

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ



УТВЕРЖДАЮ
Декан ФИиВТ

УТВЕРЖДАЮ /А.А. Кречетов/
(Ф.И.О. декана (директора института))

25.01.2023 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

Б.1.1.18 Математическая логика и теория алгоритмов

(код и наименование дисциплины по учебному плану)

Направление подготовки
(специальность)

09.03.04 Программная инженерия

Квалификация выпускника

Бакалавр

(бакалавр/магистр/специалист)

Направленность

Разработка программных систем

Курс 2
Семестр 4

Распределение учебного времени

Трудоемкость по учебному плану	180 / 5	часов/зачетных единиц
Лекции	36	часов
Лабораторные работы	36	часов
Практические занятия	-	часов
Иная контактная работа	-	часов
Всего контактной работы (без учета экз.)	72	часов
Контактная работа по экзамену	6	часов
Курсовой проект (работа)	-	семестр
Самостоятельная работа обучающихся (без учета экз.)	72	часов
Самостоятельная работа по подготовке к экзамену	30	часов
Экзамен	4	семестр
Зачет	-	семестр
БРК, ДЗ	-	семестр

(год)

Программа составлена в соответствии с требованиями ФГОС ВО направления подготовки (специальности) 09.03.04 Программная инженерия

Программу составили:

доцент с ученой степенью кандидата наук	ИиСП	СОГЛАСОВАНО	В.И. Галочкин
(должность)	(кафедра)		(И.О. Фамилия)

РАССМОТРЕНА и ОДОБРЕНА на заседании кафедры, за которой закреплена дисциплина
Кафедра информатики и системного программирования

		(наименование кафедры)	
25.01.2023	протокол №	1	
(дата)			
Заведующий кафедрой	СОГЛАСОВАНО	А.В. Бородин	
		(И.О. Фамилия)	

Рабочая программа СОГЛАСОВАНА с факультетом (институтом), выпускающей(ими)
кафедрой(ами).
СООТВЕТСТВУЕТ действующей ОП.

Заведующий кафедрой	СОГЛАСОВАНО	А.В. Бородин
		(И.О. Фамилия)

Председатель методической комиссии факультета (института), в который входит
выпускающая кафедра

СОГЛАСОВАНО	А.А. Кречетов
	(И.О. Фамилия)

Эксперт(ы): Егошин А.Б., Ген. директор ООО "Цитрус"

Рабочая программа проверена и зарегистрирована в УМЦ 14.02.2023 г.
Специалист учебно-методического центра СОГЛАСОВАНО /И.Р. Валиева/

Раздел 1. ЦЕЛЬ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Целью освоения дисциплины является достижение планируемых результатов обучения, соответствующих установленным в ОПОП индикаторам достижения компетенций:

Код и наименование компетенции	Код и наименование индикатора достижения компетенции	Результаты обучения
1. УК-1 Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач	УК-1.1 Выполняет поиск необходимой для решения поставленной задачи информации, её критический анализ, обобщение и представление на основе знаний естественно-научных дисциплин и современных информационных технологий	знания: Принципов сбора, классификации, обобщения и формализации информации, характеристики и показатели алгоритмов обработки информации умения: навыки:
	УК-1.2 Систематизирует обнаруженную информацию, полученную из разных источников, в соответствии с требованиями и условиями задачи	знания: умения: Систематизировать разнородные явления в зависимости от видов профессиональной деятельности и разрабатывать алгоритмы решения задач навыки:
	УК-1.3 Выбирает оптимальный вариант решения задачи, аргументируя свой выбор	знания: умения: навыки: Опыт работы с библиографией и технической документацией, научного поиска необходимой информации, создания научно-технических текстов
2. ОПК-1 Способен применять естественнонаучные и общеинженерные знания, методы математического анализа и моделирования, теоретического и экспериментального	ОПК-1.1 Знает основы математики, физики, вычислительной техники и программирования.	знания: Знает основы математической логики и методов оценки вычислительной сложности алгоритмов умения: навыки:
	ОПК-1.2 Умеет решать стандартные профессиональные задачи с применением естественнонаучных и обще-инженерных знаний, методов математического анализа и моделирования	знания: умения: Умеет применять комбинаторные алгоритмы для решения профессиональных задач на основе общеинженерных знаний навыки:

исследования в профессиональной деятельности	ОПК-1.3 Имеет навыки теоретического и экспериментального исследования объектов профессиональной деятельности.	знания: умения: навыки: Имеет навыки оценки сложности алгоритмов, логического и нагрузочного тестирования программных приложений
--	---	---

Раздел 2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОПОП

Дисциплина относится к обязательной части ОПОП.

