Приложение 3.РПД Б1.В.ОД.6

Филиал федерального государственного бюджетного образовательного учреждения высшего образования
«Национальный исследовательский университет «МЭИ»
в г. Смоленске

УТВЕРЖДАЮ
Зам. директора
филиала ФГБОУ ВО «НИУ «МЭИ»
в г. Смоленске
по учебно-методической работе
_____ В.В. Рожков
«___» _____ 2016 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ

(НАИМЕНОВАНИЕ ДИСЦИПЛИНЫ)

Направление подготовки: 09.03.01 Информатика и вычислительная техника

Профиль подготовки: Вычислительные машины,комплексы, системы и сети

Уровень высшего образования: бакалавриат

Нормативный срок обучения: 5 лет

Форма обучения: заочная

Смоленск – 2016 г.

1 Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения образовательной программы

Целью освоения дисциплины является подготовка обучающихся к научно-исследовательской, проектной деятельности по направлению подготовки 09.03.01 "Информатика и вычислительная техника" посредством обеспечения этапов формирования компетенций, предусмотренных ФГОС, в части представленных ниже знаний, умений и навыков.

Задачами дисциплины являются:

- познакомить обучающихся с основными понятиями и определениями, с классификацией программного обеспечения;
- дать представление об этапах создания программного продукта в рамках жизненного цикла, о современном состоянии технологий разработки программного продукта;
- познакомить обучающихся с существующими подходами к оценка качества процессов создания программного обеспечения, ;
- дать обучающемуся практические навыки проектирования программного обеспечения и расчета его надежности.

Дисциплина направлена на формирование следующих общекультурных и профессиональных компетенций:

- ОПК-1. Способностью инсталлировать программное и аппаратное обеспечение для информационных и автоматизированных систем
- ОПК-2. Способностью осваивать методики использования программных средств для решения практических задач
- ПК-1. способностью разрабатывать модели компонентов информационных систем, включая модели баз данных и модели интерфейсов "человек-электронно-вычислительная машина"
- ПК-2. Способностью разрабатывать компоненты аппаратно-программных комплексов и баз данных, используя современные инструментальные средства и технологии программирования

В результате изучения дисциплины обучаемый должен:

знать:

- принципы проектирования программных систем (ОПК-1);
- организацию процесса проектирования программного обеспечения (ОПК-2);
- методологию структурного проектирования ПО (ОПК-1, ОПК-2);
- методологию объектно-ориентированного проектирования ПО (ОПК-1, ОПК-2);
- технологические средства разработки программного обеспечения (ПК-1);
- методы декомпозиции и абстракции при проектировании ПО (ПК-1);
- методы отладки и тестирования программ (ПК-2);
- методы защиты программ и данных (ПК-2).

уметь:

- использовать методы декомпозиции и абстракции при проектировании ПО (ОПК-1);
- применять средства разработки программного обеспечения: инструментальные среды разработки, средства поддержки проекта, отладчики (ОПК-2);
- документировать и оценивать качество программных продуктов (ПК-1);
- проектировать пользовательские интерфейсы (ПК-2).

владеть:

- методами и средствами разработки и оформления технической документации (ОПК-2);
- методами проектирования программного обеспечения при структурном и объектно-ориентированном подходе (ОПК-1, ПК-1);

- методами структурного и функционального тестирования (ПК-2)
- методами совместной разработки приложений (ОПК-1, ОПК-2, ПК-1, ПК-2).

2. Место дисциплины в структуре образовательной программы

Дисциплина «Технология программирования» относится к обязательным дисциплинам вариативной части профессионального цикла Б1.В.ОД.6 основной образовательной программы подготовки бакалавров по направлению «09.03.01 «Информатика и вычислительная техника».

