МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

Школа естественных наук

«СОГЛАСОВАНО» Руководитель ОП Гузев М.А. (подпись) (Ф.И.О.) « 5 » марта 2021 г.	«УТВЕРЖДАЮ» Заведующий кафедрой Чеботарев А.Ю. (подпись) (Ф.И.О.) « 5 » марта 2021 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ
Объектно-ориентированный анализ и проектирование
Направление подготовки 09.03.03 Прикладная информатика
(Прикладная информатика в компьютерном дизайне)
Форма подготовки очная

курс 1 семестр 2
лекции 36 час.
практические занятия не предусмотрены
лабораторные работы 36 час.
в том числе с использованием МАО лек. ____ / пр. - / лаб. 36 час.
всего часов аудиторной нагрузки 72 час.
в том числе с использованием МАО 36 час.
самостоятельная работа 144 час.
в том числе на подготовку к экзамену 54 час.
контрольные работы (количество) не предусмотрены
курсовая работа / курсовой проект не предусмотрены
зачетне предусмотрен
экзамен 2 семестр

Рабочая программа составлена в соответствии с требованиями Федерального государственного образовательного стандарта по направлению подготовки 09.03.03 Прикладная информатика утвержденного приказом Министерства образования и науки РФ от от 19 сентября 2017 г. № 922

Рабочая программа обсуждена на заседании кафедры информатики, математического и компьютерного моделирования протокол №6 от « 5 » марта 2021г.

Заведующий кафедрой: Чеботарев А.Ю.
Составитель (ли): _____

Владивосток

2021

Оборотная сторона титульного листа РПД

I. Рабочая программа пересмотрена на заседании кафедры/департамента:

Протокол от «_____» _____ 20___ г. № _____

Заведующий кафедрой _____ _____
(подпись) (И.О. Фамилия)

II. Рабочая программа пересмотрена на заседании кафедры/департамента:

Протокол от «_____» _____ 20___ г. № _____

Заведующий кафедрой _____ _____
(подпись) (И.О. Фамилия)

III. Рабочая программа пересмотрена на заседании кафедры/департамента:

Протокол от «_____» _____ 20___ г. № _____

Заведующий кафедрой _____ _____
(подпись) (И.О. Фамилия)

IV. Рабочая программа пересмотрена на заседании кафедры/департамента:

Протокол от «_____» _____ 20___ г. № _____

Заведующий кафедрой _____ _____
(подпись) (И.О. Фамилия)

АННОТАЦИЯ

Рабочая программа дисциплины «Объектно-ориентированный анализ и проектирование» разработан для студентов направления «Прикладная информатика».

Дисциплина «Объектно-ориентированный анализ и проектирование» входит в базовую часть профессионального цикла.

Содержание дисциплины охватывает знания о технологических принципах разработки и сопровождения программных систем среднего и большого размера, в том числе в составе коллектива разработчиков.

Рассматриваются основные цели технологического подхода к программированию — повышение воспроизводимости, надежности и эффективности процесса разработки программного обеспечения. Уделяется внимание глубокому изучению наиболее распространённых конкретных технологий программирования, используемых ими организационных и технических инструментов. Также поверхностно рассматриваются юридические, экономические, этические и философские аспекты деятельности программиста.

Дисциплина «Объектно-ориентированный анализ и проектирование» логически и содержательно связана с такими курсами, как «Web-программирование», «Практикум на ЭВМ», «Базы данных».

Цели освоения дисциплины

В результате освоения данной дисциплины бакалавр приобретает знания, умения и навыки, обеспечивающие достижение целей основной образовательной программы «Прикладная информатика».

Дисциплина должна:

1. познакомить студентов с фундаментальными проблемами разработки сложных систем и историей их преодоления;
2. познакомить студентов с основными целями технологического подхода к разработке и арсеналом современных средств для достижения этих целей;
3. научить студентов вести разработку в составе коллектива программистов;
4. научить студентов анализ предметной области, взаимодействие с заказчиком, проектирование систем нетривиального размера;
5. познакомить студентов выбирать использовать технические средства поддержки процесса разработки.

Место дисциплины в структуре ОП бакалавриата

Дисциплина «Объектно-ориентированный анализ и проектирование» относится к циклу профессиональных дисциплин ОП.

Для изучения дисциплины студент должен:

Знать:

- основы алгоритмизации и программирования;
- базовые инструменты проектирования и структурирования программных продуктов.

Уметь:

- программировать нескольких алгоритмических языках;
- вести индивидуальную разработку программных систем небольшой сложности.

Владеть:

- методами алгоритмизации и программирования;
- навыками разработки, отладки и сопровождения небольших приложений;
- навыками коммуникации, как очной так и с помощью электронных средств связи.

Компетенции выпускника ОП бакалавриата, формируемые в результате освоения данной ОП ВПО.

В результате освоения дисциплины обучающийся должен обладать компетенциями:

- *Общекультурными*:

• способностью понимать сущность и значение информации в развитии современного информационного общества, сознавать опасности и угрозы, возникающие в этом процессе, соблюдать основные требования информационной безопасности, в том числе защиты государственной тайны

• способностью осознать социальную значимость своей будущей профессии, обладать высокой мотивацией к выполнению профессиональной деятельности;

• способностью владения навыками работы с компьютером как средством управления информацией;

• способностью работать с информацией в глобальных компьютерных сетях;

• способностью использовать в научной и познавательной деятельности, а также в социальной сфере профессиональные навыки работы с информационными и компьютерными технологиями;

• способностью работы с информацией из различных источников, включая сетевые ресурсы сети Интернет, для решения профессиональных и социальных задач.

2. Профессиональными:

способностью демонстрации общенаучных базовых знаний естественных наук, математики и информатики, понимание основных фактов, концепций, принципов теорий, связанных с прикладной математикой и информатикой;

способностью в составе научно-исследовательского и производственного коллектива решать задачи профессиональной деятельности;

способностью применять в профессиональной деятельности современные языки программирования и языки баз данных, операционные системы, электронные библиотеки и пакеты программ, сетевые технологии;

В результате освоения дисциплины студент будет:

Знать о:

1. способах внедрения и сопровождения программной системы;
2. понятии и назначении технологии, в частности технологии программирования;
3. истории развития технологий программирования;
4. структуре жизненного цикла программного продукта;
5. целях и способах анализа предметной области;
6. целях и способах проектирования программной системы;
7. целях и способах разработки программной системы;
8. целях и способах отладки программной системы;
9. целях и способах внедрения и сопровождения программной системы;
10. понятии интеллектуальной собственности, юридических и этических аспектах разработки программного обеспечения;
11. экономических аспектах разработки программного обеспечения, особенностях рынка труда программистов и рынка продажи программ;
12. видах и особенностях требований к программному обеспечению;
13. конкретных технологиях разработки, в том числе формальных методах, объектно-ориентированном и функциональном анализе, Agile-методологиях и других;
14. проблемах и методах организации работы творческих коллективов, в особенности программистских;
15. программных и организационных инструментах поддержки разработчиков.

Уметь:

1. проводить анализ предметной области, взаимодействовать с экспертами в предметной области для постановки задачи;
2. оценивать трудоёмкость и планировать процесс разработки программного продукта средней сложности;
3. участвовать во всех этапах жизненного цикла программного продукта на любой роли;
4. выбирать и использовать инструменты поддержки разработки;
5. взаимодействовать с другими разработчиками в составе коллектива.

Владеть:

1. практическим опытом разработки нетривиального программного продукта;
2. навыками разделения труда в составе творческого коллектива;
3. навыками применения технических и организационных средств поддержки разработки.

