разрабатывать собственные средства программирования. ***Владеть*** инструментарием разработки высокопроизводительного программного обеспечения; моделями архитектур вычислительных систем; инструментами разработки компиляторов языков описания предметной области. |
| разработка программного обеспечения для вычислительных систем различных архитектур | ОК-8.1 | ***Знать*** архитектуры современных вычислительных систем; средства обеспечения взаимодействия между элементами вычислительной системы;  ***Уметь*** на качественном уровне прогнозировать производительность разрабатываемого программного обеспечения на основе анализа аппаратной архитектуры вычислительного устройства; разрабатывать структуры данных и алгоритмы эвристического поиска решений.  ***Владеть*** методиками проектирования параллельных схем реализаций алгоритмов, трансляторов с языка высокого уровня, в том числе языков описания предметной области; технологиями проведения матричных операций на платформах кластерных вычислительных систем. |

**2. Место дисциплины в структуре ООП**

Изучение дисциплины «Вычислительные системы» базируется на результатах освоения следующих дисциплин: «Современные проблемы информатики и вычислительной техники», «Философские проблемы естественных, гуманитарных и технических наук»

Дисциплина является предшествующей для дисциплин: «Компьютерные технологии в науке и образовании», «История и методология информатики и вычислительной техники», «Информационная безопасность и защита данных», «Архитектура сетей и систем телекоммуникаций», «Администрирование в телекоммуникационных системах».

**3. Объем дисциплины**

Объем дисциплины составляет - 3 ЗЕТ

|  |  |  |
| --- | --- | --- |
| Вид учебной работы | Трудоемкость в академических часах (Один академический час соответствует 45 минутам астрономического часа) | |
| Всего | Семестр №1 |
| Общая трудоемкость дисциплины | 108 | 108 |
| Аудиторные занятия, в том числе: | 36 | 36 |
| лекции | 12 | 12 |
| лабораторные работы | 24 | 24 |
| практические/семинарские занятия |  |  |
| Самостоятельная работа (в т.ч. курсовое проектирование) | 72 | 72 |
| Трудоемкость промежуточной аттестации |  |  |
| Вид промежуточной аттестации (итогового контроля по дисциплине) | Зачет | Зачет |

**4. Структура и содержание дисциплины**

**4.1. Сводные данные по содержанию дисциплины**

**Семестр №1**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Наименование раздела и темы дисциплины** | **Вид контактной работы** | | | | | | | | **Форма текущего контроля и вид промежуточной аттестации** | | |
| Лекции | | ЛР | | ПЗ(СЕМ) | | СРС | | |  | |
| № | Кол. час. | № | Кол. час. | № | Кол. час. | № | Кол. час. | | |  |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | | | 11 |
| 1 | Аппаратные архитектуры современных вычислительных средств | 1 | 2 | 1 | 2 |  |  | 1, 1 | 30 | | | Устный опрос |
| 2 | Структуры данных: интенсивный подход к повышению производительности вычислительных систем | 2 | 2 | 2 | 4 |  |  | 1 | 12 | | | Защита лабораторной работе |
| 3 | Процессоры Intel и оптимизация исходного кода: экстенсивное повышение производительности вычислительных систем | 3 | 2 | 3 | 2 |  |  | 1 | 6 | | | Защита лабораторной работе |
| 4 | Разработка компилятора языка программирования высокого уровня: языки описания предметной области | 4 | 2 | 3 | 6 |  |  | 1 | 24 | | | Защита лабораторной работе |
| 5 | Библиотеки и программные среды высокого уровня для разработки параллельных программ | 5 | 2 | 4 | 4 |  |  |  |  | | | Устный опрос |
| 6 | Обработка данных на многоядерных вычислительных архитектурах | 6 | 2 | 5 | 6 |  |  |  |  | | | Устный опрос |
|  | Промежуточная аттестация |  |  |  |  |  |  |  | 0 | | | Зачет |
|  | Всего |  | 12 |  | 24 |  |  |  | 72 | | |  |