В соответствии с учебным планом по направлению "Информатика и вычислительная техника" дисциплина «Технология программирования» базируется на следующих дисциплинах:

• Б1.Б.8	Информатика
• Б1.Б.9	Инженерная графика
• Б1.В.ОД.3	Теория алгоритмов
• Б1.В.ОД.2	Дискретная математика
• Б1.Б.6	Вычислительная математика
• Б1.Б.7	Теория вероятностей и математическая статистика
• Б2.У.1	Практика по получению первичных профессиональных умений и навыков, в том числе первичных умений и навыков научно-исследовательской деятельности
• Б2.У.2	Исполнительская практика
• Б1.Б.11	Базы данных
• Б1.В.ОД.1	Программирование
• Б1.В.ДВ.3.1	Введение в оптимизацию
• Б1.В.ДВ.3.2	Теория систем
• Б1.В.ОД.12	Теория автоматов
• Б1.В.ОД.13	Основы теории управления
• Б2.П.1	Практика по получению профессиональных умений и опыта профессиональной деятельности
• Б1.Б.14.2	Схемотехника
• Б1.В.ДВ.4.1	Основы логического программирования
• Б1.В.ДВ.4.2	Кластерные вычислительные системы

Знания, умения и навыки, полученные студентами в процессе изучения дисциплины «Технология программирования» необходимы для формирования компетенций в дисциплинах:

• Б1.В.ОД.10	Защита информации
• Б1.В.ОД.8	Сетевые технологии
• Б1.В.ОД.11	Моделирование
• Б1.В.ОД.15	Сопровождение разработки программного обеспечения
• Б1.В.ОД.17	Инженерное проектирование и САПР
• Б1.В.ДВ.5.1	Прикладная статистика
• Б1.В.ДВ.5.2	Методы анализа данных
• Б1.В.ДВ.6.1	Аппаратная реализация алгоритмов
• Б1.В.ДВ.6.2	Технология проектирования устройств на ПЛИС
• Б1.В.ДВ.11.1	Интернет-технологии
• Б1.В.ДВ.11.2	Проектирование WEB-приложений
• Б2.П.3	Технологическая практика

- Б1.В.ОД.4 Операционные системы
- Б1.В.ОД.14 Тестирование программного обеспечения
- Б1.В.ОД.16 Конструирование и технологии средств вычислительной техники
- Б1.В.ОД.9 Микропроцессорные системы
- Б1.В.ДВ.7.1 Теория передачи информации
- Б1.В.ДВ.7.2 Методы и средства цифровой связи
- Б1.В.ДВ.8.1 Основы теории надежности
- Б1.В.ДВ.8.2 Надежность и диагностика технических средств
- Б1.В.ДВ.9.1 Проектирование информационных систем
- Б1.В.ДВ.9.2 Информационные технологии
- Б1.В.ДВ.10.1 Корпоративные и ведомственные сети
- Б1.В.ДВ.10.2 Технологические сети для сбора данных и управления
- Б1.В.ДВ.4.1 Введение в цифровую обработку сигналов
- Б1.В.ДВ.4.2 Теория сигналов
- Б2.П.4 Преддипломная практика
- Б3 Государственная итоговая аттестация

3 Объем дисциплины в зачетных единицах с указанием количества академических часов, выделенных на контактную работу обучающихся с преподавателем (по видам учебных занятий) и на самостоятельную работу обучающихся)

Общая трудоемкость дисциплины составляет 5 зачетных единиц, 180 часов.

Аудиторная работа

Цикл:	Б1	Курс
Часть цикла:	Вариативная Обязательная дисциплина	
Индекс дисциплины по учебному плану	Б1.В.ОД.6	
Часов всего по учебному плану	180	4
Трудоемкость в зачетных единицах (ЗЕТ)	5	4
Лекции (ЗЕТ/ часов)	0,22/8	4
Лабораторные работы (ЗЕТ/ часов)	0,33/12	4
Курсовое проектирование	0,12/4	4
Объем самостоятельной работы по учебному плану (ЗЕТ/ часов всего)	4,08/147	4
Экзамен	0,25/9	4

Самостоятельная работа студента

Вид работ	Трудоёмкость	
	ЗЕТ	*час*
Подготовка к сеансу тестирования	0,11	4
Подготовка к контрольной работе	0,44	16
Подготовка к выполнению и защите лабораторных работ (лаб)	0,56	20
Изучение дополнительного теоретического материала	1,06	38
Подготовка к лекции	0,41	15
Курсовая работа	1,5	54
Всего:	**4,08**	**147**

