Документ подписан простой электронной подписью
Информация о владельце:
ФИО: Максимов Алексей Борисович
Должность: директор департамента по образовательной политике
Дата подписания: 01.09.2023 12:26:52
Уникальный программный ключ:
8db180d1a3f02ac9e60521a5672742735c18b1d6

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

УТВЕРЖДЕНО
Декан факультета
Информационных технологий
_____ / Демидов Д.Г. /
«27» апреля 2022 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

«Программирование и основы алгоритмизации»

Направление подготовки
09.03.01 “Информатика и вычислительная техника”

Образовательная программа (профиль подготовки)
«Киберфизические системы»

Квалификация (степень) выпускника
Бакалавр

Форма обучения
Очная

Москва 2022 г.

Программа дисциплины «**Программирование и основы алгоритмизации**» составлена в соответствии с требованиями ФГОС ВО и учебным планом по направлению **09.03.01 «Информатика и вычислительная техника»** профилю подготовки **«Киберфизические системы».**

Программу составил:
к.ф.-м.н. _____ /Т.Т. Идиатуллов/

Программа дисциплины «**Программирование и основы алгоритмизации**» по направлению **09.03.01 «Информатика и вычислительная техника»** по профилю подготовки **«Киберфизические системы»** утверждена на заседании кафедры «СМАРТ-технологии» «26» апреля 2022 г. протокол № 8

И.О. Зав. кафедрой _____ /Я.В. Береснева/

1. Цели освоения дисциплины

Целью освоения дисциплины «Программирование и основы алгоритмизации»является формирование системы знаний, умений и навыков в области основ алгоритмизации и прикладного программирования.

Задачи дисциплины: изучение принципов построения алгоритмов, изучение основ алгоритмических конструкций, изучение процедурного языка программирования С, изучение методов построения алгоритмов и структур данных, используемых при решении прикладных задач в различных предметных областях с применением ЭВМ.

2. Место дисциплины в структуре ООП бакалавриата

Дисциплина «Программирование и основы алгоритмизации» относится к дисциплинам базовой части (Блока 1) Б.1.1.11. основной образовательной программы бакалавриата; изучается во 3 и 4 семестрах.

Дисциплина базируется на следующих, пройденных дисциплинах:

- «Информационные технологии»;
- «Математика».

3.Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенные с планируемыми результатами освоения образовательной программы

Код компетенции	**В результате освоения образовательной программы обучающийся должен обладать**	**Перечень планируемых результатов обучения по дисциплине**
ОПК-8	Способность разрабатывать алгоритмы и программы, пригодные для практического применения	**Знать:** - основы алгоритмизации (свойства алгоритмов, область применения алгоритмов); - методы построения алгоритмов; - структуры данных; - синтаксис и семантику универсального алгоритмического языка программирования высокого уровня; - основные принципы и методологию разработки прикладного программного обеспечения; - типовые способы организации программных данных; - подходы к построению программных алгоритмов. **Уметь:** - строить блок-схемы алгоритмов; - проводить анализ эффективности алгоритмов; - описать алгоритм, используя ключевые слова языков

		программирования, но опуская подробности и специфический синтаксис (псевдокод). **Владеть:** - навыками составления алгоритмов и их представления в виде блок-схем; - навыками формализации прикладных задач.
ПК-1	Способностью разрабатывать и отлаживать программный код	**Знать:** - инструменты разработки программного обеспечения; - виды структур данных; - диаграммы проектирования программного обеспечения; - стадии разработки программного обеспечения. **Уметь:** - уметь согласованно решать задачи разработки эффективных моделей данных и алгоритмов их обработки при создании прикладного программного обеспечения; - получать программные реализации полученных решений на процедурном языке программирования С. **Владеть:** - навыками использования инструментальных программных средств в процессе разработки программного обеспечения; - навыками построения проектирования программного обеспечения; - навыками разработки программного обеспечения.

4. Структура и содержание дисциплины

Общая трудоемкость дисциплины составляет 4 зачетных единицы, 144 академических часов (из них 72 часа – аудиторная работа, в том числе 18 часов лекций, 54 часа лабораторных занятий, и 72 часа самостоятельной работы студента).

Дисциплина изучается в 1 семестре (1 курс). Форма контроля - экзамен.

Структура и содержание дисциплины «Программирование и основы алгоритмизации» по срокам и видам работы отражены в приложении.

Содержание разделов дисциплины

Тематика лекционных занятий:

Тема 1.*Введение в процедурный язык программирования С.* Переменные и арифметика. Условные операторы. Операторы цикла. Символические константы. Ввод и вывод.

Тема 2.*Типы, операции и выражения.* Имена переменных. Типы и размеры данных. Константы. Арифметические операции. Операции отношения и логические операции. Преобразование типов. Побитовые логические операции.

Тема 3.*Поток управления.* Операторы и блоки. Условный оператор IF - ELSE. Циклы - WHILE, FOR, DO - WHILE. Оператор BREAK. Оператор CONTINUE.

Тема 4.*Функции и структура программ.* Основные сведения. Правила, определяющие область действия. Статические переменные. Инициализация. Препроцессор языка С. Включение файлов. Макроподстановка.

Тема 5.*Указатели и массивы.* Указатели и адреса. Указатели и аргументы функций. Указатели и массивы. Адресная арифметика. Указатели символов и функции. Многомерные массивы. Массивы указателей. Указатели указателей. Инициализация массивов указателей. Указатели и многомерные массивы. Командная строка аргументов. Указатели на функции.

Тема 6.*Структуры. Основные сведения.* Структуры и функции. Массивы структур. Указатели на структуры. Структуры, ссылающиеся на себя.

Тема 7.*Ввод и вывод.* Обращение к стандартной библиотеке. Стандартный ввод и вывод - функции GETCHAR и PUTCHAR. Форматный вывод - функция PRINTF. Форматный ввод - функция SCANF. Форматное преобразование в памяти. Доступ к файлам. Обработка ошибок - STDERR и EXIT. Ввод и вывод строк.

Тема 8.*Введение математического моделирования.* Базовые определения. Основные этапы математического моделирования. Прямые и обратные задачи математического моделирования.

Тема 9.*Численные методы.* Методы решения математических задач при помощи моделирования случайных величин. Методы приближённого вычисления определённого интеграла. Решение нелинейных уравнений.

Тема 10.*Основы алгоритмизации.* Понятие алгоритма. Основные характеристики алгоритмов. Способы описания алгоритмов. Структуры алгоритмов.

Тема 11.*Структуры данных.* Линейные однонаправленные (двунаправленные) списки. Древовидные структуры. Графы.

Тема 12.*Алгоритмы сортировок и поиска.* Линейный поиск. Двоичный поиск в упорядоченных данных. Сортировка выбором. Сортировка вставками. Вставка погружением. Сортировка Шелла.

Тема 13.*Основы проектирования программного обеспечения.* Диаграмма вариантов использования. Диаграмма классов. Диаграмма состояний. Диаграмма последовательности. Диаграмма кооперации. Диаграмма компонентов. Диаграмма развертывания.

Тема 14.*Стадии разработки программного обеспечения.* Начальный этап. Внутреннее тестирование. Публичное тестирование. Предварительная версия. Финальная версия. Пост-релиз. Общая доступность. Прекращение поддержки.

Тема 15.*Методологии разработки программного обеспечения.* Прогнозируемые методологии. Адаптивные методологии. SCRUM.