Дисциплина является обязательной

Для продолжения формирования заявленных компетенций необходимы знания предшествующих дисциплин: Математика (УК-1), Математика (ОПК-1), Дискретная математика (ОПК-1), Алгоритмы и структуры данных (ОПК-1)

Изучаемая дисциплина является основой для продолжения формирования указанных компетенций в следующих дисциплинах: Управление программными проектами (УК-1); государственной итоговой аттестации в форме: Подготовка к процедуре защиты и защита выпускной квалификационной работы (ОПК-1)

Раздел 3. ОПИСАНИЕ ОБРАЗОВАТЕЛЬНЫХ ТЕХНОЛОГИЙ

Для формирования заявленных компетенций используются методологические технологии, реализующие деятельностный, личностно-ориентированный, практико-ориентированный подходы.

Основными стратегическими технологиями являются: дискуссионные, исследовательские, лекционные занятия, практика, практические и лабораторные занятия, процедуры самообучения

На достижение конкретных целей обучения направлены применяемые тактические технологии: классическая лекция

Раздел 4. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

4 семестр

Виды и темы занятий	Количество часов	Формируемые компетенции
Алгоритмы перебора вариантов	16	УК-1
Лекция. Алгоритмы поиска с возвратом и в ширину	2	
Лекция. Рекуррентные соотношения. Динамическое программирование	2	
Лабораторная работа. Алгоритмы перебора вариантов	4	
Задания для самостоятельной работы, в том числе выполнение 1. Проработка лекций. 2. Подготовка к лабораторным работам. 3. Повторение способов организации статических и динамических структур данных в Паскале и C++. 4. Изучение базовых задач поиска с возвратом и в ширину. 5. Использование рекуррентных соотношений в динамическом программировании.	8	
Целочисленные алгоритмы, перестановки, подмножества	24	УК-1

Лекция. Целочисленная арифметика	2	
Лекция. Перестановки и подмножества	2	
Лекция. Метод сканирующей прямой	2	
Лабораторная работа. Целочисленные алгоритмы, перестановки, подмножества	6	
Задания для самостоятельной работы, в том числе выполнение 1. Проработка лекций. 2. Подготовка к лабораторным работам. 3. Изучение способов реализации алгоритмов в C++. 4. Отработка алгоритмов вручную на примерах малой размерности.	12	
Графы	24	УК-1
Лекция. Алгоритмы нахождения остовных деревьев	2	
Лекция. Алгоритм Беллмана-Форда. Связность графов	2	
Лекция. Циклы и ацикличность. Неявные графы	2	
Лабораторная работа. Графы	6	
Задания для самостоятельной работы, в том числе выполнение 1. Проработка лекций. 2. Подготовка к лабораторным работам. 3. Освоение алгоритмов Прима, Краскала, Беллмана-Форда на простых примерах. 4. Применение и реализация поиска в глубину для анализа связности графов.	12	
Вычислительная геометрия	16	УК-1
Лекция. Основные формулы вычислительной геометрии на плоскости	2	
Лекция. Алгоритмы Джарвиса и Грэхема поиска выпуклой оболочки	2	
Лабораторная работа. Вычислительная геометрия	4	
Задания для самостоятельной работы, в том числе выполнение 1. Проработка лекций. 2. Подготовка к лабораторным работам. 3. Повторение основных формул вычислительной геометрии на плоскости. 4. Изучение алгоритмов Джарвиса и Грэхема на тестовых примерах.	8	
Строки	16	УК-1
Лекция. Алгоритмы поиска подстроки в строке	2	
Лекция. Поиск по многим образцам. Алгоритм Ахо-Корасика	2	
Лабораторная работа. Алгоритмы поиска подстроки	4	
Задания для самостоятельной работы, в том числе выполнение 1. Проработка лекций. 2. Подготовка к лабораторным работам. 3. Сравнение характеристик алгоритмов поиска подстроки. 4. Изучение алгоритмов на тестовых примерах. 5. Освоение контейнерных классов из библиотеки STL в языке C++ для работы со строками. 5. Выполнение тестов для самоконтроля	8	
Игры двух лиц	16	УК-1