4 Содержание дисциплины, структурированное по темам с указанием отведенного на них количества академических часов и видов учебных занятий

№ п/п	Темы дисциплины	Общая трудоемкость, всего	Виды учебной работы, включая самостоятельную работу студентов и трудоемкость (в часах)								
			Аудиторные занятия				Экзамен	Самостоятельная работа			Занятия в интерактивной форме
			Всего	Лекции	Лабораторные работы	Курсовое проектирование		Всего	Курсовая работа/проект	Другая	
1	Основные понятия и подходы. Приемы обеспечения технологичности программных продуктов. Разработка технического задания	33	4	1	2	1		29	10	19	
2	Проектирование программного обеспечения при структурном подходе к программированию	31	3	1	2			28	11	17	
3	Тестирование и отладка программных продуктов при структурном подходе к программированию	35	5	2	2	1		30	11	19	
4	Проектирование программного обеспечения при объектно-ориентированном подходе к программированию	35	5	2	2	1		30	11	19	
5	Разработка пользовательских интерфейсов. Оценка качества программного обеспечения	37	7	2	4	1		30	11	19	
Экзамен		9					9				
Всего		**180**	**24**	**8**	**12**	**4**	**9**	**147**	**54**	**93**	

Тема 1. Основные понятия и подходы. Приемы обеспечения технологичности программных продуктов. Разработка технического задания

Лекция 1 (1 час)

Технология программирования и основные этапы ее развития. Проблемы разработки сложных программных систем. Блочно-иерархический подход к созданию сложных систем. Жизненный цикл и этапы разработки программного обеспечения. Эволюция моделей жизненного цикла программного обеспечения. Ускорение разработки программного обеспечения. Технология RAD. Оценка качества процессов создания программного обеспечения.

Понятие технологичности программного обеспечения. Модули и их свойства. Нисходящая и восходящая разработка программного обеспечения. Структурное и «неструктурное» программирование. Средства описания структурных алгоритмов (псевдокоды, схемы алгоритмов, Flow-формы, диаграммы Насси-Шнейдермана). Стиль оформления программы. Эффективность и технологичность. Программирование «с защитой от ошибок». Сквозной структурный контроль.

Основные эксплуатационные требования к программным продуктам. Предпроектные исследования предметной области. Разработка технического задания. Принципиальные решения начальных этапов проектирования. Классификация моделей разрабатываемого программного обеспечения.

Лабораторная работа 1 (2 часа)

Линейные программы, Циклические программы, Массивы. Оценка качества разрабатываемого программного обеспечения.

Цель работы:

- Изучение основ языка C# и знакомство с элементами управления C#. Составление линейных и циклических программ. Работа с математическими функциями C#.
- Изучить особенности работы с массивами в языке C#, свойства и методы класса System.Array. Решение задач с одномерными массивами.
- Оценка характеристик разработанных программ с помощью метрик Холстеда; метрик Джилба. Оценка надежности программных средств с помощью модели Джелински – Моранды, эвристической модели и модели Нельсона.

Самостоятельная работа

Тема учебной дисциплины	Вид самостоятельной работы студента (реферат, расчетно-графическая работа, др.)	Всего часов
Тема 1. Основные понятия и подходы. Приемы обеспечения технологичности программных продуктов. Разработка технического задания	Изучение дополнительного теоретического материала	8
	Подготовка к лекции	3
	Оформление и подготовка к защите лабораторной работы	4
	Подготовка к контрольной работе	4
	Курсовая работа	10
	ИТОГО:	29

Текущий контроль – устные опросы по лекции 1 и самостоятельно изученному материалу, контрольная работа по теме 1.

Курсовая работа (12 часов)

Студенты выбирают тему курсовой работы, изучают литературные источники и составляют техническое задание на курсовую работу. Цель курсовой работы освоить программирование в среде C# - работу с циклами, строками, унарные и бинарные логические операции, операции сдвига и т.д.