Тематика лабораторных работ

Лабораторная работа № 1. Изучение синтаксиса процедурного языка программирования С.

Лабораторная работа № 2. Изучение компилятора GCC.

Лабораторная работа № 3. Изучение циклов и условных операторов процедурного языка программирования С.

Лабораторная работа № 4. Изучение указателей и массивов процедурного языка программирования С.

Лабораторная работа № 5. Изучение структур процедурного языка программирования С.

Лабораторная работа № 6. Изучение методов ввода и вывода процедурного языка программирования С.

Лабораторная работа № 7. Изучение численных методов.

Лабораторная работа № 8. Изучение структур данных.

Лабораторная работа № 9. Изучение методов сортировок.

Лабораторная работа № 10. Изучение пакетов проектирование программного обеспечения.

Для проведения лабораторных работ требуется компьютерный класс. объеденный в локальную сеть с выходом в Интернет. Компьютеры должны работать под управлением операционной системой Linux и объединены локальной сетью. Необходим выход в всемирную систему объединённых компьютерных сетей для хранения и передачи информации(Интернет). Требуемое программное обеспечением: компилятор GCC, текстовый редактор, текстовый интерфейс общения с операционной системой (Shell), офисный пакет LibreOffice. Компьютерный класс должен иметь возможность обновления и установки дополнительного свободно распространяемого программного обеспечения.

Тематика вопросов для самостоятельного изучения

3 семестр.
Изучение тенденции применения различных языков программирования при решении разных практических задач.
4 семестр.
Изучение сред разработки, систем управления версиями.

5. Образовательные технологии

Методика преподавания дисциплины «Программирование и основы алгоритмизации» и реализация компетентностного подхода в изложении и восприятии материала предусматривает использование следующих активных и интерактивных форм проведения аудиторных и внеаудиторных занятий:

– аудиторные занятия: лекции, лабораторные работы, тестирование;

– внеаудиторные занятия: самостоятельное изучение отдельных вопросов, подготовка к лабораторным работам.

6. Оценочные средства для текущего контроля успеваемости, промежуточной аттестации по итогам освоении дисциплины и учебно-методическое обеспечение самостоятельной работы студентов

В процессе обучения в течение семестра используются оценочные средства текущего контроля успеваемости и промежуточных аттестаций. Применяются следующие оценочные средства: тест, защита лабораторных работ, зачет, экзамен.

Образцы тестовых заданий и вопросов к экзамену и зачету приведены в приложении 2.

6.1. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине (модулю).

6.1.1. Перечень компетенций с указанием этапов их формирования в процессе освоения образовательной программы.

В результате освоения дисциплины (модуля) формируются следующие компетенции:

Код компетенции	В результате освоения образовательной программы обучающийся должен обладать
ОПК-8	Способность разрабатывать алгоритмы и программы, пригодные для практического применения
ПК-1	Способностью разрабатывать и отлаживать программный код

В процессе освоения образовательной программы данная компетенция, в том числе их отдельные компоненты, формируются поэтапно в ходе освоения обучающимися дисциплин (модулей), практик в соответствии с учебным планом и календарным графиком учебного процесса.

6.1.2. Описание показателей и критериев оценивания компетенций, формируемых по итогам освоения дисциплины (модуля), описание шкал оценивания.

Показателем оценивания компетенций на различных этапах их формирования является достижение обучающимися планируемых результатов обучения по дисциплине (модулю).

ОПК-8 Способность разрабатывать алгоритмы и программы, пригодные для практического применения

Показатель	Критерии оценивания			
	2	**3**	**4**	**5**
<u>Знать:</u> - основы алгоритмизации (свойства алгоритмов, область применения алгоритмов); - методы построения алгоритмов; - структуры данных; - синтаксис и семантику универсального алгоритмического языка программирования высокого уровня; - основные принципы и методологию разработки прикладного программного обеспечения; - типовые способы организации программных данных; - подходы к построению программных алгоритмов.	Обучающийся демонстрирует полное отсутствие или недостаточное соответствие следующих знаний: основ алгоритмизации; методов построения алгоритмов, структур данных; синтаксиса и семантики универсального алгоритмического языка программирования высокого уровня; основ принципов и методологии разработки прикладного программного обеспечения; типовых способов организации программных данных; подходов к построению программных алгоритмов.	Обучающийся демонстрирует неполное соответствие следующих знаний:основ алгоритмизации; методов построения алгоритмов, структур данных; синтаксиса и семантики универсального алгоритмического языка программирования высокого уровня; основ принципов и методологии разработки прикладного программного обеспечения; типовых способов организации программных данных; подходов к построению программных алгоритмов. Допускаются значительные ошибки, проявляется недостаточность знаний, по ряду	Обучающийся демонстрирует частичное соответствие следующих знаний: основ алгоритмизации; методов построения алгоритмов, структур данных; синтаксиса и семантики универсального алгоритмического языка программирования высокого уровня; основ принципов и методологии разработки прикладного программного обеспечения; типовых способов организации программных данных; подходов к построению программных алгоритмов. Допускаются незначительные ошибки, неточности, затруднения при	Обучающийся демонстрирует полное соответствие следующих знаний: основ алгоритмизации; методов построения алгоритмов, структур данных; синтаксиса и семантики универсального алгоритмического языка программирования высокого уровня; основ принципов и методологии разработки прикладного программного обеспечения; типовых способов организации программных данных; подходов к построению программных алгоритмов. Свободно оперирует приобретенными знаниями.

		показателей, обучающийся испытывает значительные затруднения при оперировании знаниями при их переносе на новые ситуации.	аналитических операциях.	
Уметь: - строить блок-схемы алгоритмов; - проводить анализ эффективности алгоритмов; - описать алгоритм, используя ключевые слова языков программирования, но опуская подробности и специфический синтаксис (псевдокод).	Обучающийся не умеет или в недостаточной степени умеет: осуществлять синтез - строить блок-схемы алгоритмов; проводить анализ эффективности алгоритмов; описать алгоритм, используя ключевые слова языков программирования, но опуская подробности и специфический синтаксис (псевдокод).	Обучающийся демонстрирует неполное соответствие следующих умений: осуществлять синтез - строить блок-схемы алгоритмов; проводить анализ эффективности алгоритмов; описать алгоритм, используя ключевые слова языков программирования, но опуская подробности и специфический синтаксис (псевдокод).	Обучающийся демонстрирует частичное соответствие следующих умений: осуществлять синтез - строить блок-схемы алгоритмов; проводить анализ эффективности алгоритмов; описать алгоритм, используя ключевые слова языков программирования, но опуская подробности и специфический синтаксис (псевдокод).	Обучающийся демонстрирует полное соответствие следующих умений:осуществлять синтез - строить блок-схемы алгоритмов; проводить анализ эффективности алгоритмов; описать алгоритм, используя ключевые слова языков программирования, но опуская подробности и специфический синтаксис (псевдокод).
Владеть: - навыками составления алгоритмов и их представления в виде блок-схем; - навыками формализации	Обучающийся не владеет или в недостаточной степени владеет навыками составления алгоритмов и	Обучающийся владеет навыками составления алгоритмов и их представления в виде блок-схем; навыками формализации	Обучающийся частично владеет навыками составления алгоритмов и их представления в виде блок-схем; навыками формализации	Обучающийся в полном объеме владеет навыками составления алгоритмов и их представления в виде блок-схем;