Лекция. Программирование игр с известной стратегией	2	УК-1
Лекция. Программирование игр с неизвестной стратегией	2	
Лабораторная работа. Игры двух лиц	4	
Задания для самостоятельной работы, в том числе выполнение 1. Проработка лекций. 2. Подготовка к лабораторным работам. 3. Изучение подходов к простым играм на тестовых примерах. 4. Освоение процедур отсечения бесперспективных вариантов.	8	
Вычислительная сложность алгоритмов	32	
Лекция. Асимптотическая оценка сложности алгоритмов	2	
Лекция. Структуры данных для эффективной реализации алгоритмов	2	
Лекция. Методы оценки сложности алгоритмов	4	
Лабораторная работа. Вычислительная сложность алгоритмов	8	
Задания для самостоятельной работы, в том числе выполнение 1. Проработка лекций. 2. Подготовка к лабораторным работам. 3. Оценка сложности простых и усовершенствованных методов сортировки. 4. Изучение контейнерных классов C++ для повышения эффективности программ. 5. Способы разработки программ-генераторов для получения нагрузочных тестов. 6. Выполнение тестов для самоконтроля.	16	
Иная контактная работа:	0	
Подготовка к экзамену	30	
Проведение экзамена	6	

Раздел 5. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

1. Советы по планированию и организации времени, необходимого на изучение дисциплины

Не допускать значительных перерывов в занятиях. Как и при изучении иностранных языков, регулярность занятий способствует более быстрому и глубокому изучению материала. Наиболее опасно запускать предмет, а затем штурмовать его перед очередным контрольным мероприятием.

Готовиться к каждому аудиторному занятию, чтобы не вспоминать (с потерями) пройденный материал.

Весь материал, приведенный на аудиторных занятиях, должен быть детально проработан.

Фиксировать все моменты, по которым не достигнуто полное понимание, чтобы задать необходимые вопросы товарищам и преподавателю.

В каждом алгоритме полезно осознать необходимость всех условий и подбирать примеры, когда при нарушении условий алгоритм неработоспособен.

Прежде чем разрабатывать собственную программу полезно подробно рассмотреть

примеры, приводимые на занятиях, а также имеющиеся в библиотеке учебных программ. Имеет смысл изменять варианты представления данных и анализировать, что при этом изменится в коде программы.

Невозможно научиться программировать, списывая у товарищей и даже разбирая приведенные примеры. Гораздо полезнее самому решать задачу, не добившись успеха, чем использовать что-то готовое. Только решение достаточно большого объема задач способно выработать необходимые практические навыки.

Проявлять здоровую бдительность при изучении конспектов лекций, книг, методических материалов. Конспекты могут содержать ошибки, преподаватель также может ошибаться, в книгах нередко встречаются неточности и опечатки.

Безусловно полезны коллективные занятия, но они не должны носить характер перелицовки программ более сильного или трудолюбивого товарища. Известно, что объясняющий и сам, как правило, добивается более глубокого понимания темы.

Просматривать материал вперед, что значительно повышает уровень восприятия.

Изучать материал не только по конспектам лекций, но и по приведенным источникам, анализируя, чем отличаются разные подходы, термины и понятия.

Рекомендуемое время на подготовку к лекционным занятиям – 30 мин., лабораторным работам – 1 час.

2. Описание последовательности действий студента или "сценарий изучения дисциплины"

Организация процесса обучения должна подчиняться требованиям рабочей программы и технологической карты в системе РИТМ. Дисциплина состоит из семи модулей-разделов. Дадим общие рекомендации по их изучению.

Алгоритмы перебора вариантов. Прежде всего следует детально разобраться с методами в глубину (DFS) и ширину (BFS), которые лежат в основе многих алгоритмов. Нужно понять, что это общие подходы в задачах перебора вариантов, а не только те методы обхода графов, которые изучались ранее. Метод динамического программирования позволяет избежать полного перебора при решении ряда общих задач, среди которых в первую очередь стоит отметить задачи о камнях и рюкзаке. Распространенной практикой является неоправданное использование рекурсии, что может привести к абсолютно неприемлемому объему вычислений.

Целочисленные алгоритмы, перестановки, подмножества. Необходимо объяснить важность целочисленных алгоритмов в комбинаторных задачах дискретной математики. Методы генерации и случайного выбора перестановок распространены в задачах криптографии и дискретной оптимизации. В лабораторных работах некоторые студенты используют языки Python и Java для того, чтобы избежать программирование функций длинной арифметики. С одной стороны, это показывает эрудицию студентов, однако не дает полного представления о методах работы с библиотеками длинной арифметики, поэтому имеет смысл запретить использовать соответствующие среды программирования.