Тип задач	**Код и наименование профессиональной компетенции** **(результат освоения)**	**Код и наименование индикатора достижения компетенции**
научно-исследовательский	ПК-7 Способен проводить научно-исследовательские и опытно-конструкторские разработки в области цифровизации предприятий	ПК-7.1 Демонстрирует знание методологий науки и техники, методов исследования объектов профессиональной деятельности и разработки моделей, способов обеспечения качества исследований и требований стандартов по оформлению научно-исследовательских отчетов
		ПК-7.2 Исследует объекты профессиональной деятельности, выявляет и идентифицирует актуальные проблемы, предлагает гипотезы, формирует цели и задачи исследований и разработки, осуществляет сбор и обработку результатов проектных исследований, предлагает варианты решений, осуществляет выбор, составляет отчеты о проделанной работе, обзоры
		ПК-7.3 Разрабатывает модели объектов профессиональной деятельности, осуществляет оценку полученного результата, определяет качество проводимых исследований, составляет отчеты о

		проделанной работе, обзоры, готовит публикации

Код и наименование индикатора достижения компетенции	**Наименование показателя оценивания (результата обучения по дисциплине)**
ПК-7.1 Демонстрирует знание методологий науки и техники, методов исследования объектов профессиональной деятельности и разработки моделей, способов обеспечения качества исследований и требований стандартов по оформлению научно-исследовательских отчетов	Знать: методологии науки и техники, методов исследования объектов профессиональной деятельности и разработки моделей, способов обеспечения качества исследований и требований стандартов по оформлению научно-исследовательских отчетов, исследования объектов профессиональной деятельности.
	Уметь: использовать методологии науки и техники, методов исследования объектов профессиональной деятельности.
	Владеть: навыками использования знаний естественнонаучных дисциплин, вычислительной техники и программирования для решения общих задач естествознания, техники, навыками применения знаний к теоретически м и практическим исследованиям
ПК-7.2 Исследует объекты профессиональной деятельности, выявляет и идентифицирует актуальные проблемы, предлагает гипотезы, формирует цели и задачи исследований и разработки, осуществляет сбор и обработку результатов проектных исследований, предлагает варианты решений, осуществляет выбор, составляет отчеты о проделанной работе, обзоры	Знать: основы профессии, принципы архитектуры вычислительной техники и системы программного обеспечения; программную инженерию, технологии программирования и способы реализации программных проектов.
	Уметь: корректно ставить профессиональные задачи; использовать методы математического и алгоритмического моделирования при решении теоретических и прикладных задач; самостоятельно проводить анализ результатов научно-исследовательской работы, делать обоснованные вывод.
	Владеть: способностью использовать профессиональные методы при анализе проблем в области профессиональной деятельности; способностью участвовать в создании информационных и компьютерных систем, программных проектов на всех этапах жизненного цикла.
ПК-7.3 Разрабатывает модели объектов профессиональной деятельности,	Знать: определения и свойства основных объектов профессиональной деятельности

осуществляет оценку полученного результата, определяет качество проводимых исследований, составляет отчеты о проделанной работе, обзоры, готовит публикации	уметь: решать задачи вычислительного и теоретического характера, находить оптимальные решения с наименьшим риском ошибки.
	владеть: разнообразным профессиональным разработки, описания и оценки моделей объектов профессиональной деятельности

Общая трудоемкость дисциплины составляет 6 зачётных единиц (216 академических часов). 1 зачетная единица соответствует 36 академическим часам.

Видами учебных занятий и работы обучающегося по дисциплине могут являться:

Обозначение	Виды учебных занятий и работы обучающегося
Лек	Лекции
Лаб	Лабораторные работы
СР	Самостоятельная работа обучающегося в период теоретического обучения
Контроль	Самостоятельная работа обучающегося и контактная работа обучающегося с преподавателем в период промежуточной аттестации

I. СТРУКТУРА И СОДЕРЖАНИЕ ТЕОРЕТИЧЕСКОЙ ЧАСТИ КУРСА

Модуль 1. Введение и обзор (12 час.)

Тема 1.1. Сложные системы

• Требования к отчётности. Обзор курса.

• Понятие сложности и сложной системы.

• Математические и физические основы теории сложных систем.

• Теория хаоса. Непредсказуемость сложных систем.

• Борьба со сложностью как фундаментальная проблема программирования.

Тема 1.2. Технологии

• Понятие и назначение технологии.

• Примеры технологий.

• Технологии программирования как частный случай технологий.

• Классификация и исторический обзор технологий программирования.

Тема 1.3. Коллективная разработка.

• Классификация коллективной деятельности, особенности творческих коллективов.

• Сублинейный рост производительности труда как основная проблема крупных творческих коллективов.

• Методы и трудности объективной оценки и мотивации творческой деятельности.

• Конкретные критерии оценки производительности программистов и их недостатки, на примере технологии CMM.

• Открытая разработка как пример эффективной организации больших коллективов программистов.

Тема 1.4. Жизненный цикл программного продукта.

• Понятие жизненного цикла, основные этапы.

• Анализ предметной области.

• Проектирование программного продукта.

• Разработка.

• Тестирование.

• Сопровождение и эксплуатация.

Тема 1.5. Технологии управления жизненным циклом.

• Технология waterfall, кризис её применения.

• Повторяемость жизненного цикла.

• Инкрементная и итеративная разработка.

• Технологии короткого цикла, Agile, SCRUM.

Тема 1.6. Интеллектуальная собственность.

• Виды интеллектуальной собственности.

• Авторское право, лицензии.

• Закрытые и открытые лицензии.

• Свободное программное обеспечение.

• Этические и философские аспекты разработки и распространения программного обеспечения.

Модуль 2. Системный подход и системный анализ (12 час.)

Тема 2.1. Основные понятия формальной логики.

• Объект автоматизации (информатизации).

• Концептуальная модель предметной области и ее проблематизация.

• Определение задач проектирования. Решение задач проектирования.

• Определение логической схемы проектирования.

• Отношения: ассоциации, обобщения, реализация, зависимости.

Тема 2.2. Методология проектирования программных продуктов.

• Определение задач проектирования.

• Представление вариантов использования.
• Представление взаимодействия.
• Диаграмма последовательности.
• Физическое представление.
• Представление управления моделью.

Тема 2.3. Реализация программной системы.

• Моделирование и описание архитектуры программной системы. Компоненты системы.

• Сравнение полученных характеристик качества с заданными характеристиками качества и определение отклонений. Определение задач по устранению отклонений.

• Управление разработкой программного обеспечения. Управление конфигурацией системы.

• Технология и методология тестирования.

• Сопровождение и эксплуатация ИС.

Модуль 3. Объектно-ориентированной анализ (12 час.)

Тема 3.1. Методы системного анализа прикладной области.

• Технико-экономического обоснования проектных решений

• Качества автоматизации и информатизации решения прикладных задач

Тема 3.2. Модели системы.

• Общие принципы, суть и цели моделирования.
• Варианты применения в процессе проектирования
• Назначение модели.
• Уровни моделей

Тема 3.3. Объектно-ориентированная разработки систем.

• Методы и технологии объектно-ориентированной разработки систем.

• Составляющие проекта: статическая, динамическая, организационная, относящаяся к окружению.

II. СТРУКТУРА И СОДЕРЖАНИЕ ПРАКТИЧЕСКОЙ ЧАСТИ КУРСА

Практические занятия организованы в виде лабораторных занятий по темам, продолжающим тему соответствующей лекции.