**4.2. Краткое содержание разделов и тем занятий**

**Семестр №1**

|  |  |  |
| --- | --- | --- |
| **№** | **Тема** | **Краткое содержание** |
| 1 | Аппаратные архитектуры современных вычислительных средств | История эволюционирования компьютерных архитектур. Классификации архитектур. Архитектуры CISC, RISC и WLIW.Внутрипроцессорный обмен информацией, системные шины. Архитектуры Intel, PowerPC, ARMv7, MIPS, AVR. Специализированные и графические процессоры GPU/CUDA. |
| 2 | Структуры данных: интенсивный подход к повышению производительности вычислительных систем | Записи и тегирование. Ассоциативные массивы. Мультиотображения. Списки. Множества и мультимножества. Очереди и приоритетные очереди. Стек. Деревья и их разновидности. B -деревья. Хэш-таблицы. Кучи. Графы. Таблица ссылок. Управление динамической памятью. Сборка мусора. Библиотека GC. |
| 3 | Процессоры Intel и оптимизация исходного кода: экстенсивное повышение производительности вычислительных систем | История развития. Конвейер команд и гипертрейдинг. Архитектура процессоров Core i7. Набор инструкций обработки векторных данных. Средства компилятора GCC для представления циклических операций в виде векторных инструкций. |
| 4 | Разработка компилятора языка программирования высокого уровня: языки описания предметной области | Структура компилятора: лексический анализатор, синтаксический анализатор. Грамматики. Классификация грамматик по Ноаму Хомскому. Представления грамматик в форме БНФ. Компилятор компиляторов ANTLR4. Компиляция JIT, AOT, виртуальные машины Java, CIL, Parrot, LLVM. Библиотека LLVM. Язык Оберон и его грамматика. Синтаксические структуры языка Оберон. Оптимизация сгенерированного кода. Сборка запускаемых модулей. Форматы файлов ELF, EXE (MZ). Представление отладочной информации. |
| 5 | Библиотеки и программные среды высокого уровня для разработки параллельных программ | Структуры данных для распределенных и параллельных воспроизводительных вычислений. Библиотеки BLAS, ATLAS, LAPACK, ScaLAPACK. Средства параллельного программирования Intel. Библиотека Intel Threading Building Box. пользование библиотека в системах анализа данных R, Matlab и др. Средства программирования NVIDIA GPU Cuda OpenCL. |
| 6 | Обработка данных на многоядерных вычислительных архитектурах | Многоядерные процессорные архитектуры SMP и NUMA. Инструментальные средства программирования SMP-архитектур OpenMP. Многопоточные приложения. Суперкомпьютеры, кластеры, гриды и облачные вычисления. Метакопьютинг. Библиотека программирования кластерных ВС MPI и ее реализации. Системы MOSIX. |

**4.3. Перечень лабораторных работ**

**Семестр №1**

|  |  |  |
| --- | --- | --- |
| № п/п | Наименование лабораторной работы | Кол-во акад. часов |
| 1 | Исследование компилятора GCC | 2 |
| 2 | Эвристический поиск в графе пространства состояний | 4 |
| 3 | Разработка компилятора языка Оберон | 8 |
| 4 | Параллельная обработка массива при помощи технологии OpenMP | 4 |
| 5 | Разработка параллельной программы для кластерного вычислительного устройства | 6 |
|  | Итого | 24 |

**4.4. Перечень практических занятий**

Практических занятий не предусмотрено.

**4.5. Самостоятельная работа**

**Семестр №1**

|  |  |  |
| --- | --- | --- |
| № п/п | Вид СРС | Кол-во акад. часов |
| 1 | Подготовка к практическим занятиям (лабораторным работам) | 72 |
|  | Итого | 72 |

В ходе проведения лекций, практических и лабораторных работ используются интерактивные методы обучения, представленные в следующей таблице.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Технологии** | **Количество часов по учебному плану**  **(по видам занятий)** | | | | |
| **Лекции** | **ЛР** | **ПЗ (СЕМ)** | **СРС** | **КП (КР)** |
| 1 | *Интерактивные лекции* (применение слайд-материалов с последующей дискуссией по теме лекции);  1) визуальная демонстрация работы средств GCC.  2) полемическое взаимодействие в форме дискуссии. | 1 | 1 |  |  |  |
| 2 | *Опережающее обучение*  Постановка вопроса о возможности применения высокопроизводительных вычислительных систем на примере конкретной задачи, чтобы простимулировать студентов к анализу бизнес-процессов на производстве, где они работают, в рамках СРС | 1 | 1 |  |  |  |
| 3 | *Проблемное обучение* | 1 |  |  |  |  |
| 4 | *Работа в команде* – совместная разработка функциональных блоков лабораторной работы 3 |  | 1 |  |  |  |
| 6 | *Исследовательский метод* |  | 1 |  | 1 |  |
| 7 | Синхронное выполнение действий с преподавателем (с использованием ПК и мультимедиа-техники) | 1 | 1 |  | 1 |  |
| ИТОГО | | 4 | 5 |  | 2 |  |