Графы. Недостаток лекционного времени не позволяет охватить все многообразие алгоритмов на графах. Действительно, в ряде университетов для математиков читается полугодовой курс по теории графов, даже не связанный с программированием прикладных задач. Необходимо объяснить, что применение графов не сводится к поиску путей в различных сетях. Помимо задач исследования связности, поиска циклов, выделения каркасных деревьев имеется много задач, в которых появляются неявные графы. Ряд заданий на лабораторные работы сводятся к построению и использованию неявных графов. Во всех алгоритмах необходимо приводить оценку трудоемкости и понять важность этой оценки на конкретных примерах. Важно учитывать специфику каждой задачи, чтобы выбрать оптимальную структуру представления графов при программировании.

Вычислительная геометрия. Вычислительная геометрия – один из разделов информатики. Существует большое количество научных и прикладных задач, связанных с вычислительной геометрией. Нужно обратить внимание на методы работы с вещественными типами данных, которые часто встречаются при решении геометрических задач. Полезно провести сравнительный анализ алгоритмов Джарвиса и Грэхема построения выпуклой оболочки. Использование свойств скалярного и векторного произведений позволяет существенно упростить решение ряда задач.

Строки. Строки являются одной из самых распространенных структур данных. Битовые строки соответствуют числам. Строки цифр образуют телефонные номера, строки символов – слова. Из слов состоят фразы, из фраз – документы и книги. Многие задачи связаны с проблемой нахождения нужных слов в тексте. Современные программы обработки текста имеют возможности поиска и замена фрагментов, инкапсулированные во многие языки программирования высокого уровня. Но если такого рода поиск является ключевой задачей программы, необходимо знать принципы организации функций поиска. Наконец, область применения функции поиска не ограничивается одними лишь текстовыми редакторами. Следует отметить использование алгоритмов поиска при индексации страниц поисковым роботом, где актуальность информации напрямую зависит от скорости нахождения ключевых слов в тексте. Рекомендуется провести многофакторное сравнение алгоритмов и выделить специфику множественного поиска образцов.

Игры двух лиц. В настоящее время программирование игр является мощной индустрией. Но игровые модели имеют значение не только в развлекательных целях. Существует, например, класс так называемых “игр с природой”, когда необходимо принимать решения, позволяющие добиться наибольшей выгоды, избегая риска катастрофических последствий. Пример другого применения игр – противодействие (ответные ходы) попыткам доступа к конфиденциальной информации. На примере простых игр можно ввести понятия дерева игры, выигрышных и проигрышных позиций, принципа минимакса. Процедура альфа-бета отсечения помимо использования в играх служит хорошей иллюстрацией применения поиска в глубину и метода ветвей и границ.

Вычислительная сложность алгоритмов. Интуитивное понятие вычислительной сложности должно использоваться с начала курса. К сожалению, студенты часто не принимают во внимание трудоемкость алгоритмов. Многие простодушно считают, что отладка приложения на небольших по объему тестах при современной скорости компьютеров достаточна для окончательного успеха. Нужно сразу же приучать студентов генерировать нагрузочные тесты для практической проверки эффективности алгоритмов. Обычно даже лучшие студенты этого избегают. Теоретический материал по оценке сложности алгоритмов позволяет обобщить и закрепить интуитивные понятия. Ввиду особой важности этого раздела, в курсе предусматривается отдельный тест, связанный с оценкой сложности алгоритмов.

3. Рекомендации по использованию материалов рабочей программы

Для более быстрого и методически правильного освоения дисциплины необходимо начать ее изучение с внимательного рассмотрения рабочей программы. Рабочая программа позволит оценить трудоемкость освоения дисциплины, укажет на контрольные точки, на длительность изучения дисциплины, наличие контрольных мероприятий.

Следует посмотреть рекомендуемую литературу и взять ее в библиотеке, причем потребуются литература как по освоению теоретического материала, так и по выполнению лабораторных работ.

Следует периодически обращаться к контрольным материалам, размещенным в учебно-методическом комплексе дисциплины.

При подготовке к экзамену посмотреть вариант билета, проработать экзаменационные вопросы и просмотреть рекомендуемую литературу.

4. Рекомендации по работе с литературой

В библиотеке имеется достаточное количество экземпляров учебно-методической литературы. Помимо изданных пособий много материалов представлено в электронном виде и имеется на кафедре и у преподавателя.