прикладных задач.	их представления в виде блок-схем; навыками формализации прикладных задач.	прикладных задач.	прикладных задач.	навыками формализации прикладных задач.
ПК-1 Способность разрабатывать и отлаживать программный код				
Знать: - инструменты разработки программного обеспечения; - виды структур данных; - диаграммы проектирования программного обеспечения; - стадии разработки программного обеспечения.	Обучающийся демонстрирует полное отсутствие или недостаточное соответствие следующих знаний: инструментов разработки программного обеспечения; видов структур данных; диаграмм проектирования программного обеспечения; стадий разработки программного обеспечения.	Обучающийся демонстрирует неполное соответствие следующих знаний:инструментов разработки программного обеспечения; видов структур данных; диаграмм проектирования программного обеспечения; стадий разработки программного обеспечения. Допускаются значительные ошибки, проявляется недостаточность знаний, по ряду показателей, обучающийся испытывает значительные затруднения при оперировании знаниями при их переносе на новые ситуации.	Обучающийся демонстрирует частичное соответствие следующих знаний:инструментов разработки программного обеспечения; видов структур данных; диаграмм проектирования программного обеспечения; стадий разработки программного обеспечения. Допускаются незначительные ошибки, неточности, затруднения при аналитических операциях.	Обучающийся демонстрирует полное соответствие следующих знаний:инструментов разработки программного обеспечения; видов структур данных; диаграмм проектирования программного обеспечения; стадий разработки программного обеспечения. Свободно оперирует приобретенными знаниями.

Уметь: - согласованно решать задачи разработки эффективных моделей данных и алгоритмов их обработки при создании прикладного программного обеспечения; - получать программные реализации полученных решений на процедурном языке программирования С.	Обучающийся не умеет или в недостаточной степени умеет согласованно решать задачи разработки эффективных моделей данных и алгоритмов их обработки при создании прикладного программного обеспечения; получать программные реализации полученных решений на процедурном языке программирования С.	Обучающийся демонстрирует неполное соответствие следующих умений: согласованно решать задачи разработки эффективных моделей данных и алгоритмов их обработки при создании прикладного программного обеспечения; получать программные реализации полученных решений на процедурном языке программирования С. Обучающийся испытывает значительные затруднения при оперировании умениями при их переносе на новые ситуации.	Обучающийся демонстрирует частичное соответствие следующих умений: согласованно решать задачи разработки эффективных моделей данных и алгоритмов их обработки при создании прикладного программного обеспечения; получать программные реализации полученных решений на процедурном языке программирования С. Умения освоены, но допускаются незначительные ошибки, неточности, затруднения при аналитических операциях, переносе умений на новые, нестандартные ситуации.	Обучающийся демонстрирует полное соответствие следующих умений:согласованно решать задачи разработки эффективных моделей данных и алгоритмов их обработки при создании прикладного программного обеспечения; получать программные реализации полученных решений на процедурном языке программирования С. Свободно оперирует приобретенными умениями, применяет их в ситуациях повышенной сложности.
Владеть: - навыками использования инструментальных программных средств в	Обучающийся не владеет или в недостаточной степени владеет	Обучающийся владеет навыками навыками использования инструментальн	Обучающийся частично владеет навыками навыками использования инструментальн	Обучающийся в полном объеме владеет навыками навыками использования

процессе разработки программного обеспечения; - навыками построения проектирования программного обеспечения; - навыками разработки программного обеспечения.	навыками использования инструментальных программных средств в процессе разработки программного обеспечения; навыками построения проектирования программного обеспечения; навыками разработки программного обеспечения.	ых программных средств в процессе разработки программного обеспечения; навыками построения проектирования программного обеспечения; навыками разработки программного обеспечения. Обучающийся испытывает значительные затруднения при применении навыков в новых ситуациях.	ых программных средств в процессе разработки программного обеспечения; навыками построения проектирования программного обеспечения; навыками разработки программного обеспечения, но допускаются незначительные ошибки, неточности, затруднения при аналитических операциях, переносе умений на новые, нестандартные ситуации.	инструментальных программных средств в процессе разработки программного обеспечения; навыками построения проектирования программного обеспечения; навыками разработки программного обеспечения, свободно применяет полученные навыки в ситуациях повышенной сложности.

Шкалы оценивания результатов промежуточной аттестации и их описание:

Форма промежуточной аттестации: зачет.

Обязательными условиями подготовки студента к промежуточной аттестации является выполнение и защита студентом лабораторных работ, предусмотренных рабочей программой.

Шкала оценивания	**Описание**
Зачтено	Выполнены все обязательные условия подготовки студента к промежуточной аттестации, предусмотренные программой дисциплины. Студент демонстрирует соответствие знаний, умений, навыков приведенным в таблицах показателей, оперирует приобретенными знаниями, умениями, навыками, применяет их в ситуациях повышенной сложности. При этом могут быть допущены незначительные ошибки, неточности, затруднения при аналитических операциях, переносе знаний и умений на новые, нестандартные ситуации.

Не зачтено	Не выполнены обязательные условия подготовки студента к промежуточной аттестации, предусмотренные программой дисциплины, или студент демонстрирует неполное соответствие знаний, умений, навыков приведенным в таблицах показателей, допускаются значительные ошибки, проявляется отсутствие знаний, умений, навыков по ряду показателей, студент испытывает значительные затруднения при оперировании знаниями и умениями при их переносе на новые ситуации.

Форма промежуточной аттестации: экзамен.

Обязательными условиями подготовки студента к промежуточной аттестации является выполнение и защита студентом лабораторных работ, предусмотренных рабочей программой.

Шкала оценивания	**Описание**
Отлично	*Выполнены все виды учебной работы, предусмотренные рабочей программой. Студент демонстрирует соответствие знаний, умений, навыков приведенным в таблицах показателей, оперирует приобретенными знаниями, умениями, навыками, применяет их в ситуациях повышенной сложности. При этом могут быть допущены незначительные ошибки, неточности, затруднения при аналитических операциях, переносе знаний и умений на новые, нестандартные ситуации.*
Хорошо	*Выполнены все виды учебной работы, предусмотренные учебным планом.. Студент демонстрирует неполное, правильное соответствие знаний, умений, навыков приведенным в таблицах показателей, либо если при этом были допущены 2-3 несущественные ошибки.*
Удовлетворительно	*Выполнены все виды учебной работы, предусмотренные учебным планом. Студент демонстрирует соответствие знаний, в котором освещена основная, наиболее важная часть материала, но при этом допущена одна значительная ошибка или неточность.*
Неудовлетворительно	*Не выполнен один или более видов учебной работы, предусмотренных учебным планом. Студент демонстрирует неполное соответствие знаний, умений, навыков приведенным в таблицах показателей, допускаются значительные ошибки, проявляется отсутствие знаний, умений, навыков по ряду показателей, студент испытывает значительные затруднения при оперировании знаниями и умениями при их переносе на новые ситуации.*

Фонд оценочных средств представлен в приложении 2 к рабочей программе.

7. Учебно-методическое и информационное обеспечение дисциплины

а) Основная литература:

1. Язык Си и особенности работы с ним: учебное пособие. Костюкова Н. И., Калинина Н. А. (http://www.knigafund.ru/books/197466)

Интернет-Университет Информационных Технологий 2006 г.

