Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«Национальный исследовательский ядерный университет «МИФИ»

ИНСТИТУТ ИНТЕЛЛЕКТУАЛЬНЫХ КИБЕРНЕТИЧЕСКИХ СИСТЕМ

КАФЕДРА КИБЕРНЕТИКИ

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ПРЕПОДАВАТЕЛЕЙ**

**ПО ДИСЦИПЛИНЕ**

ЯЗЫКИ ПРОГРАММИРОВАНИЯ И МЕТОДЫ ПРОГРАММИРОВАНИЯ

|  |  |
| --- | --- |
| Направление подготовки (специальность) | 09.03.04 Программная инженерия |
| Профиль подготовки | *[при его наличии]* |
| Наименование образовательной программы (специализация) | Математическое и программное обеспечение вычислительных машин и компьютерных сетей |
| Квалификация (степень) выпускника | бакалавр |
| Форма обучения | очная |

**АННОТАЦИЯ**

Дисциплина призвана обеспечить освоение студентами навыков и знаний проектирования программ, абстрактных и пользовательских типов данных и классов, документирования программ, а также получить знания и практический опыт освоения новых языков программирования. Студенты активно изучают продвинутые приемы объектно-ориентированного программирования. Изучаются продвинутые структуры данных и алгоритмические задачи, связанные поиском данных. Изучаются такие структуры, как деревья поиска, в т.ч. B и B+-деревья, хеш-таблицы, работа с графами. Дополнительно дисциплина позволяет получить студентам общее представление о промышленных методах разработки программного обеспечения и процессе тестирования программного обеспечения. В дисциплене также рассматриваются такие вопросы как алгоритмизация и сложность алгоритма, обработка ошибок в программном коде и особенности программирования при работе со строками. В рамках курса студенты выполняют индивидуальные задания, которые формируются на основе их собственного выбора, что обеспечивает элементы индивидуальности учебной траектории студента в рамках курса.

**СТРУКТУРА ДИСЦИПЛИНЫ**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Семестр** | **Интерактив** | **Трудоемкость, кред.** | **Общий объем курса, час.** | **Лекции, час.** | **Практич. занятия, час.** | **Лаборат. работы, час.** | **СРС, час.** | **КСР, час.** | **Форма(ы) контроля, экз./зач./КР/КП** |
| 2 |  | 3 | 108 | 32 | 0 | 32 | 44 | 0 | Э |
| ИТОГО | 0 | 3 | 108 | 32 | 0 | 32 | 44 | 0 |  |

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

**Указания к выполнению лабораторных работ**

По курсу предусмотрено выполнение трех лабораторных работ в формате самостоятельного выполнения индивидуального задания по каждой работе. Состав заданий и методические указания приводятся в соответствующих приложениях – по каждой лабораторной работе. На выполнение каждой лабораторной отводится по 5 недель. В процессе выполнения на любом этапе студент может получать консультации по всем вопросам, касающимся выполнения задания. Выполнение лабораторной работы оценивается по итогам защиты, которая проводится в форме устного собеседования с обязательной демонстрацией программы в действии и разбором исходного кода.

**Методика оценки результатов выполнения лабораторных работ**

Общая методика оценки выполнения индивидуального задания приведена ниже. В каждой лабораторной работе она может незначительно корректироваться.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  | ЛР1+2, ЛР-4 | ЛР-3 |
|  | Качество программного кода: | * стиль (в т.ч.: имена, отступы и проч.) (0-2) * структурированность (напр. декомпозиция сложных функций на более простые) (0-2) * качество основных и второстепенных алгоритмов (напр. обработка граничных случаев и некорректных исходных данных и т.п.) (0-3) | 0-4 баллов |  |
|  | Качество пользовательского интерфейса: | * предоставляемые им возможности (0-2) * наличие ручного/автоматического ввода исходных данных (0-2) * настройка параметров для автоматического режима   отображение исходных данных и промежуточных и конечных результатов и др. (0-2) | 0-4 баллов |  |
|  | Качество тестов | * степень покрытия * читаемость * качество проверки (граничные и некорректные значения, и др.) | 0-4 баллов |  |
|  | Полнота выполнения задания и качество ТЗ | Оценивается качество подготовки ТЗ, полнота выполнений минимальных требований | 0-4 баллов |  |
|  | Владение теорией | знание алгоритмов, области их применимости, умение сравнивать с аналогами, оценить сложность, корректность реализации | 0-3 баллов |  |
|  | Оригинальность реализации | оцениваются отличительные особенности конкретной реализации – например, общность структур данных, наличие продвинутых графических средств, средств ввода-вывода, интеграции с внешними системами и др. | 0-4 баллов |  |
|  | Итого | | 0-25  баллов | 0-30 баллов |