При работе с литературой рекомендуется пытаться в первую очередь понять смысл вводимых терминов, их связь, практическую значимость, а не заучивать определения, формулы, элементы структур. Последовательность подачи материала соответствует рекомендациям учебной программы;

Для успешного выполнения лабораторных работ имеет смысл в качестве тренировки выполнять на компьютере базовые программы из библиотеки учебных программ, предоставляемой преподавателем, осмысливая получаемые результаты. Указанные программы допускается использовать в качестве фрагментов выполняемых лабораторных работ.

5. Советы по подготовке к экзамену, критерии экзаменационных (зачетных) оценок

Навыки в разработке алгоритмов и программ вырабатываются и оцениваются на лабораторных занятиях, а также при выполнении и разборе контрольных работ. Достигнутые результаты непосредственно влияют на рейтинг, с которым студент выходит на экзамен.

Экзаменационный билет состоит из двух теоретических вопросов. По каждому из них задаются дополнительные задания качественного характера. Студент должен показать

понимание вопросов и применение теоретических положений на практике. К сожалению, это бывает не всегда. Некоторые студенты, даже обладая хорошей техникой программирования, с трудом усваивают теоретический материал. Это приводит к тому, что разработанные программы правильно работают на небольших по объему тестах, но "виснут" на реальных данных.

Совершенно недопустимо заучивать термины и определения без уяснения их смысла. Особое внимание стоит обратить на те вопросы, по которым в семестре выявлялись определенные проблемы.

Критерии оценки за экзаменационный ответ:

На тройку необходимо знать основные теоретические положения по каждому вопросу билета.

На четверку необходимо показать уровень тройки и дать полную и правильную интерпретацию применения теории по заданному преподавателем примеру (привести подробные шаги алгоритма, представить программные фрагменты, обосновать получаемые результаты, описать необходимые структуры данных).

На пятерку необходимо показать уровень четверки и выполнить дополнительное задание по связанным с вопросом темам (привести другие подходы, описать применение в других областях, оценить эффективность и указать способы ее повышения и т. п.).

В случае спорной оценки может быть дано дополнительное задание, в том числе не связанное напрямую с вопросами билета.

Для успешной сдачи экзамена необходимо иметь конспект лекций. Учебное пособие следует использовать в качестве консультанта по неясным вопросам. С другой стороны, в учебном пособии могут отсутствовать важные примеры, приводимые на лекциях;

Конечная оценка по дисциплине формируется в соответствии с правилами системы РИТМ.

Все расчеты по формулам проводятся по нормированным баллам: минимум – 40, максимум – 60.

На последней учебной неделе семестра обучающийся, полностью выполнивший программу, т.е. набравший не менее 40 баллов, допускается к итоговому контрольному испытанию (опрос). Итоговое контрольное задание максимально оценивается в 20 баллов.

Обучающиеся, набравшие 14 и более баллов по итогам контрольного испытания, освобождаются от экзамена. Суммарный балл, на основании которого выставляется экзаменационная оценка по четырех балльной шкале, определяется по формуле

$$N^C = N_{\text{обяз}} + N_{\text{доп}} + N_{\text{ск}}$$

где $N_{\text{обяз}}$ – баллы за обязательные виды работ, $N_{\text{доп}}$ – баллы за дополнительные работы, $N_{\text{ск}}$ – количество баллов по итогам семестрового контроля, N^C – суммарный балл.

Дополнительные баллы начисляются: за подготовку мини-доклада – 10 баллов.

Любой обучающийся, участвующий в системе РИТМ, имеет право освобождения от экзамена при условии, что он выдержал итоговый семестровый контроль.

Обучающиеся, освобожденные от экзамена, могут сдавать экзамен с целью повышения суммарного балла. В этом случае им гарантируется оценка, полученная по итогам работы в семестре.

Обучающиеся, набравшие на контрольном испытании менее 14 баллов, остаются участниками системы РИТМ и сдают экзамен.

Балл N_{Σ} за экзамен определяется в интервале от 20 до 40 баллов с учетом качества ответа.

Для всех обучающихся, сдающих экзамен, итоговый балл, при положительной оценке на экзамене, определяется по формуле

$$N^C = (N_{\text{обяз}} + N_{\text{доп}}) + N_{\text{экз.}}$$

Накопленные баллы (расчетные единицы) переводятся в четырех балльную шкалу:

- «отлично» - 90 и более
- «хорошо» - от 75 – до 89
- «удовлетворительно» - от 50 – до 74
- «неудовлетворительно» - менее 50.