2. Алгоритмизация прикладных задач: учебное пособие. Долгов А. И. Флинта 2011 г. (http://www.knigafund.ru/books/179100)

б) Дополнительная литература:

1. Программирование на языке Си : Методические рекомендации и задачи по программированию. Костюкова Н. И. Сибирское университетское издательство 2003 г. (http://www.knigafund.ru/books/178192)

б) Интернет ресурсы:

1. https://habrahabr.ru/
2. https://tproger.ru/tag/c-language/
3. https://prog-cpp.ru/c/

8. *Материально-техническое обеспечение дисциплины.*

Для проведения лабораторных работ требуется компьютерный класс. объеденный в локальную сеть с выходом в Интернет. Компьютеры должны работать под управлением операционной системой Linux и объединены локальной сетью. Необходим выход в всемирную систему объединённых компьютерных сетей для хранения и передачи информации(Интернет). Требуемое программное обеспечением: компилятор GCC, текстовый редактор, текстовый интерфейс общения с операционной системой (Shell), офисный пакет LibreOffice. Компьютерный класс должен иметь возможность обновления и установки дополнительного свободно распространяемого программного обеспечения.

9. Методические рекомендации для самостоятельной работы студентов

Самостоятельная работа студентов направлена на решение следующих задач:

Самостоятельная работа является одним из видов учебных занятий. Цель самостоятельной работы – практическое усвоение студентами вопросов автоматизации управления жизненным циклом изделия, рассматриваемых в процессе изучения дисциплины.

Аудиторная самостоятельная работа по дисциплине выполняется на учебных занятиях под непосредственным руководством преподавателя и по его заданию.

Внеаудиторная самостоятельная работа выполняется студентом по заданию преподавателя, но без его непосредственного участия.

Задачи самостоятельной работы студента:

- развитие навыков самостоятельной учебной работы;
- освоение содержания дисциплины;
- углубление содержания и осознание основных понятий дисциплины;
- использование материала, собранного и полученного в ходе самостоятельных занятий для эффективной подготовки к зачету.

Виды внеаудиторной самостоятельной работы:

- самостоятельное изучение отдельных тем дисциплины;
- подготовка к лекционным занятиям;
- подготовка к семинарам и практическим занятиям;
- оформление отчетов по выполненным лабораторным работам и подготовка к их защите;
- выполнение расчетно-графической работы.

Для выполнения любого вида самостоятельной работы необходимо пройти следующие этапы:

- определение цели самостоятельной работы;
- конкретизация познавательной задачи;

- самооценка готовности к самостоятельной работе;
- выбор адекватного способа действия, ведущего к решению задачи;
- планирование работы (самостоятельной или с помощью преподавателя) над заданием;
- осуществление в процессе выполнения самостоятельной работы самоконтроля (промежуточного и конечного) результатов работы и корректировка выполнения работы;
- рефлексия;
- презентация работы.

10. Методические рекомендации для преподавателя

На первом занятии по дисциплине необходимо ознакомить студентов с порядком ее изучения (темами курса, формами занятий, текущего и промежуточного контроля), раскрыть место и роль дисциплины в системе наук, ее практическое значение, довести до студентов требования к форме отчетности и применения видов контроля. Выдаются задания для подготовки к семинарским занятиям.

При подготовке **к семинарскому занятию** по перечню объявленных тем преподавателю необходимо уточнить план их проведения, продумать формулировки и содержание учебных вопросов, выносимых на обсуждение, ознакомиться с перечнем вопросов по теме семинара.

В ходе семинара во вступительном слове раскрыть практическую значимость темы семинарского занятия, определить порядок его проведения, время на обсуждение каждого учебного вопроса. Применяя фронтальный опрос дать возможность выступить всем студентам, присутствующим на занятии.

Целесообразно в ходе защиты лабораторных работ задавать выступающим и аудитории дополнительные и уточняющие вопросы с целью выяснения их позиций по существу обсуждаемых проблем.

Следует предоставить возможность выступления с места в виде кратких сообщений по подготовленному заранее вопросу.

В заключительной части семинарского занятия следует подвести его итоги: дать оценку выступлений каждого студента и учебной группы в целом. Раскрыть положительные стороны и недостатки проведенного семинарского занятия. Ответить на вопросы студентов. Выдать задания для самостоятельной работы по подготовке к следующему занятию.

Приложение 1.

Структура и содержание дисциплины «Программирование и основы алгоритмизации» по направлению подготовки **09.03.01 «Информатика и вычислительная техника»** и профилю подготовки **«Киберфизические системы»**

n/n	Раздел	Семестр	Неделя семестра	Виды учебной работы, включая самостоятельную работу студентов, и трудоемкость в часах					Виды самостоятельной работы студентов					Формы аттестации	
				Л	П/С	Лаб	СРС	КСР	ПЛР*	СИ**	Т	Реферат	К/р	Э	З
	Тема 1. Введение в процедурный язык программирования С. Переменные и арифметика. Условные операторы. Операторы цикла. Символические константы. Ввод и вывод.	4	1	1			1								
	Тема 2. Типы, операции и выражения. Имена переменных. Типы и размеры данных. Константы. Арифметические операции. Операции отношения и логические операции. Преобразование типов. Побитовые логические операции.	4	1	1			1								
	Тема 3. Поток управления. Операторы и блоки. Условныйоператор IF - ELSE. Циклы - WHILE, FOR, DO - WHILE. Оператор BREAK. Оператор CONTINUE.	4		1			1								
	Лабораторная работа № 1. Изучение синтаксиса процедурного языка программирования С.	4				8	8								
	Лабораторная работа № 2. Изучение компилятора GCC.	4				8	8								
	Лабораторная работа № 3. Изучение циклов и условных операторов процедурного языка программирования С.	4				12	12								
	Тема 4. Функции и структура программ. Основные сведения. Правила, определяющие область действия.	4		1			1								

	Статические переменные. Инициализация. Препроцессор языка С. Включение файлов. Макроподстановка.														
	Лабораторная работа № 4. Изучение структур процедурного языка программирования С.	4				12	12								
	Тема 5. Указатели и массивы. Указатели и адреса. Указатели и аргументы функций. Указатели и массивы. Адресная арифметика. Указатели символов и функции. Многомерные массивы. Массивы указателей. Указатели указателей. Инициализация массивов указателей. Указатели и многомерные массивы. Командная строка аргументов. Указатели на функции.	4		1			1								
	Тема 6. Структуры. Основные сведения. Структуры и функции. Массивы сруктур. Указатели на структуры. Структуры, ссылающиеся на себя.	4		1			1								
	Тема 7. Ввод и вывод. Обращение к стандартной библиотеке. Стандартный ввод и вывод - функции GETCHAR и PUTCHAR. Форматный вывод - функция PRINTF. Форматный ввод - функция SCANF. Форматное преобразование в памяти. Доступ к файлам. Обработка ошибок - STDERR и EXIT. Ввод и вывод строк.	4		1			1								
	Лабораторная работа № 5. Изучение методов ввода и вывода процедурного языка программирования С.	4				8	8								
	Тема 8. Введение математического моделирования. Базовые опредения. Основные этапы математического моделирования. Прямые и обратные задачи математического моделирования.	4		1			1								