Лабораторная работа №1 (8 час.)

Демонстрация общенаучных базовых знаний естественных наук, математики и информатики, понимание основных фактов, концепций, принципов теорий, связанных с прикладной математикой и информатикой.

Лабораторная работа №2 (8 час.)

Решение задачи профессиональной деятельности в составе научно-исследовательского и производственного коллектива, проблемы и методы организации работы творческих коллективов, в особенности программистских, разделения труда в составе творческого коллектива, взаимодействие с другими разработчиками в составе коллектива.

Лабораторная работа №3 (8 час.)

Применение в профессиональной деятельности современных языков программирования и языков баз данных, операционных систем, электронных библиотек и пакетов программ, сетевых технологий, использование инструментов поддержки разработки.

Лабораторная работа №4 (8 час.)

Сложные программные системы, их свойства фундаментальные проблемы при их создании, принципы устройства работоспособных сложных информационных систем, разработка нетривиального программного продукта, применение формальных методов, объектно-ориентированного и функционального анализа, Agile-методологий и пр.

Лабораторная работа №5 (8 час.)

Понятие технологии, ее назначение, технология программирования, истории развития технологий программирования, структура жизненного цикла программного продукта, применения технических и организационных средств поддержки разработки, роли разработчиков на этапах жизненного цикла программного продукта.

Лабораторная работа №6 (6 час.)

Технологические процессы этапов жизненного цикла, проведение анализа предметной области различными способами, организация проектирования программной системы, разработка программной системы, отладка программной системы, внедрение и сопровождение программной системы, взаимодействие с экспертами в предметной области для постановки задачи.

Лабораторная работа №7(8 час.)

Экономические аспекты разработки программного обеспечения, оценка трудоёмкости и планирование процесса разработки программного продукта средней сложности, особенности рынка труда программистов и рынка продажи программ, понятие интеллектуальной собственности, юридические и этические аспекты разработки программного обеспечения, требования к программному обеспечению.

III. УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ

Учебно-методическое обеспечение самостоятельной работы обучающихся по дисциплине «Основы информатики и программирования» представлено в Приложении 1 и включает в себя:

план-график выполнения самостоятельной работы по дисциплине, в том числе примерные нормы времени на выполнение по каждому заданию;

характеристика заданий для самостоятельной работы обучающихся и методические рекомендации по их выполнению;

требования к представлению и оформлению результатов самостоятельной работы;

критерии оценки выполнения самостоятельной работы.

План-график выполнения самостоятельной работы по дисциплине

№ п/п, название	**Дата/сроки выполнения**	**Вид СРС**	**Примерные нормы времени на выполнение**	**Форма контроля**
1. Понятие технологии, особенности технологии программирования	Вторая неделя 2 семестра	ИДЗ	2 недели	Коллоквиум
2. Прототипное проектирование информационных систем	Четвертая неделя 2 семестра	ИДЗ	1 неделя	Коллоквиум
1. Типовое проектирование информационных систем.	Пятая неделя 2 семестра	ИДЗ	2 недели	Документальный отчет
1. Оценка работоспособности информационных систем	Шестая неделя 2 семестра	ИДЗ	3 неделя	Реферат, доклад
1. Методики обеспечения корректности: тестирование и отладка	Восьмая неделя 2 семестра	ИДЗ	4 недели	Программный код
3. Язык UML	Десятая неделя 2 семестра	ИДЗ	2 недели	Коллоквиум
4. Прототип проекта	Двенадцатая неделя 2 семестра	ИДЗ	1 неделя	Отчетная документация

5. Оценка функционального пакета прикладных программ.	Тринадцатая неделя 2 семестра	ИДЗ	1 неделя	Собеседование
6. Система контроля версий: назначение, классификация, принципы работы	Четырнадцатая неделя 2 семестра	ИДЗ	1 неделя	Практическое задание
7. Разработка диаграмм UML проекта	Пятнадцатая неделя 2 семестра	ИДЗ	1 неделя	Документальный отчет
8. Проект и реализация программного продукта	Шестнадцатая неделя 2 семестра	ИДЗ	1 неделя	Документальный отчет
1. Презентация программного продукта	Сессия	ИДЗ	1 час	Защита

Материалы для самостоятельной работы студентов подготовлены в виде индивидуальных домашних заданий по каждой теме (образцы типовых ИДЗ представлены в разделе «Материалы для самостоятельной работы студентов»). Работа должна быть отправлена преподавателю на проверку в системе Bb dvfu по соответствующему «Назначению». Оформление в формате PDF. Критерии оценки: студент получает максимальный балл, если работа выполнена без ошибок и оформлена в соответствии с требованиями преподавателя.

ТЕМЫ ДЛЯ САМОСТОЯТЕЛЬНОЙ ПОДГОТОВКИ

Вопросы для самопроверки знаний

1. История методологии разработки информационных систем.
2. Понятие технологии. Особенности технологии программирования.
3. Понятие и структура проекта информационной системы.
4. Основные понятия проектирования информационных систем .
5. Классификация методов проектирования информационных систем.
6. Параметрически-ориентированное проектирование информационных систем.
7. Технологическая сеть модельно-ориентированного проектирования информационных систем.
8. Прототипное проектирование информационных систем.
9. Жизненный цикл программного обеспечения.
10. Типовое проектирование информационных систем. Виды типового проектирования.

11. Методы типового проектирования информационных систем: элементный, подсистемный, объектный.
12. Стадии и этапы канонического проектирования ИС.
13. Автоматизированное проектирование информационных систем. Стадии и этапы процесса проектирования ИС с применением CASE–технологии.
14. Определение критериев оценки функционального пакета прикладных программ.
15. Оценка работоспособности информационных систем.
16. Методики обеспечения корректности: тестирование и отладка.
17. Особенности и методы коллективной разработки проектов.
18. Контроль версий: назначение, классификация, принципы работы.
19. Инструменты проектирования программных систем.
20. Визуальное проектирование, язык UML: назначение и общие принципы.
21. Диаграммы UML: вариантов использования.
22. Диаграммы UML: состояний.
23. Диаграммы UML: сущностей.
24. Диаграммы UML: классов.
25. Динамические диаграммы UML.
26. Базовые принципы разработки и реализации диалоговых систем.
27. Документирование программных систем.
28. Дополнительные вопросы
29. Модель функций.
30. Модель процессов.
31. Модели объектов (данных).
32. Модель организационной структуры.
33. Модели бизнес-правил.
34. Требования к программной системе.

УЧЕБНЫЕ МАТЕРИАЛЫ ЗАНЯТИЙ

Основные понятия

- Объектно-ориентированный анализ - дисциплина, изучающая технологические процессы программирования и порядок их прохождения.
- Аспектно-ориентированное сборочное программирование - разновидность сборочного программирования, основанная на сборке полнофункциональных приложений из многоаспектных компонентов, инкапсулирующих различные варианты реализации.

Восходящее программирование

- Программирование "снизу вверх"
- Восходящее программирование - методика разработки программ, при которой крупные блоки собираются из ранее созданных мелких блоков.

- Восходящее программирование начинается с разработки ключевых процедур и подпрограмм, которые затем постоянно модифицируются.