Обучающиеся, не набравшие за текущую работу в семестре 40 баллов, считаются выбывшими из РИТМа и обязаны сдавать экзамен после того, как выполнят всю запланированную работу.

Обучающийся, который погашает свои задолженности после окончания сессии, получает минимальный балл за оценку, выставленную на экзамене за вычетом 7 баллов:

- «отлично» - 83
- «хорошо» - 68
- «удовлетворительно» - 43

Раздел 6. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ И УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

6.1. Учебно-методическое обеспечение

№№ п/п	Список используемой литературы	Количество экземпляров печатных изданий, имеющих в библиотеке, или электронный адрес издания (ресурса) в сети Интернет
УЧЕБНЫЕ, УЧЕБНО-МЕТОДИЧЕСКИЕ И НАУЧНЫЕ ИЗДАНИЯ		
1.	Галочкин, Владимир Иванович. Структуры и алгоритмы обработки данных [Текст] : учеб. пособие / В. И. Галочкин. Йошкар-Ола: МарГТУ, 2004. - 147 с. ISBN 5-8158-0350-2. Экземпляры: всего 58.	58
2.	Галочкин, Владимир Иванович. Базы данных [Текст] : учеб. пособие / В. И. Галочкин. Йошкар-Ола: МарГТУ, 2009. - 199 с. ISBN 978-5-8158-0688-7. Экземпляры: всего 90.	90
3.	Галочкин, Владимир Иванович. Алгоритмы и программы [Текст] : задачи повышенной сложности : учеб. пособие / В. И. Галочкин; М-во образования и науки Рос. Федерации, ФГБОУ ВПО "Мар. гос. техн. ун-т". Йошкар-Ола: МарГТУ, 2012. - 207 с. ISBN 978-5-8158-0968-0. Экземпляры: всего 88.	88 / https://portal.volgatech.net/books/Galochkin_Algoritmy_i_programmy.pdf
ЭЛЕКТРОННЫЕ ОБРАЗОВАТЕЛЬНЫЕ РЕСУРСЫ		
1.	Научная электронная библиотека eLIBRARY.RU	http://elibrary.ru
2.	ОСНОВЫ АЛГОРИТМОВ И СТРУКТУР ДАННЫХ	https://www.elibrary.ru/item.asp?id=25309178

6.2. Материально-техническая база и программное обеспечение

№№ п/п	Аудитории для проведения учебных занятий, самостоятельной работы и проведения государственной итоговой аттестации	Перечень основного оборудования	Программное обеспечение
1.	522 (I)	Персональный компьютер 3 Atlant A2X4/4G(3)/512Mb/монитор Pyama 2209/3Y (1), ПК RAMEC GALE LCD LG 23"/Intel i5 4590/MSI B85M-E45/2x4DDR3/GT740 2Gb/500Gb/клав,мышь (28), Проектор мультимедийный Hitachi CP-EX250 (1), Проектор мультимедийный Hitachi CP-EX251N (1), Комплект учебной мебели (1)	Microsoft Office Standard, Агент Dr.Web, Комплект ПО для решения основных пользовательских задач

Раздел 7. ФОРМЫ КОНТРОЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ/ ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

Критерии оценивания индикаторов достижения компетенций направлены на:

- усвоение теоретического материала (объем знаний, глубина усвоения), предусмотренного рабочей программой;
- умение излагать материал (четкость, грамотность изложения материала, точность и

полнота воспроизведения учебного материала);
 - умение применять теоретические знания при решении практических заданий.
 Шкала оценивания представлена ниже.

Уровень сформированности элементов компетенции	Критерии оценивания	Шкала оценивания
Пороговый уровень	Обучающийся имеет знания основного материала, проявляет умение логично его излагать, но может допускать неточности в изложении материала, недостаточно правильные формулировки, испытывает затруднения в выполнении практических заданий.	удовлетворительно
Продвинутый уровень	Обучающийся твердо знает программный материал, излагает его грамотно и по существу, не допускает существенных неточностей в ответе на вопрос, правильно применяет теоретические положения при решении практических вопросов и задач, владеет необходимыми навыками и приемами их выполнения	хорошо
Высокий уровень	Обучающийся глубоко и прочно усвоил программный материал, грамотно и логически стройно его излагает, дает исчерпывающие ответы на поставленные вопросы. В ответе тесно увязывается теория с практикой, при этом обучающийся не затрудняется с ответом при видоизменении задания, свободно справляется с задачами, вопросами и другими видами применения знаний, показывает знакомство с монографической литературой, периодическими изданиями, правильно обосновывает принятые решения, свободно владеет разносторонними навыками, приемами выполнения практических работ	отлично