	Тема 9. Численные методы. Методы решения математических задач при помощи моделирования случайных величин. Методы приближённого вычисления определённого интеграла. Решение нелинейных уравнений.	4		1			1								
	Лабораторная работа № 6. Изучение численных методов.	4				12	12								
	Тема 10. Основы алгоритмизации. Понятие алгоритма. Основные характеристики алгоритмов. Способы описания алгоритмов. Структуры алгоритмов.	4		1			1								
	Тема 11. Структуры данных. Линейные однонаправленные (двухнаправленные) списки. Древовидные структуры. Графы.	4		1			1								
	Тема 12. Алгоритмы сортировок и поиска. Линейный поиск. Двоичный поиск в упорядоченных данных. Сортировка выбором. Сортировка вставками. Вставка погружением. Сортировка Шелла.	4		1			1								
	Лабораторная работа № 7. Изучение структур данных.	4				8	8								
	Лабораторная работа № 8. Изучение методов сортировок.	4				8	8								
	Тема 13. Основы проектирования программного обеспечения. Диаграмма вариантов использования. Диаграмма классов. Диаграмма состояний. Диаграмма последовательности. Диаграмма кооперации. Диаграмма компонентов. Диаграмма развертывания.	4		2			2								
	Тема 14. Стадии разработки программного обеспечения. Начальный этап. Внутреннее тестирование. Публичное тестирование. Предварительная версия. Финальная	4		2			2								

	версия. Пост-релиз. Общая доступность. Прекращение поддержки.														
	Тема 15. Методологии разработки программного обеспечения. Прогнозируемые методологии. Адаптивные методологии. SCRUM.	4		2			2								
	Лабораторная работа № 9. Изучение пакетов проектирование программного обеспечения.	4				12	12								
	Итого в 4 семестре			***18***		***90***	***108***							+	
	Итого:			***18***		***90***	***108***							**+**	**+**

*ПЛР – написание отчета и подготовка к защите лабораторной работы
СИ** - самостоятельное изучение

Приложение 2

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(МОСКОВСКИЙ ПОЛИТЕХ)

Направление подготовки:
09.03.01 «Информатика и вычислительная техника»

Профиль подготовки
«Киберфизические системы»

Форма обучения: очная

Кафедра «СМАРТ-технологии»

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

ПО ДИСЦИПЛИНЕ

Программирование и основы алгоритмизации

Состав:
1. Паспорт фонда оценочных средств
2. Описание оценочных средств:
Перечень вопросов для экзамена
Перечень вопросов для зачета
Перечень вопросов для защиты лабораторных работ
Тестовые задания

Москва, 2022 год

1. **Паспорт фонда оценочных средств**

Таблица 1

ПОКАЗАТЕЛЬ УРОВНЯ СФОРМИРОВАННОСТИ КОМПЕТЕНЦИЙ
ПРОГРАММИРОВАНИЕ И ОСНОВЫ АЛГОРИТМИЗАЦИИ

ФГОС ВО **27.03.04 «Управление в технических системах»**

В процессе освоения данной дисциплины студент формирует и демонстрирует следующие **компетенции**:

КОМПЕТЕНЦИИ		Перечень компонентов	Технология формирования компетенций	Форма оценочного средства**	Степени уровней освоения компетенций
ИНДЕКС	ФОРМУЛИРОВКА				
ОПК-8	способность разрабатывать алгоритмы и программы, пригодные для практического применения	***Знать:*** - основы алгоритмизации (свойства алгоритмов, область применения алгоритмов); - методы построения алгоритмов; - структуры данных; - синтаксис и семантику универсального алгоритмического языка программирования высокого уровня; - основные принципы и методологию разработки прикладного программного обеспечения; - типовые способы организации программных данных; - подходы к построению программных алгоритмов. ***Уметь:***	лекция, лабораторные работы, самостоятельная работа	ЗЛР, Т, З	**Базовый уровень:** воспроизводство полученных знаний в ходе текущего контроля; умение решать типовые задачи, принимать профессиональные решения по известным алгоритмам, правилам и методикам **Повышенный уровень:** практическое применение полученных знаний в процессе изучения дисциплины; готовность решать практические задачи повышенной сложности, нетиповые задачи, принимать профессиональные решения в условиях неполной определенности, при недостаточном документальном, нормативном и методическом обеспечении

		- строить блок-схемы алгоритмов; - проводить анализ эффективности алгоритмов; - описать алгоритм, используя ключевые слова языков программирования, но опуская подробности и специфический синтаксис (псевдокод). ***<u>Владеть:</u>*** - навыками составления алгоритмов и их представления в виде блок-схем; - навыками формализации прикладных задач.			
ПК-1	Способность разрабатывать и отлаживать программный код	***<u>Знать:</u>*** - инструменты разработки программного обеспечения; - виды структур данных; - диаграммы проектирования программного обеспечения; - стадии разработки программного обеспечения. ***<u>Уметь:</u>*** - уметь согласованно решать задачи разработки эффективных моделей данных и алгоритмов их обработки при создании прикладного программного обеспечения; - получать программные реализации полученных	лекция, лабораторные работы, самостоятельная работа	ЗЛР, Т, Э	**Базовый уровень:** воспроизводство полученных знаний в ходе текущего контроля; умение решать типовые задачи, принимать профессиональные решения по известным алгоритмам, правилам и методикам **Повышенный уровень:** практическое применение полученных знаний в процессе изучения дисциплины; готовность решать практические задачи повышенной сложности, нетиповые задачи, принимать профессиональные решения в

решений на процедурном языке программирования С.

Владеть:
- навыками использования инструментальных программных средств в процессе разработки программного обеспечения;
- навыками построения проектирования программного обеспечения;
- навыками разработки программного обеспечения.

условиях неполной определенности, при недостаточном документальном, нормативном и методическом обеспечении

2. **Перечень оценочных средств по дисциплине**

Программирование и основы алгоритмизации

№ ОС	Наименование оценочногосредства	Краткая характеристика оценочного средства	Представление оценочного средства в ФОС
1	Тест (Т)	Система стандартизированных заданий, позволяющая автоматизировать процедуру измерения уровня знаний и умений обучающегося.	Фонд тестовых заданий
2	ЗЛР	Средство проверки умений и навыков применять полученные знания для решения практических задач с помощью инструментальных средств.	Задания для защиты лабораторных работ
3	З		
4	Э		

2.1. Перечень вопросов для зачета (3 семестр) (ОПК-8)

1. Блоки и правила видимости переменных.
2. Виды операторов присваивания в языке Си.
3. Глобальные и внешние переменные.
4. Директива #define препроцессора и ее использование. Макроопределения с параметром.
5. Директива #include препроцессора и ее использование.
6. Директивы условной компиляции препроцессора и их использование.
7. Использование функций: заголовок, тело и вызов функции.
8. Логические (булевские) операторы и операторы сравнения.
9. Локальные, глобальные, статические переменные.
10. Массивы. Передача массивов в функции.
11. Массивы: одномерные и двумерные.
12. Модульный подход в программировании. Использование *.h файлов. Раздельная компиляция.
13. Оператор typedef. Приведение типов.
14. Операторы в выражениях языка Си. Приоритет операторов. Оператор sizeof().
15. Операторы инкремента и декремента.
16. Основы синтаксиса языка Си. Ключевые слова. Фундаментальные типы данных. Определение переменных и констант. Выражения, операции, комментарии.
17. Передача параметров в функции. Передача параметров по значению.

18. Процедурный подход программирования. Определение функции. Прототип функции.

2.2. Перечень вопросов для экзамена (4 семестр) ОПК-6

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
(МОСКОВСКИЙ ПОЛИТЕХ)

Факультет информатики и систем управления, Кафедра «СМАРТ-технологии»
Дисциплина «Программирование и основы алгоритмизации»
Образовательная программа 09.03.01 «Информатика и вычислительная техника»
ОП Киберфизические системы
Курс 2, семестр 4

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ №3

1. Введение математического моделирования. Базовые опредения.
2. Решение нелинейных уравнений.