**5. Перечень учебно-методического обеспечения дисциплины**

**5.1. Методические указания для обучающихся по освоению дисциплины**

Лекционные занятия предназначены для изложения аудитории основных теоретических положений разделов курса. Вводная лекция служит для создания общего впечатления о дисциплине, представления класса задач, а также оценке современного состояния перспектив развития современных вычислительных систем. На занятии до сведения учащегося доводятся основные вопросы дисциплины, показывается ее роль и место в соответствующей области знаний, определяется значение дисциплины для формирования компетенций. Для закрепления материала обучающимся на лекции предлагается провести дискуссию по темам:

* Роль высокопроизводительных вычислений в обществе, современной науке и образовании;
* Понятие интенсивного и экстенсивного подхода к решению задач в информатике;
* Использование дополнительной информации в переборных алгоритмах – путь к концептуальному повышению производительности;
* Архитектуры современных кластерных систем, варианты их использования.

Неотъемлемой частью изучения дисциплины «Вычислительные системы» является выполнение лабораторных работ, основной целью которых является выработка навыков проектирования программного обеспечения решения комбинаторных задач, создания параллельных программ, разработка собственных компиляторов, изучения технологий интегрирования высокопроизводительных средств с информационными системами, разработка приложений для обеспечения взаимодействия пользователя со сложной вычислительной системой.

**5.1.1. Методические указания для обучающихся по лабораторным работам**

В лабораторных работах студент должен выполнить конкретное задание (лабораторные работы 1-5). Предметную область лабораторных работ 2-5 студент выбирает самостоятельно, согласовывая с преподавателем.

Содержание лабораторных работ представлено в методических изданиях:

1. Ахо А. В. Структуры данных и алгоритмы [Электронный ресурс] : пер. с англ. / А. В. Ахо, Д. Э. Хопкрофт, Д. У. Ульман ; ред. А. А. Минько, 2000. - 384.
2. Гагарина Л. Г. Введение в теорию алгоритмических языков и компиляторов : учебное пособие для вузов по направлению 230100 "Информатика и вычислительная техника" / Л. Г. Гагарина , Е. В. Кокорева, 2011. - 175.
3. Малявко А. А. Параллельное программирование на основе технологий OpenMP, MPI, CUDA : учебное пособие для академического бакалавриата / А. А. Малявко, 2018. - 115.

**5.1.2. Методические указания для обучающихся по самостоятельной работе**

Совокупность методических указаний по выполнению заданий лабораторных работ и СРС, существующее открытое программное обеспечение, а также материалов конспекта лекций, обеспечивают достаточный объем информации для успешного освоения дисциплины.

**Подготовка к лабораторным работам**

**Цель работы:**

Изучение основных методов, методик и технологий реализации программ на электронных вычислительных машинах различной архитектуры,

**Содержание задания на СРС**

В ходе подготовки к лабораторным занятиям студент должен самостоятельно изучить основные инструменты и технологии, используемые в процессе разработки программ, для чего необходимо найти и освоить справочную документацию соответствующую темам лабораторных работ.

**Темы для поиска и изучения справочной документации**

Для каждой лабораторной работы предусмотрен свой список тем для поиска и изучения справочной документации.

Лабораторная работа № 1: Компиляторы GCC, CLANG, Intel C Compiler.

Лабораторная работа № 2: Алгоритм А\*, Метод ветвей и границ, эволюционные алгоритмы: генетические алгоритмы, алгоритмы муравья, отжига и т.п.

Лабораторная работа № 3: Изучить или разработать грамматику БНФ для языка Оберон или другого понравившегося языка программирования или собственного языка описания предметной области.

Лабораторная работа № 4: Найти в своей библиотеке или реализовать программу вычисления над векторами и матрицами. Добавить к этой программе инструкции OpenMP.

Лабораторная работа № 5: Изучить методический материал, разработать стратегию реализации варианта программы на учебном кластере.