**Список вопросов к зачету**

1. Алгоритмы сортировки
   1. Алгоритмы сортировки: классификация
   2. Обменные сортировки: общий принцип, примеры, асимптотическая оценка сложности
   3. Сортировки выбором: общий принцип, примеры, асимптотическая оценка сложности
   4. Сортировки вставкой: общий принцип, примеры, асимптотическая оценка сложности
   5. Сортировка пузырьком и шейкерная сортировка: алгоритм, асимптотическая оценка сложности
   6. Сортировка подсчетом: алгоритм, асимптотическая оценка сложности
   7. Метод простых вставок: алгоритм, асимптотическая оценка сложности
   8. Метод простого выбора: алгоритм, асимптотическая оценка сложности. Возможности усовершенствования.
   9. Метод двоичных вставок: алгоритм, асимптотическая оценка сложности
   10. Сортировка слиянием: алгоритм, асимптотическая оценка сложности
   11. Пирамидальная сортировка: алгоритм, асимптотическая оценка сложности
   12. Быстрая сортировка: алгоритм, асимптотическая оценка сложности
   13. Сортировка Шелла: алгоритм, асимптотическая оценка сложности
   14. Сортировка Бэтчера: алгоритм, асимптотическая оценка сложности
2. Методы поиска
   1. Сбалансированное бинарное дерево поиска: принципы организации, реализация полиморфного дерева поиска на C++ (основные идеи)
   2. Сбалансированное бинарное дерево поиска (полиморфное): вставка элемента
   3. Сбалансированное бинарное дерево поиска (полиморфное): поиск элемента по ключу
   4. Сбалансированное бинарное дерево поиска (полиморфное): удаление элемента
   5. B-дерево: принципы организации, реализация полиморфного дерева поиска на C++ (основные идеи)
   6. B-дерево (полиморфное): алгоритм вставки элемента
   7. B-дерево (полиморфное): расщепление узла при вставке
   8. B-дерево (полиморфное): поиск элемента по ключу
   9. Полиморфный АТД «Множество»: реализация на базе хеш-таблиц. Асимптотическая оценка сложности вставки элемента и поиска (проверки на вхождение в множество).
   10. Полиморфный АТД «Множество»: реализация на базе дерева поиска (бинарного или B-дерева) . Асимптотическая оценка сложности вставки элемента и поиска (проверки на вхождение в множество).
   11. Понятие итератора, основные операции.
   12. Организация пространства состояний или решений в виде n-арного дерева: основные идеи
3. Ассоциативная память
   1. Хеш-таблицы. Хеш-функции. Разрешение коллизий в хеш-таблицах.
   2. Ассоциативная память, словарь. Варианты реализации: с помощью хеш-таблиц, деревьев поиска. Сравнение асимптотических оценок характеристик алгоритмов (вставка, поиск по ключу).
   3. Структуры данных для представления ориентированного графа с взвешенным дугами
   4. Структуры данных для представления неориентированного графа с взвешенным ребрами
   5. Обход ориентированного графа в ширину (алгоритм)
   6. Обход ориентированного графа в глубину (алгоритм)
   7. Обход неориентированного графа в ширину (алгоритм)
   8. Обход неориентированного графа в глубину (алгоритм)
   9. Полиморфный АТД «Очередь с приоритетами»: реализация на основе бинарной кучи, основные операции, асимптотическая оценка сложности
   10. Полиморфный АТД «Очередь с приоритетами»: реализация на основе связанного списка, основные операции, асимптотическая оценка сложности

Поиск кратчайших путей: алгоритм Дейкстры