Утверждено на заседании кафедры «29» декабря 2016 г., протокол №5.

Зав. кафедрой _____ /А.В. Кузнецов/

1. Введение математического моделирования. Базовые опредения.
2. Основные этапы математического моделирования. Прямые и обратные задачи математического моделирования.
3. Численные методы. Методы решения математических задач при помощи моделирования случайных величин.
4. Методы приближённого вычисления определённого интеграла.
5. Решение нелинейных уравнений.
6. Основы алгоритмизации. Понятие алгоритма. Основные характеристики алгоритмов. Способы описания алгоритмов. Структуры алгоритмов.
7. Структуры данных. Линейные однонаправленные (двухнаправленные) списки. Древовидные структуры. Графы.
8. Алгоритмы сортировок и поиска. Линейный поиск. Двоичный поиск в упорядоченных данных. Сортировка выбором. Сортировка вставками. Вставка погружением. Сортировка Шелла.
9. Основы проектирования программного обеспечения. Диаграмма вариантов использования. Диаграмма классов. Диаграмма состояний.
10. Диаграмма последовательности. Диаграмма кооперации. Диаграмма компонентов. Диаграмма развертывания.
11. Стадии разработки программного обеспечения. Начальный этап. Внутреннее тестирование. Публичное тестирование. Предварительная версия. Финальная версия. Пост-релиз. Общая доступность. Прекращение поддержки.
12. Методологии разработки программного обеспечения. Прогнозируемые методологии. Адаптивные методологии. SCRUM.

2.3. Фонд тестовых заданий

3 семестр (ОПК-9)

1. Что такое функция?

a) Некоторая часть программы, содержащая описание переменных и констант основной программы
b) Некоторая часть программы, имеющая собственное имя и которая может вызываться из основной программы
c) Некоторая часть программы, содержащая вредоносный код, и блокирует определенные действия системы
d) Некоторая часть программы, в которой происходит начальная инициализация всех полей структур, массивов, переменных.

Правильный ответ: b

2. Что такое массив?

a) Именованный набор переменных имеющих различные типы данных, и располагающихся в одной памяти
b) Именованный набор переменных и функций, которые располагаются в одной области памяти
c) Именованный набор переменных имеющий один тип данных, и располагающихся в одной области памяти
d) Именованный набор переменных имеющих символьный тип данных, и располагающихся в одной области памяти

Правильный ответ: d

3. Как написать следующее выражение на языке С «Переменной a присвоено значение b»?

a) a==b
b) a=b
c) b=a
d) a:=b

Правильный ответ: b

4. Как написать следующее выражение «Второму элементу массива Myarray присвоено значение пяти »?

a) int [1] Myarray=«пять»
b) int Myarray [1] = 5
c) int Myarray [2] = «пять»
d) int Myarray [2] = 5

Правильный ответ: d

5. Как написать следующее выражение «Если переменная index больше size то мы инкрементируем переменную count »?

a) if (index>size) { count++; }

b) if (index<size) { count--; }
c) if (index>=size) { ++count; }
d) if (index<size) { --count; }

Правильный ответ: a

6. Какой диапазон значений имеет тип int для 32-разрядных вычислительных систем:

a) от 0 до 255
b) от -32768 до 32767
c) от 0 до 65535
d) от 0 до 4 294 967 295

Правильный ответ: b

7. Какой размер в байтах имеет переменная вещественного типа float

a) 2
b) 4
c) 8
d) 10

Правильный ответ: a

8. Дан массив int L[3][3] = { { 2, 3, 4 }, { 3, 4, 8 }, { 1, 0, 9 } };. Чему будет равно значение элемента этого массива L[1][2]

a) 2
b) 3
c) 4
d) 8

Правильный ответ: d

9. Объявление char *buf; соответствует

a) созданию символьной переменной buf
b) созданию строковой переменной buf
c) созданию указателя buf на символьное значение
d) созданию указателя buf на строку

Правильный ответ: c

10. Что называется прототипом функции?

a) описание функции, включая ее имя, тип возвращаемого значения, имена и типы параметров
b) описание функции, включая ее имя, тип возвращаемого значения, типы параметров
c) имя функции и тип возвращаемого значения
d) описание функции, включая ее имя, тип возвращаемого значения, имена и типы параметров, тело функции

Правильный ответ: c

11. Как обозначается в языке С следующий режим работы с потоком - создание нового файла для записи и чтения?

a) a+
b) wb
c) w+
d) w+b

Правильный ответ: b

12. Какая функция, описанная в заголовочном файле читает строку символов из файла?

a) gets()
b) fputs()
c) fgets()
d) fscanf()

Правильный ответ: a

13. Какой размер массива M будет после выполнения кода:
char M[ ]="\nGoodlive" ?

a) 10
b) 8
c) 9
d) Не определен

Правильный ответ: b

14. В каких случаях необходимо использовать оператор return в теле функции?

a) Всегда
b) если необходимо, чтобы функция вернула значение
c) если необходимо обеспечить выход из функции в произвольном месте
d) если указан тип возвращаемого значения, в том числе и void

Правильный ответ: a

15. При открытии файла выполняется следующее действие:

a) физический файл связывается с логическим (файловой переменной)
b) устанавливается тип файла (текстовый или бинарный)
c) устанавливается вид (режим) использования файла
d) функцией открытия файла возвращается результат (ошибка)

Правильный ответ: a

16. Какое ключевое слово языка С используется для описания структурированного типа данных, все элементы которого в памяти начинаются с одного байта?

a) struct
b) union
c) enum

d) template

Правильный ответ: a

17. Каким способом можно задать многострочный комментарий в языке C++

a) /*комментарии к программе*/
b) //комментарии к программе//
c) //комментарии к программе
d) {комментарии к программе}

Правильный ответ: a

18. Логическое выражение может возвращать результат типа

a) integer
b) boolean
c) char
d) logical

Правильный ответ: b

19. Выберите правильный вариант записи на языке C формулы 0<x<10

a) x>0, x<=10
b) 0<x<=10
c) x>0 AND x<=10
d) (x>0) AND (x<10)

Правильный ответ: d

20. Укажите правильный вариант записи условного оператора в языке C

a) IF x>0 Do y:=sqrt (x)
b) IF y:=sqrt (x) then x>0
c) IF x>0 then y:=sqrt (x)
d) IF (x>0) { y:=sqrt (x)}

Правильный ответ: d

21. Выберите правильный вариант записи на языке C следующего условия: « x принадлежит диапазону [0;10)»

a) x>=0; x<10
b) 0<=x<10
c) (x>0 AND (x<=10)
d) (x>=0) AND (x<10)

Правильный ответ: d

22. Укажите группу, содержащую последовательность правильно записанных на языке C знаков операций отношений

a) ~ >, <, =, ?

b) =, <>, ><, >
c) =, >=, <=, !=
d) ~ =>,=<, =, <

Правильный ответ: d

23. Тело какого цикла всегда будет выполнено хотя бы один раз, независимо от истинности условия:

a) While
b) Do While
c) For
d) Нет такого цикла в языке С

Правильный ответ: b

24. В результате выполнения кода
int i=2; switch (i) { case 1: i += 2;case 2: i *= 3; case 6: i /= 2;
default: ; }

a) переменная i примет значение 6
b) переменная i примет значение 3
c) переменная i примет значение 2
d) тело оператора switch не поменяет значение переменной i

Правильный ответ: a

25. Укажите директиву препроцессора, которую необходимо подключить для организации форматированного ввода-вывода данных:
a) #include <stdlib.h>
b) #include <stdio.h>
c) #include <math.h>
d) #include <time.h>

Правильный ответ: b

4 семестр (ОПК-9)

Вариант 1.