**Порядок выполнения**

1. Ознакомиться с заданием на текущую лабораторную работу.
2. Найти и самостоятельно изучить справочную информацию в соответствии с темой лабораторной работы
3. Подготовить вопросы преподавателю по теме выполнения лабораторной работы.

**6. Фонд оценочных средств для контроля текущей успеваемости и проведения промежуточной аттестации по дисциплине**

**6.1. Оценочные средства для проведения текущего контроля**

**6.1.1. Входной контроль (ВК)**

**Описание процедуры:**

Студентам предлагается ответить на несколько вопросов по различным разделам информатики. Процедура проводится в начале первой лекции. Цель процедуры – определить готовность аудитории воспринимать материал курса.

**Пример:**

В аудиторию задается ряд вопросов, например

1. В чем состоит суть решения проблем при помощи информатики?
2. Покажите примеры использования системного подхода при проектировании программного обеспечения.
3. Дайте определение интуитивному понятию алгоритма.
4. Приведите примеры алгоритмически неразрешимых проблем.

***Критерии оценки:***

Группа студентов считается сдавшей опрос, если на все вопросы получены адекватные ответы. В обратном случае уделяется некоторое время лекции на разъяснение заданий, вызвавших затруднение.

**6.1.2. Защита лабораторных работ**

**Тема (раздел)**

«Исследование компилятора GCC» (лабораторная работа № 1), «Эвристический поиск в графе пространства состояний» (лабораторная работа № 2), «Разработка компилятора языка Оберон» (лабораторная работа № 3), «Параллельная обработка массива при помощи технологии OpenMP» (лабораторная работа № 4), «Разработка параллельной программы для кластерного вычислительного устройства» (лабораторная работа № 5).

**Описание процедуры:**

Для успешной сдачи лабораторной работы студенту необходимо защитить результат - программу. В ходе защиты отчета студенту необходимо дать краткое изложение основных результатов, полученных в ходе выполнения лабораторной работы, показать работоспособность программы, устно ответить на теоретические вопросы по теме лабораторной работы, а также продемонстрировать умение ориентироваться в полученных результатах выполнения.

**Примеры вопросов для контроля:**

Лабораторная работа № 1:

1. Продемонстрируйте работу компилятора на примере.
2. Какие функциональные блоки включает компилятор GCC?
3. Какова общая синтаксическая структура языка программирования С?

Лабораторная работа № 2:

1. Дайте определение графу пространства состояний.
2. Как задаются эвристические функции?
3. Зачем использовать эвристические функции при решении переборных задач?

Лабораторная работа № 3:

1. Прокомментируйте тот факт, что синтаксис большинства языков программирования соответствует контекстно-свободной грамматике?
2. Что такое компилятор языка высокого уровня?
3. Какие основные функции реализует библиотека LLVM?

Лабораторная работа № 4:

1. В чем суть применения технологии OpenMP при реализации программ?
2. Каким образом задаются инструкции OpenMP в GCC?
3. Как организовать параллельную обработку случайных событий?

Лабораторная работа № 5:

1. Что такое вычислительный кластер?
2. Как работает буферизация данных в MPI?
3. Какие функции библиотеки MPI используются для неблокирующей передачи информации между узлами?

***Критерии оценки:***

Лабораторная работа считается сданной, если разработанная программа функционирует правильно на примерах, заданных преподавателем, а в ходе ответа на контрольные вопросы студент демонстрирует знание и понимание теоретического материала необходимого для выполнения работ, а также свободно ориентируется в отчетных материалах.

**6.1.3. Устный опрос**

**Описание процедуры:**

Студентам предлагается ответить на несколько вопросов. Процедура проводится в начале лекции, следующей за циклом лекций, представляющем тематику опроса.

**Пример:**

В аудиторию задается ряд вопросов, например

1. Из каких функциональных блоков состоит современный компилятор языка программирования высокого уровня?
2. Какие существуют общие модели вычислительных устройств?
3. Приведите пример структуры, соответствующей сбалансированному двоичному дереву.
4. Какие синтаксические структуры используются для задания параллельных вычислительных процессов OpenMP в языке C?

***Критерии оценки:***

Группа студентов считается сдавшей опрос, если на все вопросы получены адекватные ответы. В обратном случае уделяется некоторое время лекции на разъяснение заданий, вызвавших затруднение.