7.1. Текущий контроль успеваемости

Текущий контроль успеваемости обеспечивает оценивание хода освоения дисциплины (модуля) и производится с применением технологии рейтингового контроля в соответствии с технологической картой дисциплины. Порядок составления технологической карты и алгоритм проведения процедуры оценивания видов деятельности обучающихся, направленных на освоение знаний, умений, навыков и/или опыта деятельности, по накопительной системе в баллах устанавливается положением о системе РИТМ в ФГБОУ ВО «ПГТУ»

7.2. Промежуточная аттестация обучающихся

Промежуточная аттестация обучающихся направлена на оценивание результатов обучения по дисциплине (модулю) и проводится с использованием фондов оценочных средств.

Примеры типовых контрольных заданий из базы фонда оценочных средств по образовательной программе.

1. Из следующих структур и методов

- 1) очередь
- 2) стек

- 3) рекурсия
- 4) волновой алгоритм

для перебора вариантов с возвратом можно применять

- 1) 1 и 2
- 2) 2 и 3
- 3) 3 и 4
- 4) 1 и 4
- 5) только 1
- 6) только 2
- 7) только 3.

2. Из следующих структур и методов

- 1) стек
- 2) рекурсия
- 3) очередь
- 4) дерево вариантов

для поиска в ширину можно применять

- 1) 1 и 2
- 2) 2 и 3
- 3) 3 и 4
- 4) 1 и 4
- 5) только 1
- 6) только 2
- 7) только 3.

3. Алгоритмы поиска в глубину какого-либо пути на графе из начальной вершины в конечную и перечисления всех таких путей имеют сложность:

- 1) Оба полиномиальную
- 2) Оба экспоненциальную
- 3) Первый алгоритм полиномиальную, второй экспоненциальную
- 4) Первый алгоритм экспоненциальную, второй полиномиальную.

4. Рекуррентное соотношение определяет результат на текущем шаге в зависимости

- 1) только от предыдущего шага
- 2) не более, чем от двух последних шагов
- 3) возможно, от всех предыдущих шагов

- 4) возможно, от всех предыдущих шагов от порядка перехода от состояния к состоянию на предыдущих шагах.
5. В динамическом программировании первый проход выполняется
 - 1) всегда от конца к началу
 - 2) всегда от начала к концу
 - 3) от конца к началу или от начала к концу
 - 4) от некоторого срединного состояния многошагового процесса к его началу и концу.
6. В динамическом программировании условным управлением считают:
 - 1) решение, принятое при условии независимости порядка выбираемых шагов
 - 2) решение, принятое при условии попадания процесса в данное состояние
 - 3) решение, принятое без полной гарантии выполнения условий
 - 4) решение, принятое при условии фиксированной стоимости каждого шага многошагового процесса.
7. Алгоритм Прима построения остова по сравнению с алгоритмом Краскала
 - 1) всегда более трудоемкий
 - 2) всегда менее трудоемкий
 - 3) более трудоемкий для разреженных графов
 - 4) менее трудоемкий для разреженных графов.
8. Обход в глубину неориентированного графа из N вершин и E ребер с целью нумерации его вершин имеет трудоемкость:
 - 1) $O(N)$
 - 2) $O(N+E)$
 - 3) $O(N \log N)$
 - 4) $O(N^2 + NE)$
 - 5) $O(\text{Exp}(N))$.
9. Поиск максимального пути между двумя вершинами на ориентированном графе с положительными длинами дуг имеет в общем случае трудоемкость:
 - 1) $O(N)$
 - 2) $O(N \log N)$
 - 3) $O(N^2)$
 - 4) $O(N^3)$
 - 5) $O(\text{Exp}(N))$.
10. Не требует предварительного анализа образца
 - 1) алгоритм Рабина

- 2) последовательный алгоритм
- 3) алгоритм Кнута-Морриса-Пратта
- 4) алгоритм Бойера-Мура.

11. Линейную оценку трудоемкости для всех случаев имеет алгоритм:

- 1) последовательный
- 2) Рабина
- 3) Кнута-Морриса-Пратта
- 4) Бойера-Мура.

12. Быстрейшим из неспециализированных алгоритмов поиска образца считается:

- 1) алгоритм Рабина
- 2) последовательный алгоритм
- 3) алгоритм Кнута-Морриса-Пратта
- 4) алгоритм Бойера-Мура.