1) Статическими данными называются такие данные, которые…

а) не меняют свои значения в течение всего времени существования.

б) не меняют свои размеры в течение всего времени существования.

в) не меняют своего имени в течение всего времени существования.

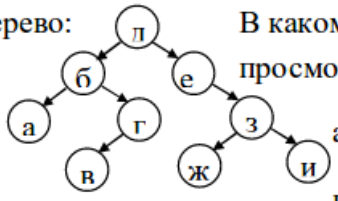
г) вычисляют статистические характеристики последовательностей чисел.

2) Стеком называется структура…

а) "первым пришел, последним вышел" б) "последним пришел, последним вышел"

в) "последним пришел, первым вышел" г) "первым пришел, первым вышел"

3) Дано дерево:



В каком порядке просматриваются узлы при прямом просмотре ?

а) а, б, в, г, д, е, ж, з, и б) д, б, а, г, в, е, з, ж, и

в) г, е, ж, з, б, д, и, в, а г) а, в, г, б, ж, и, з, е, д

4) Какова основная цель сортировки объектов?

а) Облегчить выполнение последующих математических операций в отсортированном множестве.

б) Облегчить выполнение последующих статистических операций в отсортированном множестве.

в) Облегчить выполнение последующих логических операций в отсортированном множестве.

г) Облегчить последующий поиск элементов в отсортированном множестве.

5) Какой метод сортировки описывает следующий алгоритм:

for i := 1 to n-1 do begin

присвоить k индекс наименьшего элемента из a[i] … a[n];

поменять местами a[i] и a[k] end;

а) Сортировку простыми включениями. б) Сортировку простым обменом.

в) Быструю сортировку. г) Сортировку простым выбором.

6) Сортировку файлов называют внешней сортировкой потому, что сортируются последовательности элементов…

а) на таких устройствах, как диски, ленты, и доступ к элементам является последовательным.

б) на таких устройствах, как диски, ленты, и доступ к элементам является произвольным.

в) в оперативной памяти и доступ к элементам является произвольным.

г) в оперативной памяти и доступ к элементам является последовательным.

вод рисунка; г) редактирование программы.

Вариант 2.

1) Динамическими данными называют такие данные, которые…

а) могут менять свое имя в течение всего времени существования.

б) могут менять свои значения в течение всего времени существования.

в) могут менять свои размеры в течение всего времени существования.

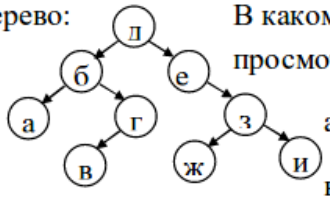
г) характеризуют динамику поведения некоторых процессов.

2) Очередью называется структура…

а) "первым пришел, последним вышел" б) "последним пришел, последним вышел"

в) "последним пришел, первым вышел" г) "первым пришел, первым вышел"

3) Дано дерево:



В каком порядке просматриваются узлы при прямом просмотре ?

а) а, б, в, г, д, е, ж, з, и б) д, б, а, г, в, е, з, ж, и

в) г, е, ж, з, б, д, и, в, а г) а, в, г, б, ж, и, з, е, д

4) На какие два класса разбиты методы сортировки в зависимости от структуры обрабатываемых данных?

а) Сортировка массивов и сортировка списочных структур.

б) Сортировка линейных списков и сортировка деревьев.

в) Сортировка массивов и сортировка файлов (последовательностей).

г) Сортировка файлов с последовательным и с прямым доступом.

5) Какой метод сортировки описывает следующий алгоритм:

for i := 2 to n do begin x := a[i];

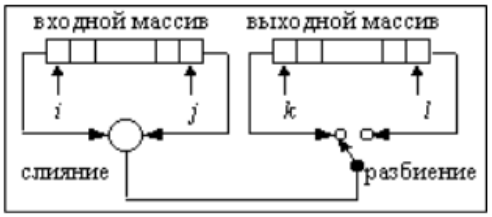
вставить x на подходящее место среди a[1] … a[i] end;

а) Сортировку простыми включениями. б) Сортировку простым обменом.

в) Сортировку включениями с убывающими приращениями.

г) Сортировку простым выбором.

6) Какую сортировку описывает схема:



а) Естественное слияние.

б) Сбалансированное многопутевое слияние.

в) Простое слияние.

г) Многофазную сортировку.

Вариант 3.

1) Что означает служебное слово **NIL** ?

а) Ноль.

б) Процедуру, которая создает новую динамическую переменную и устанавливает на нее ссылку.

в) Процедуру, которая уничтожает динамическую переменную.

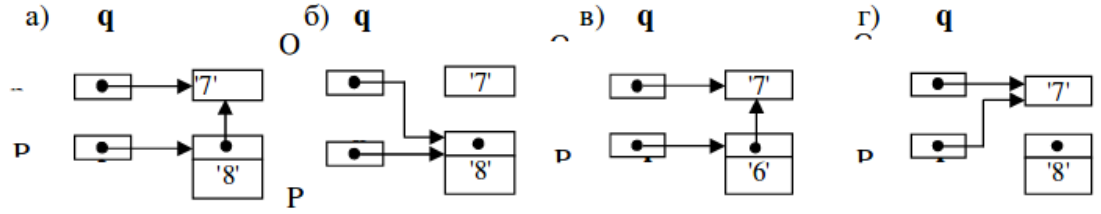
г) Константу ссылочного типа, которая означает пустую ссылку, т.е. ссылку, которая "никуда не указывает".

2) Type A = ^char; B = Record f1 : char; f2 : A End;

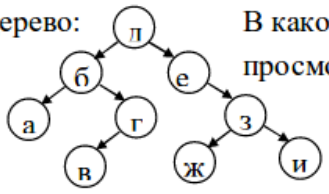
Var p : ^B; q : A;

Какова будет структура значений переменных **p** и **q** после выполнения следующих операторов:

new(q); q^ := '7'; new(p); p^.f1 := succ(q^); p^.f2 := q



3) Дано дерево:



В каком порядке просматриваются узлы при концевом просмотре ?

а) а, б, в, г, д, е, ж, з, и

б) д, б, а, г, в, е, з, ж, и

в) г, е, ж, з, б, д, и, в, а

г) а, в, г, б, ж, и, з, е, д

4) Метод сортировки называется устойчивым, если…

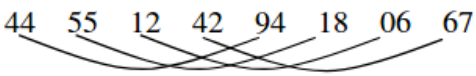
а) его эффективность не зависит от первоначального расположения элементов.

б) его эффективность зависит от первоначального расположения элементов.

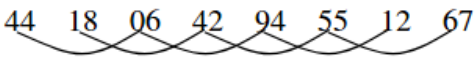
в) его эффективность выше, если первоначальная последовательность элементов почти рассортирована.

г) в процессе сортировки порядок элементов с равными ключами не изменяется.

5) Какой метод сортировки демонстрируется следующим примером:



а) Пирамидальная сортировка.



б) Сортировка Шелла.