**6.2. Оценочные средства промежуточной аттестации**

**6.2.1. Показатели и критерии оценивания компетенций на этапах их формирования**

|  |  |  |  |
| --- | --- | --- | --- |
| **Код этапа освоения компетенции** | **Показатель оценивания** | **Критерий оценивания** | **Средства (методы) оценивания промежуточной аттестации** |
| ОК-2.1 | Знание свойств различных архитектур вычислительных систем.  Умение использовать структуры данных для повышения производительности алгоритмов.  Навыки проведения научных исследований с использованием вычислительной техники. | Уверенно демонстрирует полученные знания согласно показателям, приводит примеры, отвечает на вопросы. Работает со специализированными программными средствами. Способен выявлять в программе места, требующие повышения производительности при помощи внедрения новых структур данных и алгоритмов. | Защита лабораторных работ, устное собеседование по теоретическим вопросам |
| ОК-8.1 | Знание основных принципов построения программных систем и использование системного подхода к проектированию программного обеспечения.  Умение реализовать программную систему, используя классические методы проектирования.  Навыки реализации параллельных схем алгоритмов на многоядерных вычислительных архитектурах. | Уверенно демонстрирует полученные знания согласно показателям, приводит примеры, отвечает на вопросы. Использует классические методы построения программ. Способен реализовывать решения вычислительных, переборных и оптимизационных задач на различных вычислительных архитектурах. | Защита лабораторных работ, устное собеседование по теоретическим вопросам |

**6.2.2. Типовые оценочные средства промежуточной аттестации**

**6.2.2.1. Типовые оценочные средства для проведения зачета по дисциплине**

***Контрольные вопросы по дисциплине***

1. Перечислить и охарактеризовать современные аппаратные архитектуры популярных вычислительных средств.
2. Показать основные отличие популярных кластерных и микроконтроллерных архитектур от Д. фон Неймановских архитектуры.
3. Показать основные этапы эволюции вычислительных архитектур.
4. Задать варианты классификации архитектур, показать примеры.
5. Особенности архитектуры процессоров и систем команд CISC, RISC и WLIW.
6. Дать характеристику разновидностям внутрипроцессорного обмена информацией для разных архитектур процессоров.
7. Охарактеризовать архитектуры Intel, PowerPC, ARMv7, MIPS, AVR.
8. Специализированные и графические процессоры GPU/CUDA. Перечислить основные особенности и классы задач, эффективно решаемых на этой архитектуре.
9. Дать понятия модели данных, структур данных и метаинформации.
10. Перечислить сложные динамические структуры данных, используемые для построения эффективных процедур дискретной математики.
11. Охарактеризовать основные структуры данных для повышения производительности доступа к данным: записи и тегирование, ассоциативные массивы, B+-деревья, хэш-таблица, куча.
12. Показать эффективные (по количеству операций и/или объему используемой памяти) реализации мультиотображений, списков, множеств, мультимножеств, деревьев, графов.
13. Особенности использования очередей и приоритетных очередей, стеков, деревиев в имитационном моделировании.
14. Охарактеризовать методы управления динамической памятью и их реализации.
15. Область применения процедуры сборки мусора при использовании динамической памяти. Перечислить популярные библиотеки для реализации сборки мусора.
16. Особенности процессоров Intel и оптимизации исходного кода для эффективного использования его ресурсов.
17. Перечислить особенности конвейерного исполнения команд процессора Intel Pentium.
18. Особенности архитектур процессоров Core i7.
19. Перечислить и охарактеризовать набор инструкций обработки векторных данных процессора Intel.
20. Показать на примере использование средства компилятора GCC для представления циклических операций в виде векторных инструкций.
21. Что такое компилятор для языка программирования? Из каких функциональных блоков он состоит?
22. Охарактеризовать структуру языка задания компилятора ANTLR4.
23. Дать определение Just ini time – компиляции и Ahead of time – компилфции.
24. Перечислить особенности виртуальной машины LLVM.
25. Перечислить современные методы, подходы и средства разработки параллельных программ.
26. Показать особенности реализации структур данных, используемых в распределенных и параллельных высокопроизводительных вычислений.
27. Охарактеризовать библиотеки BLAS, ATLAS, LAPACK, ScaLAPACK как средства разработки параллельных программ.
28. Перечислить средства параллельного программирования фирмы Intel.
29. Показать особенности библиотеки Intel Threading Building Box как средства многопоточного программирования.
30. Использование специализированных библиотек в системах анализа данных R, Matlab и др.
31. Перечислить методики и средства программирования на основе NVIDIA GPU Cuda OpenCL.
32. Перечислить методы обработки данных на многоядерных вычислительных архитектурах.
33. Охарактеризовать многоядерные процессорные архитектуры SMP и NUMA, показать их основные различия и особенности.
34. Инструментальные средства программирования SMP-архитектур OpenMP.
35. Показать особенности параллельного программирования для кластерных систем.
36. Методы построения аппаратных архитектур суперкомпьютеров, кластеров, гридов и систем облачных вычислений.
37. Понятие метакопьютинга, методы и средства реализации.
38. Библиотеки и инструментальные средства программирования кластерных вычислительных систем MPI и ее реализации.
39. Операционная система MOSIX, особенности проектирования программ для этой операционной системы.