Диаграмма функционального моделирования

- Structured analysis and design technique (SADT)
- Диаграмма функционального моделирования - инструмент разработки функциональных спецификаций в виде диаграмм, фрагментов текста и глоссария, связанных перекрестными ссылками. В состав диаграммы входят:
- - блоки, изображающие активность моделируемой системы; и
- - дуги, связывающие блоки вместе и изображающие взаимодействия и взаимосвязи между ними.
- Место соединения дуги с блоком определяет тип интерфейса:
- - управляющая информация входит в блок сверху;
- - входная информация, подвергающаяся обработке, показывается с левой стороны блока;
- - выходная информация показывается с правой стороны;
- - механизм, осуществляющий операцию, представляется дугой, входящей в блок снизу.
- Заглушка - структурном программировании - процедура, представленная точной спецификацией заголовка и пустым телом. Заглушка позволяет компилировать и выполнять программу в отладочном режиме.

>> Императивное программирование

- Императивное программирование - технология программирования, характеризующаяся принципом последовательного изменения состояния вычислителя пошаговым образом. При этом управление изменениями полностью определено и полностью контролируемо.

Инструментарий технологии программирования

- Инструментарий технологии программирования - программные продукты, предназначенные для поддержки технологии программирования.

Компонентное сборочное программирование

- Компонентное сборочное программирование - объектно-ориентированное сборочное программирование, основанное на распространении классов в бинарном виде и предоставление доступа к методам класса через строго определенные интерфейсы.
- Компонентное сборочное программирование поддерживают технологические подходы COM, CORBA, .Net.

Компьютерный дарвинизм

- Компьютерный дарвинизм - подход к разработке программных систем, основанный на принципе восходящей разработки при интенсивном тестировании. Подход состоит из трех основных процессов: макетирования, тестирования и отладки.

- Логическое программирование - программирование в терминах фактов и правил вывода, с использованием языка, основанного на формальных исчислениях.
- Метод восходящего проектирования - подход, при котором в первую очередь определяются вспомогательные модули, которые потребуются для проектируемой программы.
- Метод расширения ядра - метод восходящего программирования, при котором основное внимание уделяется выявлению множества вспомогательных модулей, а не определению функции всей программы в целом.
- Модульное программирование - метод разработки программ, предполагающий разбиение программы на независимые модули. Считается, что:
- - оптимальный по размерам модуль целиком помещается на экране дисплея;
- - разделение большой программы на модули облегчает ее разработку, отладку и сопровождение.
- Модульное сборочное программирование - разновидность сборочного программирования, основанная на процедурах и функциях методологии структурного императивного программирования.
- Нисходящее программирование
- Программирование "сверху вниз"
- Нисходящее программирование - методика разработки программ, при которой разработка начинается с определения целей решения проблемы, после чего идет последовательная детализация, заканчивающаяся детальной программой.

>> Объектно-ориентированное программирование

- Объектно-ориентированное программирование - технология программирования, при которой программа рассматривается как набор дискретных объектов, содержащих, в свою очередь, наборы структур данных и процедур, взаимодействующих с другими объектами.
- Объектно-ориентированное сборочное программирование - разновидность сборочного программирования:
- - основанная на методологии объектно-ориентированного программирования; и
- - предполагающая распространение библиотек классов в виде исходного кода или упаковку классов в динамически компонуемую библиотеку.
- Сборочное программирование - технология программирования, при которой программа собирается посредством повторного использования уже известных фрагментов программ.
- Синтезирующее программирование - программирование, предполагающее синтез программы по ее спецификации.

Структурное программирование - методология и технология разработки программных комплексов, основанная на принципах:

- - программирования "сверху-вниз";
- - модульного программирования.
- При этом логика алгоритма и программы должны использовать три основные структуры: последовательное выполнение, ветвление и повторение.
- Компонент - Составная часть распределенного приложения
- Инкапсуляция, encapsulation - механизм, который объединяет данные и код, манипулирующий зтими данными, а также защищает и то, и другое от внешнего вмешательства или неправильного использования. В объектно-ориентированном программировании код и данные могут быть объединены вместе; в этом случае говорят, что создаётся так называемый "чёрный ящик". Когда коды и данные объединяются таким способом, создаётся объект (object). Другими словами, объект - это то, что поддерживает инкапсуляцию.
- Внутри объекта коды и данные могут быть закрытыми (private). Закрытые коды или данные доступны только для других частей этого объекта. Таким образом, закрытые коды и данные недоступны для тех частей программы, которые существуют вне объекта. Если коды и данные являются открытыми, то, несмотря на то, что они заданы внутри объекта, они доступны и для других частей программы. Характерной является ситуация, когда открытая часть объекта используется для того, чтобы обеспечить контролируемый интерфейс закрытых элементов объекта.
- На самом деле объект является переменной определённого пользователем типа. Может показаться странным, что объект, который объединяет коды и данные, можно рассматривать как переменную. Однако применительно к объектно-ориентированному программированию это именно так. Каждый элемент данных такого типа является составной переменной.
- Полиморфизм, polymorphism - (от греческого polymorphos) - это свойство, которое позволяет одно и то же имя использовать для решения двух или более схожих, но технически разных задач. Целью полиморфизма, применительно к объектно-ориентированному программированию, является использование одного имени для задания общих для класса действий. Выполнение каждого конкретного действия будет определяться типом данных. Например для языка Си, в котором полиморфизм поддерживается недостаточно, нахождение абсолютной величины числа требует трёх различных функций: abs(), labs() и fabs(). Эти функции подсчитывают и возвращают абсолютную величину целых, длинных целых и чисел с плавающей точкой соответственно. В C++ каждая из этих функций может быть названа abs(). Тип данных, который используется при вызове функции, определяет, какая конкретная версия функции действительно выполняется. В C++ можно использовать одно имя функции для

множества различных действий. Это называется перегрузкой функций (function overloading).

- В более общем смысле, концепцией полиморфизма является идея "один интерфейс, множество методов". Это означает, что можно создать общий интерфейс для группы близких по смыслу действий. Преимуществом полиморфизма является то, что он помогает снижать сложность программ, разрешая использование того же интерфейса для задания единого класса действий. Выбор же конкретного действия, в зависимости от ситуации, возлагается на компилятор. Вам, как программисту, не нужно делать этот выбор самому. Нужно только помнить и использовать общий интерфейс. Пример из предыдущего абзаца показывает, как, имея три имени для функции определения абсолютной величины числа вместо одного, обычная задача становится более сложной, чем это действительно необходимо.
- Полиморфизм может применяться также и к операторам. Фактически во всех языках программирования ограниченно применяется полиморфизм, например, в арифметических операторах. Так, в Си, символ + используется для складывания целых, длинных целых, символьных переменных и чисел с плавающей точкой. В этом случае компилятор автоматически определяет, какой тип арифметики требуется. В C++ вы можете применить эту концепцию и к другим, заданным вами, типам данных. Такой тип полиморфизма называется перегрузкой операторов (operator overloading).
- Ключевым в понимании полиморфизма является то, что он позволяет вам манипулировать объектами различной степени сложности путём создания общего для них стандартного интерфейса для реализации похожих действий.
- Наследование, inheritance - это процесс, посредством которого один объект может приобретать свойства другого. Точнее, объект может наследовать основные свойства другого объекта и добавлять к ним черты, характерные только для него. Наследование является важным, поскольку оно позволяет поддерживать концепцию иерархии классов (hierarchical classification).
- Применение иерархии классов делает управляемыми большие потоки информации. Например, подумайте об описании жилого дома. Дом - это часть общего класса, называемого строением. С другой стороны, строение - это часть более общего класса - конструкции, который является частью ещё более общего класса объектов, который можно назвать созданием рук человека. В каждом случае порождённый класс наследует все, связанные с родителем, качества и добавляет к ним свои собственные определяющие характеристики. Без использования иерархии классов, для каждого объекта пришлось бы задать все характеристики, которые бы исчерпывающи его определяли. Однако при использовании наследования можно описать объект путём определения того общего класса (или классов), к которому он

относится, с теми специальными чертами, которые делают объект уникальным. Наследование играет очень важную роль в ООП.