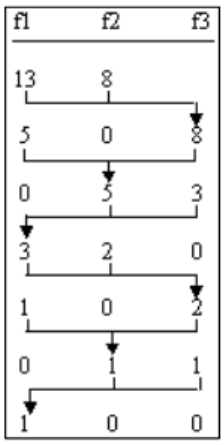
06 18 12 42 44 55 94 67

в) Быстрая сортировка.

06 12 18 42 44 55 67 97

г) Сортировка простым обменом.

6) Какую сортировку описывает схема:



а) Естественное слияние.

б) Сбалансированное многопутевое слияние.

в) Простое слияние.

г) Многофазную сортировку.

Вариант 4.

1) Что означает служебное слово **NEW** ?

а) Процедуру, которая создает новую динамическую переменную и устанавливает на нее ссылку.

б) Процедуру, которая уничтожает динамическую переменную.

в) Константу ссылочного типа, которая означает пустую ссылку, т.е. ссылку, которая "никуда не указывает".

г) Функцию, возвращающую адрес заданного объекта.

2) Имеется описание: **Type связь = ^объект; объект = record следующий: связь;**

данные: тип_данных end;

Что делает следующая процедура?

```
Procedure X(Var первый, головной, замыкающий: связь);
begin первый:=головной;
    if головной<>nil then begin
        головной:=головной^.следующий;
    if головной = nil then замыкающий := nil end
end;
```

а) Удаляет из списка последнюю компоненту и настривает на нее ссылку **первый**.

б) Помещает в конец очереди новую компоненту. Ссылка на новую компоненту содержится в переменной **первый**.

в) Исключает из очереди первую компоненту и настраивает на нее ссылку **первый**.

г) Исключает из очереди последнюю компоненту и настраивает на нее ссылку **первый**.

3) Перечислите порядок просмотра двоичного дерева при прямом его просмотре.

а) Правое поддерево, левое поддерево, узел.

б) Левое поддерево, правое поддерево, узел.

в) Узел, левое поддерево, правое поддерево.

г) Узел, правое поддерево, левое поддерево.

4) Метод сортировки обладает неестественным поведением, если…

а) его эффективность не зависит от первоначального расположения элементов.

б) его эффективность выше, если исходная последовательность элементов более или менее рассортирована в обратном порядке.

в) его эффективность выше, если исходная последовательность элементов почти рассортирована в прямом порядке.

г) если в процессе сортировки порядок элементов с равными ключами не изменяется.

5) Какой усовершенствованный метод основан на принципе обмена?

а) Сортировка Шелла. б) Сортировка вставками.

в) Сортировка с разделениями. г) Пирамидальная сортировка.

6) Многофазная сортировка более эффективна, чем сбалансированная сортировка, поскольку, если даны N лент, то она всегда имеет дело …

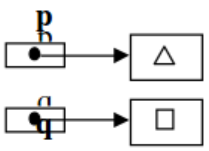
а) с N-путевым слиянием, вместо N/2-путевого слияния.

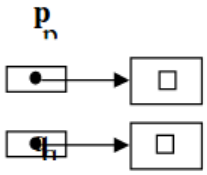
б) с N-путевым слиянием, вместо N/2-путевого слияния.

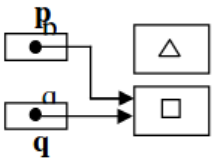
в) с (N-1)-путевым слиянием, вместо N-путевого слияния.

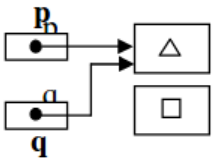
г) с (N+1)-путевым слиянием, вместо N/2-путевого слияния.

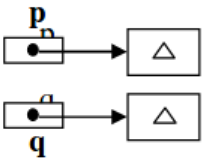
Вариант 5.

1) Пусть имеются следующие ссылки:  Что произойдет после выполнения присваивания: **p :=q** ?

а) 

б) 

в) 

г) 

2) Имеется описание: **Type связь = ^объект; объект = record всс, нсс : связь;**

данные: тип_данных end;

Что делает следующая процедура?

Procedure X(ваня, вася : связь);

begin вася^.всс := ваня^.всс; вася^.нсс := ваня;

ваня^.всс^.нсс := вася; ваня^.всс := вася

end;

а) В линейном списке вставляет объект **вася** перед объектом **ваня**.

б) В линейном списке вставляет объект **вася** после объекта **ваня**.

в) В двусвязном кольце вставляет объект **вася** перед объектом **ваня**.

г) В двусвязном кольце вставляет объект **вася** перед объектом **ваня**.

3) Перечислите порядок просмотра двоичного дерева при обратном его просмотре.

а) Правое поддерево, левое поддерево, узел. б) Левое поддерево, узел, правое поддерево.

в) Узел, левое поддерево, правое поддерево. г) Узел, правое поддерево, левое поддерево.

4) Какие параметры используются в качестве меры эффективности метода сортировки?

а) Число необходимых сравнений ключей и число перестановок элементов.

б) Число необходимых сравнений ключей и число присваиваний значений элементам.

в) Число необходимых проходов по последовательности элементов и число перестановок элементов.

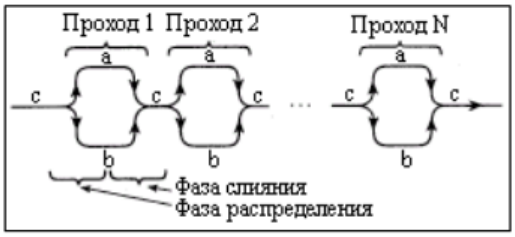
г) Число необходимых проходов по последовательности элементов и число присваиваний значений элементам.

5) Какой усовершенствованный метод основан на принципе простого выбора?

а) Сортировка Шелла. б) Сортировка вставками.

в) Сортировка с разделениями. г) Пирамидальная сортировка.

6) Какую сортировку описывает схема:



а) Естественное слияние.

б) Сбалансированное многопутевое слияние.

в) Простое слияние.

г) Многофазную сортировку.

2.4. Примеры защиты лабораторных работ

3 семестр (ОПК-9)

1. Реализовать и защитить программу: Вычислить факториал числа n.
2. Реализовать и защитить программу: Вычислить степень числа рекурсивным способом.
3. Реализовать и защитить программу: Программа "Скобки" должна определять правильность введённой скобочной структуры. Правильная скобочная структура, это то, что можно получить из арифметического выражения со скобками, выкинув все знаки операций и цифры.
4. Реализовать и защитить программу: Построить строку простых чисел, на которые можно делить трехзначительное целое число. Выяснить сумму строки.

5. Реализовать и защитить программу: Выяснить наиболее часто встречаемую начальную букву слов. Построить график расположения начальных букв слов. Вывести таблицу начальных букв слов в файл.
6. Реализовать и защитить программу: В двухмерном массиве сделать перестановку элементов первой четверти квадратной матрицы с элементами четвертой четверти той же матрицы.
7. Реализовать и защитить программу: Дано 20 вещественных чисел. Найти разницу между минимальным и максимальным из них.

4 семестр (ОПК-9)

1. Реализовать структуру данных стек с функциями: положить элемент на вершину стека, вытащить элемент с вершины стека, отсортировать стек.
2. Реализовать структуру данных очередь: положить элемент в начало очереди, вытащить элемент с конца очереди.
3. Реализовать структуру данных односвязный список: добавить элемент в начало списка, положить элемент в конец списка, положить элемент по определенному адресу.
4. Реализовать структура данных двоичное дерево с функциями: инициализация дерева, добавление листа в дерево.