Таблица распределения вопросов по компетенциям

|  |  |
| --- | --- |
| Компетенция | Номера вопросов |
| ОК-2.1 | 3-4,9-15,21-26,32-33,35-37 |
| ОК-8.1 | 1-2,5-8,16-20,27-31,34,38-39 |

**6.2.2.1.1. Описание процедуры зачета**

В ходе зачета проводится устное собеседование по теоретическим вопросам, решение практических задач. Студенту предлагается подготовить ответ на один из теоретических вопросов и/или решить задачу. Преподаватель может задавать уточняющие вопросы по теме теоретического вопроса, а также по существу практического задания.

**Пример теоретического вопроса:**

В чем состоит функция синтаксического анализатора компилятора языка высокого уровня?

**Пример ответа:**

Задача синтаксического анализа в компиляторе состоит в том, чтобы а) проверить соответствие входной цепочки лексем грамматике языка программирования, б) выделять в распознаваемых грамматических структурах информацию и передавать ее в блок преобразований (блок перевода на другой язык или внутренние структуры данных компилятора).

**Пример практического задания:**

Представьте в языке ANTLR4 следующую грамматическую БНФ-структуру:

module = MODULE ident ";" [ImportList] DeclarationSequence [BEGIN StatementSequence] END ident ".".

**Пример ответа:**

В данной структуре присутствуют обязательные и необязательные компоненты. Кроме того, правило включает как терминальные так и нетерминальные символы. Поэтому в ANTLR4 данная структура будет выглядеть следующим образом:

module:

MODULE ident SEMICOLON ( importList )? declarationSequence

( BEGIN statementSequence )? END ident DOT;

Здесь SEMICOLON и DOT, соответственно, обозначают символы «;» и «.».

**6.2.2.1.2. Критерии оценивания**

|  |  |
| --- | --- |
| **Зачтено** | **Не зачтено** |
| Способен демонстрировать наличие знания пройденного программного материала при успешном применении его на практике. Допускаются некоторые ошибки в выполненном практическом задании, уверенно исправляемые после дополнительных вопросов. | Наличие грубых ошибок в решении, при отсутствии возможности их устранения, непонимание сущности излагаемого вопроса. |

**7. Основная учебная литература**

1. Крупский, В. Н. Теория алгоритмов. Введение в сложность вычислений : учеб. пособие для бакалавриата и магистратуры / В. Н. Крупский. — 2-е изд., испр. и доп. — М. : Издательство Юрайт, 2019. — 117 с. — (Серия : Авторский учебник). — ISBN 978-5-534-04817-9. — Режим доступа : <https://www.biblio-online.ru/book/C76C6ABA-2603-445F-985B-C7C64FA296E3>.
2. Кубенский, А. А. Функциональное программирование : учебник и практикум для академического бакалавриата / А. А. Кубенский. — М. : Издательство Юрайт, 2019. — 348 с. — (Серия : Бакалавр. Академический курс). — ISBN 978-5-9916-9242-7. — Режим доступа : <https://www.biblio-online.ru/book/33D51687-6F35-422C-8769-A66C0BC4F5E8>.
3. Малявко, А. А. Параллельное программирование на основе технологий openmp, mpi, cuda : учеб. пособие для академического бакалавриата / А. А. Малявко. — 2-е изд., испр. и доп. — М. : Издательство Юрайт, 2019. — 116 с. — (Серия : Бакалавр. Академический курс). — ISBN 978-5-534-09825-9. — Режим доступа : <http://www.biblio-online.ru/book/ED89BDC3-0F8A-4D1E-A66C-39BC068CBDF9>.
4. Малявко, А. А. Формальные языки и компиляторы : учеб. пособие для вузов / А. А. Малявко. — М. : Издательство Юрайт, 2019. — 429 с. — (Серия : Университеты России). — ISBN 978-5-534-04288-7. — Режим доступа : <https://www.biblio-online.ru/book/702DB8E4-0A35-4676-B815-04059CB553AE>.