- Контейнер в программировании - структура, позволяющая инкапсулировать в себя объекты разных типов.
- Среди "широких масс" программистов наиболее известны контейнеры, построенные на основе шаблонов, однако существуют и реализации в виде библиотек (наиболее широко известна библиотека GLib). Кроме того, применяются и узкоспециализированные решения. Примерами контейнеров являются контейнеры из стандартной библиотеки (STL) - map, vector и др. В контейнерах часто встречается реализация алгоритмов для них. В ряде языков программирования (особенно скриптовых типа Perl или PHP) контейнеры и работа с ними встроена в язык.
- Контейнер, в отличие от коллекции, в общем случае, обычно не допускает явного задания числа элементов и обычно не поддерживает ветвистой структуры. Впрочем, это сильно зависит от реализации, поскольку многие реализации (особенно ориентированные на persistant storage) позволяют задавать размеры при создании контейнера.
- Распределенные вычисления - Парадигма организации приложений, в которой различные части программы могут исполняться на разных компьютерах в сети.
- Active Directory - Сетевая служба каталогов Microsoft, которую корпорация включила в состав Windows с версии 2000.
- ActiveX - Предлагаемый Microsoft способ разработки программных компонентов; название группы технологий, разработанных Microsoft для программирования компонентных объектных приложений на основе модели COM.
- ActiveX control - управляющий элемент ActiveX; введенное в 1996 г. Microsoft новое название независимых программируемых компонентов, ранее называемых OLE controls, OCXs, OLE custom controls; в отличие от последних позволяют работать с Internet.
- CDN, Content Delivery Network, Content Distribution Network, Сеть доставки и дистрибуции контента - географически распределённая сетевая инфраструктура, позволяющая оптимизировать доставку и дистрибуцию контента конечным пользователям в сети Интернет. Использование контент-провайдерами CDN способствует увеличению скорости загрузки интернет-пользователями аудио-, видео-, программного, игрового и других видов цифрового контента в точках присутствия сети CDN.
- Простыми словами, CDN-это промежуточный хостинг. Контент вашего сайта сначала загружается на CDN хостинг, а затем отдается пользователю. Часто используется в криминальных целях воровства контента, информационного подавления неугодных интернет-ресурсов, перехвата пользовательского трафика и позиций в поисковых системах.

- Популярные бесплатные CDN-сервисы: CloudFlare, Incapsula.
- COM, Component Object Model - Программная архитектура Microsoft, поддерживающая компонентный подход к разработке приложений; модель компонентных объектов Microsoft; стандартный механизм, включающий интерфейсы, с помощью которых одни объекты предоставляют свои сервисы другим; является основой многих объектных технологий, в том числе OLE и ActiveX).
- COM+ - Модернизация COM и Microsoft Transaction Server, который упрощает разработку сложных распределенных приложений.
- Common Object Request Broker Architecture, CORBA - Основной конкурент DCOM в области построения распределенных программных систем.
- DLL, Dynamic Link Library - динамически подключаемая библиотека, понятие операционной системы Microsoft Windows; динамическая библиотека, позволяющая многократное применение различными программными приложениями. К DLL иногда причисляют также элементы управления ActiveX и драйвера. В мире UNIX аналогичные функции выполняют т. н. shared objects (<разделяемые объекты>). Формат файлов *.dll придерживается тех же соглашений, что и формат исполняемых файлов *.exe, сочетая код, таблицы и ресурсы.
- Microsoft Transaction Server, MTS - Усовершенствование в составе COM, которое реализует поддержку транзакций баз данных.
- OLE, Object Linking and Embedding - общее название (до 1996 г.) группы объектно-ориентированных технологий Microsoft на основе COM (OLE 1, OLE 2, OLE automation, OLE Database и др.).
- OCX, OLE Custom eXtension - перемещаемые элементы управления, OLE custom control, OLE control. Упрощенно можно сказать, что файлы *.ocx - это элементы управления ActiveX, выполняющие примерно те же функции, что и файлы *.dll.
- - OLE custom control
- специализированный управляющий элемент OLE, OLE control.
- OLE control - управляющие элементы OLE, программируемые компоненты-приложения с интерфейсом на базе OLE, позволяющим легко включать их в другие приложения; с 1996 г. называются ActiveX control. Синонимы: OCX, OLE custom control.
- Remote Procedure Call, RPC - Удаленный вызов процедуры - сообщение, посылаемое по сети, которое позволяет программе, установленной на одном компьютере, инициировать выполнение необходимой операции на другом.

ТСР направлена на развитие интеллектуальных умений, комплекса универсальных (общекультурных) и профессиональных компетенций, повышение творческого потенциала бакалавров и заключается в:

- поиске, анализе, структурировании и презентации информации;

- подготовке рефератов и докладов;
- исследовательской работе и участии в научных студенческих семинарах и олимпиадах;
- анализе научных публикаций по заранее определенной преподавателем теме.

Материалы для самостоятельной работы студентов подготовлены в виде индивидуальных домашних заданий по каждой теме (образцы типовых ИДЗ представлены в разделе «Материалы для самостоятельной работы студентов»). Работа должна быть отправлена преподавателю на проверку в системе CATS либо по электронной почте по соответствующему адресу. Оформление: исходный или исполняемый файл в зависимости от требований задания. Критерии оценки: студент получает максимальный балл, если работа выполнена без ошибок и оформлена в соответствии с требованиями преподавателя.

ХАРАКТЕРИСТИКА ЗАДАНИЙ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. Ознакомление с основными концепциями динамического распределения памяти
2. Знакомство с научной и научно-популярной литературой по методам сортировки и поиска информации.
3.Знакомство с периодическими изданиями по теории алгоритмизации.
4. Изучение статей по дисциплине.
5. Знакомство с приложениями алгоритмов сортировки и поиска современных информационных технологиях.

ТРЕБОВАНИЯ К ПРЕДСТАВЛЕНИЮ И ОФОРМЛЕНИЮ РЕЗУЛЬТАТОВ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. Устный доклад по указанной теме.
2. Презентация к докладу по указанной теме.
3. Демонстрация программного кода (мультимедийный экран), подтверждение работоспособности программы.

   Дополнительный вариант:

   Реферат по указанной теме.

КРИТЕРИИ ОЦЕНКИ ВЫПОЛНЕНИЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. Качество доклада (10 баллов)
2. Качество презентации (10 баллов)
3. Качество программного кода (10 баллов)
4. Качество ответов на вопросы (10 баллов)

ТЕКУЩАЯ СРС

- работа с лекционным материалом, поиск и обзор литературы и электронных источников информации по заданной теме;
- изучение тем, вынесенных на самостоятельную проработку;
- подготовка к лабораторным работам;

- подготовка к зачёту.

СОДЕРЖАНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ ПО ДИСЦИПЛИНЕ

1. Изучение тем и разделов, не уложившихся в лекции.
2. Подготовка к экзамену.