13. Из следующих операций над массивом

- 1) нахождение минимума на диапазоне индексов
- 2) нахождение частичной суммы на диапазоне индексов
- 3) прибавления некоторой величины к одному из элементов
- 4) прибавления некоторой величины ко всем элементам из заданного диапазона индексов

дерево Фенвика позволяет выполнять с трудоемкостью $O(\log N)$ операции:

- 1) 1 и 2
- 2) 2 и 3
- 3) 3 и 4
- 4) 1 и 4
- 5) все операции
- 6) только 1
- 7) только 2
- 8) только 3
- 9) ни одну из операций.

14. Изменение элементов массива на новую величину в заданном диапазоне индексов с трудоемкостью $O(\log N)$ возможно с помощью:

- 1) дерева отрезков
- 2) дерева Фенвика
- 3) обоих деревьев

- 4) ни одного из деревьев
15. Построение дерева Фенвика имеет трудоемкость:
- 1) $O(\sqrt{N})$
 - 2) $O(N)$
 - 3) $O(N \log N)$
 - 4) $O(N^2)$.
16. Рекурсивный алгоритм поиска в глубину кратчайшего пути между двумя вершинами на взвешенном графе имеет трудоемкость:
- 1) $O(N)$
 - 2) $O(N \log N)$
 - 3) $O(N^2)$
 - 4) $O(N^3)$
 - 5) $O(\text{Exp}(N))$.
17. Требуется не менее N раз находить запись с определенным значением ключа в массиве. Сравниваются два алгоритма. Первый из них находит необходимую запись перебором. Во втором массив оптимальным образом сортируется по ключу и далее применяется бинарный поиск. Эти алгоритмы имеют трудоемкость соответственно:
- 1) $O(N \log N)$ и $O(N \log N)$
 - 2) $O(N \log N)$ и $O(N^2)$
 - 3) $O(N^2)$ и $O(N \log N)$
 - 4) $O(N^2)$ и $O(N^2)$
 - 5) не указанную в пунктах 1-4.
18. Бинарный и тернарный поиск применяются соответственно для нахождения:
- 1) экстремумов унимодальных функций и нулей монотонных функций
 - 2) нулей монотонных функций и экстремумов унимодальных функций
 - 3) экстремумов унимодальных функций – оба метода
 - 4) нулей монотонных функций – оба метода.

Перечень вопросов для проведения промежуточной аттестации

1. Перебор с возвратом. Окраска 4-связной области.
2. Поиск в ширину. Волновой алгоритм.
3. Рекуррентные соотношения.
4. Динамическое программирование. Задача о черепашке.
5. Динамическое программирование. Задачи о камнях и рюкзаке.

6. Целочисленная арифметика. Наибольший общий делитель и наименьшее общее кратное.
7. Простота чисел. Решето Эратосфена.
8. Длинная арифметика.
9. Метод сканирующей прямой.
10. Перестановки. Алгоритм лексикографического перечисления перестановок.
11. Перестановки. Вектора инверсий. Выдача перестановок путем транспозиции смежных элементов.
12. Подмножества множеств. Коды Грея. Выдача K-подмножеств в лексикографическом порядке.
13. Трудоемкость алгоритмов. Примеры оценки трудоемкости.
14. Вычислительная геометрия. Скалярное и векторное произведения. Выпуклая оболочка. Алгоритм Джарвиса.
15. Алгоритм Грэхема нахождения выпуклой оболочки.
16. Строки. Задача поиска образца. Последовательный алгоритм и Алгоритм Рабина.
17. Алгоритм Кнута-Морриса-Пратта поиска образца.
18. Алгоритм Бойера-Мура поиска образца.
19. Графы. Остовные деревья. Алгоритмы Прима и Краскала.
20. Графы. Обход в глубину. Связность и сильная связность.
21. Графы. Топологическая сортировка. Циклы и ацикличность.
22. Игры двух лиц. Дерево игры. Принцип минимакса.
23. Игры двух лиц. Альфа-бета процедура отсечения.
24. Игры двух лиц. Примеры игр с известной стратегией.
25. Бинарная куча. Возможности. Оценка трудоемкости операций.
27. Дерево Фенвика. Возможности. Оценка трудоемкости операций.
26. Дерево отрезков. Возможности. Оценка трудоемкости операций.
28. Принципы тестирования. Нагрузочные тесты. Генераторы тестов. Чекеры. Проверка на малых размерностях.