**8. Дополнительная учебная и справочная литература**

1. Вирт Никлаус. Алгоритмы и структуры данных / Никлаус Вирт; Пеp. с англ. Д. Б. Подшивалова, 1989. - 360.
2. Ахо А. В. Построение и анализ вычислительных алгоритмов / А. В. Ахо, Дж. Э. Хопкрофт, Д. Д. Ульман, 1979. - 536.
3. Вирт Никлаус. Алгоритмы + структуры данных = программы / Никлаус Вирт; Пеp. с англ. Л. Ю. Йоффе, 1985. - 406.
4. Гагарина Л. Г. Введение в теорию алгоритмических языков и компиляторов : учебное пособие для вузов по направлению 230100 "Информатика и вычислительная техника" / Л. Г. Гагарина , Е. В. Кокорева, 2011. - 175.
5. Ахо А. В. Структуры данных и алгоритмы [Электронный ресурс] : пер. с англ. / А. В. Ахо, Д. Э. Хопкрофт, Д. У. Ульман ; ред. А. А. Минько, 2000. - 384.
6. Тюкачев Н. А. C#. Алгоритмы и структуры данных : учебное пособие / Н. А. Тюкачев, В. Г. Хлебостроев, 2018. - 232.
7. Малявко А. А. Параллельное программирование на основе технологий OpenMP, MPI, CUDA : учебное пособие для академического бакалавриата / А. А. Малявко, 2018. - 115.
8. Малявко А. А. Формальные языки и компиляторы : Учебное пособие / Малявко А.А., 2018. - 429.
9. Фролов Александр Вячеславович. Защищенный режим процессоров Intel 80286/80386/80486 : практ. рук. по использованию защищ. режима / Александр Вячеславович Фролов, Григорий Вячеславович Фролов, 1993. - 234.
10. Шагурин И. И. Процессоры семейства Intel Р6 Pentium II, Pentium III, Celeron и др. : архитектура, программирование, интерфейс: [Справочник] / И. И. Шагурин, Е. М. Бердышев, 2000. - 244.
11. Ирвин Кип Р. Язык ассемблера для процессоров Intel : пер с. англ. / Кип Р. Ирвин, 2002. - 614.
12. Вирт Н. Построение компиляторов : учебное пособие / Н. Вирт; пер. с англ. Е. В. Борисова, Л. Н. Чернышова, 2010. - 192.

**9. Ресурсы сети Интернет**

1. <http://library.istu.edu/>
2. <https://e.lanbook.com/>
3. <http://www.gentoo.org/>
4. <https://github.com/>
5. <http://new.fips.ru/>

**10. Перечень информационных технологий, лицензионных и свободно распространяемых специализированных программных средств, информационных справочных систем**

1. Microsoft Windows (Подписка DreamSpark Premium Electronic Software Delivery (3 years). Сублицензионный договор №14527/МОС2957 от 18.08.16г.);

Свободное программное обеспечение:

1. Дистрибутив Arch Linux [http://archlinux.org](http://archlinux.org/);
2. Среда разработки IntelliJ Idea for Java, <https://www.jetbrains.com/idea/>;
3. Любая свободная или условно-бесплатная среда программирования, EMACS, <https://www.gnu.org/software/emacs/>.

**11. Материально-техническое обеспечение дисциплины**

Лекции по дисциплине проводятся в мультимедийном классе, оборудованном проектором и экраном (В-108). Лабораторные работы проводятся в учебно-исследовательской лаборатории аппаратных и программных средств вычислительной техники кафедры вычислительной техники (В-106/208), оборудованной проектором с экраном, 16 ПК).

1. Проектор ACER Х1261Р.DLP projector. XGA 1024\*768. Nvidia 3DDLP 3D. (оборуд. НИЧ, Дорофеев);
2. Проектор TOSHIBA TLP-X3000;
3. Класс персональных компьютеров В106 и В208 (по 16 пк).