КОНТРОЛЬНО- ИЗМЕРИТЕЛЬНЫЕ МАТЕРИАЛЫ

Контролируемые темы

1. Жизненный цикл программного обеспечения.
2. История методологии разработки информационных систем.
3. Понятие и структура проекта информационной системы.
4. Понятие технологии. Особенности технологии программирования.
5. Основные понятия проектирования информационных систем .
6. Классификация методов проектирования информационных систем.
7. Параметрически-ориентированное проектирование информационных систем.
8. Технологическая сеть модельно-ориентированного проектирования информационных систем.
9. Прототипное проектирование информационных систем.
10. Типовое проектирование информационных систем. Виды типового проектирования.
11. Методы типового проектирования информационных систем: элементный, подсистемный, объектный.
12. Стадии и этапы канонического проектирования ИС.
13. Автоматизированное проектирование информационных систем. Стадии и этапы процесса проектирования ИС с применением CASE–технологии.
14. Определение критериев оценки функционального пакета прикладных программ.
15. Требования к программной системе.
16. Оценка работоспособности информационных систем.
17. Методики обеспечения корректности: тестирование и отладка.
18. Особенности и методы коллективной разработки проектов.
19. Контроль версий: назначение, классификация, принципы работы.
20. Инструменты проектирования программных систем.
21. Визуальное проектирование, язык UML: назначение и общие принципы.
22. Базовые принципы разработки и реализации диалоговых систем.
23. Документирование программных систем.
24. Дополнительные вопросы
25. Модель функций.
26. Модель процессов.
27. Модели объектов (данных).

28.Модель организационной структуры.
29.Модели бизнес-правил.

IV. КОНТРОЛЬ ДОСТИЖЕНИЯ ЦЕЛЕЙ КУРСА

Изучение дисциплины «Объектно-ориентированный анализ и проектирование» предусматривает:

- предоставление теоретического материала в соответствии с программой, с использованием электронного раздаточного материала;
- предоставление теоретического материала посредством электронных обучающих программ.
- выполнение домашних заданий;
- выполнение индивидуальных заданий;
- обязательная проработка материала, который будет разбираться на занятии с подбором дополнительных материалов.

Текущий контроль. Предусматривает учет посещения студентами занятий в течение периода обучения и оценку своевременности и качества выполнения студентами тестов и домашних заданий.

Итоговый контроль. Предусматривает рейтинговую оценку по учебной дисциплине в течение семестра и зачет.

№ п/п	Контролируемые разделы / темы дисциплины	Коды и этапы формирования компетенций	Оценочные средства - наименование	
			текущий контроль	промежуточная аттестация
1.	Технологии программирования как частный случай технологий	ПК-7 знает понятия и назначение технологии, в частности технологии программирования; историю развития технологий программирования; конкретные технологии разработки, в том числе формальные методы. умеет оценивать трудоёмкость и планировать процесс разработки программного продукта средней сложности владеет практическим опытом разработки нетривиального программного продукта	Коллоквиум (УО-2), контрольная работа (ПР-2) 1 - 11	

2.	Классификация и организация коллективной деятельности	ПК-7 знает понятие интеллектуальной собственности, юридические и этические аспекты разработки программного обеспечения; проблемы и методы организации работы творческих коллективов, в особенности программистских умеет взаимодействовать с другими разработчиками в составе коллектива владеет навыками разделения труда в составе творческого коллектива	Лабораторная работа (ПР-6) Отчеты по лабораторным работам 1, 2, 4
3.	Жизненный цикл программного продукта	ПК-7 знает о целях и способах разработки программной системы; структуру жизненного цикла программного продукта; о целях и способах отладки программной системы; о целях и способах внедрения и сопровождения программной системы; умеет участвовать во всех этапах жизненного цикла программного продукта на любой роли владеет навыками применения технических и организационных средств поддержки разработки	Лабораторная работа (ПР-6) Отчеты по лабораторным работам 2, 3, 5
4.	Системный подход и системный анализ	ПК-7 знает целях и способах анализа предметной области; объектно-ориентированном и функциональном анализе, Agile-методологиях и др. умеет проводить анализ предметной области,	Коллоквиум (УО-2), Контрольная работа (ПР-2) 22 - 25

		взаимодействовать с экспертами в предметной области для постановки задачи; владеет навыками взаимодействия с экспертами в предметной области	
5.	Методология проектирования программных продуктов	ПК-7 знает способы проектирования, внедрения и сопровождения программной системы умеет выбирать и использовать инструменты поддержки разработки программных продуктов владеет методологиями проектирования программных продуктов; навыками внедрения и сопровождения программной системы	Лабораторная работа (ПР-6) Отчеты по лабораторным работам 1, 2, 4

;
;
;

Типовые контрольные задания, методические материалы, определяющие процедуры оценивания знаний, умений и навыков и (или) опыта деятельности, а также критерии и показатели, необходимые для оценки знаний, умений, навыков и характеризующие этапы формирования компетенций в процессе освоения образовательной программы, представлены в Приложении 2.

V. СПИСОК УЧЕБНОЙ ЛИТЕРАТУРЫ И ИНФОРМАЦИОННО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Основная литература
(электронные и печатные издания)

1. Beginning Object-Oriented Analysis and Design with C+ [Электронный ресурс]: учебное пособие/ Liberty, J.– Chicago : Wrox Press Ltd, 2020. https://lib.dvfu.ru/lib/item?id=chamo:321974&theme=FEFU

2. Применение UML и шаблонов проектирования: введение в объектно-ориентированный анализ, проектирование и унифицированный процесс UP [Электронный ресурс]: учебное пособие/ К. Ларман 2020. https://lib.dvfu.ru/lib/item?id=chamo:395155&theme=FEFU
3. Леоненков А.В. Объектно-ориентированный анализ и проектирование с использованием UML и IBM Rational Rose. Курс лекций [Электронный ресурс]: учебное пособие для студентов вузов, обучающихся по специальностям в области информационных технологий/ Леоненков А.В.— Электрон. текстовые данные.— Москва, Саратов: Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2017.— 318 c.— Режим доступа: http://www.iprbookshop.ru/67388.html.— ЭБС «IPRbooks»
4. Алгоритмизация и программирование: Учебное пособие / С.А. Канцедал. - М.: ИД ФОРУМ: НИЦ ИНФРА-М, 2013. - 352 с.: ил.; 60x90 1/16. - (Профессиональное образование). (переплет) ISBN 978-5-8199-0355-1
5. Структурный анализ и проектирование информационных систем : учебное пособие / В. В. Ивин ; Дальневосточный федеральный университет, Школа экономики и менеджмента. - Владивосток : Изд-во Дальневосточного федерального университета, 2013. - 182 с.: И 253 004.41(075.8) ДВФУ.
6. Сысолетин Е.Г. Проектирование интернет-приложений [Электронный ресурс]: учебно-методическое пособие/ Сысолетин Е.Г., Ростунцев С.Д.— Электрон. текстовые данные.— Екатеринбург: Уральский федеральный университет, ЭБС АСВ, 2015.— 92 c.— Режим доступа: http://www.iprbookshop.ru/66582.html.— ЭБС «IPRbooks»

Дополнительная литература
(печатные и электронные издания)

1. Митина О.А. Прикладное программирование [Электронный ресурс]: учебное пособие/ Митина О.А.— Электрон. текстовые данные.— М.: Московская государственная академия водного транспорта, 2017.— 94 c.— Режим доступа: http://www.iprbookshop.ru/76716.html.— ЭБС «IPRbooks»
2. Комлев Н.Ю. Полезное программирование [Электронный ресурс]/ Комлев Н.Ю.— Электрон. текстовые данные.— М.: СОЛОН-ПРЕСС, 2016.— 256 c.— Режим доступа: http://www.iprbookshop.ru/53837.html.— ЭБС «IPRbooks»

Перечень ресурсов информационно-телекоммуникационной сети «Интернет»

1. Тузовский А.Ф. Проектирование и разработка web-приложений [Электронный ресурс]: учебное пособие/ Тузовский А.Ф.— Электрон.

текстовые данные.— Томск: Томский политехнический университет, 2014.— 219 с.— Режим доступа: http://www.iprbookshop.ru/34702.html.— ЭБС «IPRbooks»
2. 1С: Предприятие. Проектирование приложений: Учебное пособие / Дадян Э.Г. - М.:Вузовский учебник, НИЦ ИНФРА-М, 2015. - 288 с.: 60x90 1/16 (Переплёт 7БЦ) ISBN 978-5-9558-0394-4 - Режим доступа: http://znanium.com/catalog/product/480629

Перечень информационных технологий и программного обеспечения

1. Электронная презентация
2. Электронная рассылка
3. Электронный журнал успеваемости
4. ЭУК «Информатика и программирование» в интегрированной платформе электронного обучения Blackboard ДВФУ.
5. Электронные поисковые приложения.

VI. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

На изучение дисциплины отводится 72 часов аудиторных (лекционных и лабораторных) занятий. На занятиях перед выдачей индивидуальных заданий преподаватель объясняет теоретический материал по заданной теме. Вводит основные требования к его выполнению. Приводит примеры. Поддерживает непрерывный контакт с аудиторией, отвечает на возникающие у студентов вопросы. На лабораторных занятиях преподаватель разбирает принципы и аспекты реализации задания по заданной теме.

Во второй части занятия студентам предлагается работать самостоятельно, выполняя задания по теме. Преподаватель контролирует самостоятельную работу студентов, на консультациях отвечает на возникающие вопросы, подсказывает ход и метод решения. Если знаний полученных в аудитории оказалось недостаточно, студент может самостоятельно повторно просмотреть презентацию, просмотреть практикум с разобранными примерами, которые собраны в изучаемом курсе в системе Teams.

После выполнения задания, студент отправляет его на проверку преподавателю по электронной почте, либо предъявляет на компьютере во время занятия. Работа должна быть отослана либо одним документом – исходным файлом на языке программирования, либо архивом.

По данному курсу разработаны методические указания, доступные по указанному электронному адресу. Для успешного достижения учебных целей занятий должны выполняться следующие основные требования:

-соответствие действий обучающихся ранее изученным на лекционных и семинарских занятиях методикам и методам.

-максимальное приближение действий студентов к реальным, соответствующим будущим функциональным обязанностям.

-поэтапное формирование умений и навыков, т.е. движение от знаний к умениям и навыкам, от простого к сложному и т.д..

-использование при работе на тренажерах или действующей технике фактических документов, технологических карт, бланков и т.п.

-выработка соответствующих индивидуальных и коллективных умений и навыков.

-распределение времени, отведенного на занятие, на решение каждой задачи;

-подбор иллюстративного материала (графиков, таблиц, схем), необходимого для решения задач, продумывание расположения рисунков и записей на доске или экране.

Студент должен:

-научиться работать с книгой, документацией и схемами, пользоваться справочной и научной литературой.

-научиться работать с электронными литературными источниками.

-формировать умение учиться самостоятельно, т.е. овладевать методами, способами и приемами самообучения, саморазвития и самоконтроля.

Рекомендации по подготовке к зачету:

При ответе на каждый вопрос зачета студент должен продемонстрировать знание определения указанного понятия, связанных с ним особенностей реализации и применения, умение реализовать указанный алгоритм / структуру данных, а также навыки иллюстрации теоретических принципов на предложенных простых примерах.

VII. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

1. Компьютерные классы ДВФУ (кампус на о. Русском, Аякс 10)
2. Системное и прикладное обеспечение ПЭВМ.
3. Мультимедийные технические средства.

ФОНДЫ ОЦЕНОЧНЫХ СРЕДСТВ

Шкала оценивания уровня сформированности компетенций

ПК-7 Способность изготавливать компоненты информационных систем, включая программные комплексы, базы данных и интерфейсы "человек - электронно-вычислительная машина", использовать современные инструментальные средства разработки, и программно-технологические платформы информационных систем

Планируемые результаты обучения* (показатели достижения заданного уровня освоения компетенций)	**Критерии оценивания результатов обучения****				
	1	2	3	4	5
Знает: базовые методы технологии программирования, математические методы формализации задачи, требования к разработке программных приложений.	Отсутствие знания базовых методов технологии программирования, математических методов формализации задачи, требования к разработке программных приложений.	Фрагментарное знание знания базовых методов технологии программирования, математических методов формализации задачи, требования к разработке программных приложений,	Неполное знание знания базовых методов технологии программирования, математических методов формализации задачи, требования к разработке программных приложений,	В целом сформировавшееся знание знания базовых методов технологии программирования, математических методов формализации задачи, требования к разработке программных приложений.	Сформировавшееся систематическое знание знания базовых методов технологии программирования, математических методов формализации задачи, требования к разработке программных приложений
Умеет: реализовывать программные приложения малой сложности, создавать программные прототипы решения прикладных задач.	Отсутствие умения реализовывать программные приложения малой сложности, создавать программные прототипы решения прикладных задач.	Фрагментарное умение реализовывать программные приложения малой сложности, создавать программные прототипы решения прикладных задач.	Неполное умение реализовывать программные приложения малой сложности, создавать программные прототипы решения прикладных задач.	В целом сформировавшееся умение реализовывать программные приложения малой сложности, создавать программные прототипы решения	Сформировавшееся систематическое умение реализовывать программные приложения малой сложности, создавать программные прототипы решения

				прикладных задач**.**	прикладных задач**.**
Владеет: навыками разработки, тестирования и отладки программных приложений.	Отсутствие владения навыками разработки, тестирования и отладки программных приложений.	Фрагментарное владение навыками разработки, тестирования и отладки программных приложений.	Неполное владение навыками разработки, тестирования и отладки программных приложений.	В целом сформировавшееся владение навыками разработки, тестирования и отладки программных приложений кодирования.	Сформировавшееся систематическое владение навыками разработки, тестирования и отладки программных приложений кодирования.
Шкала оценивания (соотношение с традиционными формами аттестации)	0–8 неудовлетворительно	9–12 неудовлетворительно	13–15 удовлетворительно	16–18 хорошо	19–20 отлично

Оценочные средства для промежуточной аттестации

БИЛЕТЫ К ЭКЗАМЕНУ

Вариант №1

1. Дать определение понятия "сигнал". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "состояние носителя информации" от объектов понятия "сигнал".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: записанная в журнале оценка "2". Доказать справедливость ответа.
3. Определить длину информационного сообщения, содержащего ваши фамилию, имя и отчество (с пробелами) после кодирования его символами: & $ %.

Вариант №2

1. Дать определение понятия "носитель информации". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "состояние носителя информации" от объектов понятия "параметр носителя информации".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: включеная лампочка. Доказать справедливость ответа.

3. Определить минимальную мощность набора сигналов, достаточную для построения сообщения, содержащего ваши фамилию, имя и отчество (с пробелами).

Вариант №3
1. Дать определение понятия "состояние носителя информации". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "состояние носителя информации" от объектов понятия "носитель информации".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: коробка с акварельными красками (из магазина). Доказать справедливость ответа.
3. Определить исходную длину информационного сообщения, состоящего из латинских заглавных и малых букв, если его длина после кодирования символами: & $ % стала равна 60.

Вариант №4
1. Дать определение понятия "интерпретация сигнала". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "код сигнала" от объектов понятия "интерпретация сигнала".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: десятиминутная барабанная дробь. Доказать справедливость ответа.
3. Определить как изменится длина информационного сообщения, состоящего из латинских заглавных и малых букв и трех знаков препинания, после кодирования его символами: & $ % #.

Вариант №5
1. Дать определение понятия "таблица кодировки". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "таблица кодировки" от объектов понятия "алфавит".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: на циферблате электронных часов высвечивается "13:08". Доказать справедливость ответа.
3. Определить длину графического информационного сообщения, созданного с помощью цветовой гаммы: белый, черный, синий, красный, зеленый в двоичной к кодировке. Разрешение выбрать самостоятельно.

Вариант №6
1. Дать определение понятия "код сигнала". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "код сигнала" от объектов понятия "сигнал".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: письменная запись "1234567890". Доказать справедливость ответа.

3. Составит таблицу кодировки для кодирования буквами: А У гафического сообщения, созданного с помощью цветовой гаммы: желтый, черный, синий, красный, зеленый, оранжевый, белый, голубой, коричневый .

Вариант №7

1. Дать определение понятия "разрешение". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "состояние носителя информации" от объектов понятия "носитель информации".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: клавиша рояля . Доказать справедливость ответа.
3. Определить длину кода для кодирования информационного сообщения, состоящего из латинских и русских заглавных и малых букв, символами: & $ %

Вариант №8

1. Дать определение понятия "дискретизация". Привести три примера с доказательством. Сформулировать утверждение о соотношении понятий "сигнал" и "дискретизация".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: горящий факел. Доказать справедливость ответа.
3. Определить как изменится длина информационного сообщения, состоящего из латинских заглавных и малых букв и трех знаков препинания, после кодирования его символами: & $ % #.

Вариант №9

1. Дать определение понятия "мощность набора сигналов". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "мощность набора сигналов" от объектов понятия "основание системы счисления".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: пароль (при авторизации). Доказать справедливость ответа.
3. Определить мощность конечного набора, при которой длина информационного сообщения, состоящего из латинских заглавных и малых букв, после кодирования увеличится втрое.

Вариант №10

1. Дать определение понятия "интерпретация сигнала". Привести три примера с доказательством. Сформулировать утверждение о соотношении понятий " символ" и "сигнал".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: положение жезла полицейского. Доказать справедливость ответа.
3. Информационное сообщение состоит из русских и английских малых букв и пробелов. Определить, сколько символов должно быть в конечном наборе, чтобы длина сообщения увеличилась в 2 раза.

Вариант №11

1. Дать определение понятия "сигнал". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "состояние носителя информации" от объектов понятия "сигнал".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: записанная в журнале оценка "2". Доказать справедливость ответа.
3. Определить длину информационного сообщения, содержаего ваши фамилию, имя и отчество (с пробелами) после кодирования его символами: & $ %.

Вариант №12
1. Дать определение понятия "носитель информации". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "состояние носителя информации" от объектов понятия "параметр носителя информации".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: включеная лампочка. Доказать справедливость ответа.
3. 3. Определить минимальную мощность набора сигналов, достаточную для построения сообщения, содержащего ваши фамилию, имя и отчество (с пробелами).

Вариант №13
1. Дать определение понятия "состояние носителя информации". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "состояние носителя информации" от объектов понятия "носитель информации".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: коробка с акварельными красками (из магазина). Доказать справедливость ответа.
3. Определить исходную длину информационного сообщения, состоящего из латинских заглавных и малых букв, если после кодирования его символами: & $ % его длина стала равна 60.

Вариант №14
1. Дать определение понятия "интерпретация сигнала". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "код сигнала" от объектов понятия "интерпретация сигнала".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: десятиминутная барабанная дробь. Доказать справедливость ответа.
3. Определить как изменится длина информационного сообщения, состоящего из латинских заглавных и малых букв и трех знаков препинания, после кодирования его символами: & $ % #.

Вариант №15
1. Дать определение понятия "таблица кодировки". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "таблица кодировки" от объектов понятия "алфавит".

2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: на циферблате электронных часов высвечивается "13:08". Доказать справедливость ответа.
3. Определить длину графического информационного сообщения, созданного с помощью цветовой гаммы: белый, черный, синий, красный, зеленый в двоичной кодировке. Разрешение выбрать самостоятельно.

Вариант №16
1. Дать определение понятия "код сигнала". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "код сигнала" от объектов понятия "сигнал".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: письменная запись "1234567890". Доказать справедливость ответа.
3. Составит таблицу кодировки для кодирования гафического сообщения, созданного с помощью цветовой гаммы: желтый, черный, синий, красный, зеленый, оранжевый, белый, голубой, коричневый буквами: А У.

Вариант №17
1. Дать определение понятия "разрешение". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "состояние носителя информации" от объектов понятия "носитель информации".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: клавиша рояля . Доказать справедливость ответа.
3. Определить длину кода для кодирования информационного сообщения, состоящего из латинских и русских заглавных и малых букв, символами: & $ %

Вариант №18
1. Дать определение понятия "дискретизация". Привести три примера с доказательством. Сформулировать утверждение о соотношении понятий "сигнала" и "дискретизация".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: горящий факел. Доказать справедливость ответа.
3. Определить как изменится длина информационного сообщения, состоящего из латинских заглавных и малых букв и трех знаков препинания, после кодирования его символами: & $ % #.

Вариант №19
1. Дать определение понятия "мощность набора сигналов". Привести три примера с доказательством. Указать основной признак, отличающий объекты понятия "мощность набора сигналов" от объектов понятия "основание системы счисления".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: пароль (при авторизации). Доказать справедливость ответа.

3. Определить мощность конечного набора, при которой длина информационного сообщения, состоящего из латинских заглавных и малых букв, после кодирования увеличится втрое.

Вариант №20
1. Дать определение понятия "интерпретация сигнала". Привести три примера с доказательством. Сформулировать утверждение о соотношении понятий " символ" и "сигнал".
2. Определить понятие(я) информатики, которому(ым) соответствует следующий пример: положение жезла полицейского. Доказать справедливость ответа.
3. Информационное сообщение состоит из русских и английских малых букв и пробелов. Определить, сколько символов должно быть в конечном наборе, чтобы длина сообщения увеличилась в 2 раза.

Оценочные средства для текущей аттестации

Темы для подготовки к коллоквиуму (теория)

- Значение информации в развитии современного информационного общества.
- Тенденции и проблемы развития современного информационного общества.
- Основные понятия информатики.
- Виды и формы информации. Информационные процессы.
- Технические и программные средства реализации информационных процессов.
- Принципы кодирования информации.
- Машиночитаемые носители информации. Правила кодирования.
- Определение информационного объема.
- Опасности и угрозы информационного общества.
- Основы защиты информации
- Структура программного обеспечения.
- Типы и структуры данных.
- Потоки управления в программах.
- Базовые алгоритмы обработки информации. Оценка сложности и эффективности алгоритмов.
- Структура программ.
- Стиль программирования.
- Этапы создания программ.
- Статическое распределение памяти.
- Динамическое распределение памяти.
- Динамические структуры данных: стек, очередь, список, дерево.
- Языки программирования высокого уровня.
- Базовые типы данных.

- Составные типы данных: массив, строка, запись.
- Целочисленная арифметика.
- Организация и средства человеко-машинного интерфейса.
- Основные алгоритмические конструкции.
- Базовые алгоритмы.
- Подпрограммы: процедуры, функции.
- Двумерные массивы.
- Алгоритмы сортировки.
- Организация работы с файлами
- Метод рекурсии.
- Алгоритмы поиска.
- Метод перебора.
- Технология тестирования.
- Синтаксис и семантика языковых конструкций.