Контрольная работа
Цель контрольной работы – закрепление следующих вопросов: технологичность программного обеспечения, модули и их свойства, нисходящая и восходящая разработка программного обеспечения, структурное и «неструктурное» программирование. Средства описания структурных алгоритмов, стиль оформления программы, эффективность и технологичность.
Коды формируемых компетенций: ОПК-1, ПК-1, ПК-2
Результаты освоения:
ОПК-1: Освоение среды программирования Visual Studio – создание проектов, тестирование, отладка.
ПК-1: освоение методики разработки технического задания, методов предпроектных исследований, осознанный выбор среды программирования
ПК-2: Разработка технологичных многомодульных программных продуктов с учетом сцепления и связности модулей, освоение методов проектирования структурных алгоритмов.

Тема 2. Проектирование программного обеспечения при структурном подходе к программированию

Лекция 2 (1 час)
Анализ требований и определение спецификаций при структурном подходе. Спецификации процессов. Словарь терминов. Диаграммы переходов состояний (SDT-диаграммы). Функциональные диаграммы (IDEF0). Диаграммы потоков данных (DFD).
Диаграммы отношений компонентов данных: диаграммы Джексона и скобочные диаграммы Орра, сетевая модель данных (диаграмма «сущность-связь»).
Структурная схема разрабатываемого программного обеспечения. Функциональная схема. Метод пошаговой детализации при составлении алгоритмов. Структурные карты Константайна. Структурные карты Джексона.
Лабораторная работа 2 (2 часа)
Работа со строками в C#. Оценка степени отлаженности разрабатываемых приложений.
Цель работы: Освоить работу со строками в C#. Изучить методы классов String и StringBuilder. Оценить степень отлаженности разрабатываемых приложений с помощью модели Миллса. Разработать структурную и функциональную схему разрабатываемого программного обеспечения.
Самостоятельная работа

Тема учебной дисциплины	Вид самостоятельной работы студента (реферат, расчетно-графическая работа, др.)	Всего часов
Тема 2. Проектирование программного обеспечения при структурном подходе к программированию	Подготовка к лекции	3
	Оформление и подготовка к защите лабораторной работы	4
	Подготовка к сеансу тестирования	4
	Изучение дополнительного теоретического материала	6
	Курсовая работа	11
	ИТОГО:	28

Текущий контроль – устные опросы по лекции 2 и самостоятельно изученному материалу, тестирование.
Тестирование
Тестирование является итоговым заданием по основам языка программирования C#. Тест содержит вопросы по следующим темам: особенности работы с переменными различных

типов, арифметические, логические операции языка программирования C#, поразрядные операции, особенности работы с математическими функциями в C# (библиотека Math), операторы ветвления (if) и выбора (switch), операторы циклов (for, foreach, while и do…while).

Курсовая работа (12 часов)

Студенты выполняют проектирование курсовой работы в соответствии с выбранным подходом к проектированию.

Коды формируемых компетенций: ОПК-2

Результаты освоения:

ОПК-2: Осваивать методики использования программных средств для решения практических задач. Освоить методику структурного проектирования SADT, разработку диаграмм переходов состояний, структурных и функциональных схем, методику Джексона, Структурные карты Константайна.

Тема 3. Тестирование и отладка программных продуктов при структурном подходе

Лекция 3 (2 часа)

Виды контроля качества разрабатываемого программного обеспечения. Ручной контроль программного обеспечения. Структурное тестирование программного обеспечения. Особенности структурного тестирования. Способ тестирования базового пути. Потоковый граф. Цикломатическая сложность. Шаги способа тестирования базового пути. Способы тестирования условий. Тестирование ветвей и операторов отношений. Способ тестирования потоков данных. Тестирование циклов. Простые циклы. Вложенные циклы. Объединенные циклы. Неструктурированные циклы.

Функциональное тестирование программного обеспечения. Особенности функционального тестирования. Разбиение на классы эквивалентности. Анализ граничных значений. Диаграммы причинно-следственных связей.

Отладка программного обеспечения. Классификация ошибок. Методы отладки программного обеспечения. Методы и средства получения дополнительной информации. Общая методика отладки программного обеспечения.

Лабораторная работа 3 (2 часа)

Тестирование программного обеспечения при структурном подходе к программированию.

Цель работы:

- Изучение методов тестирования и отладки программного обеспечения.
- Выполнение ручного тестирования программ.
- Выполнение структурного тестирования программ следующими методами: тестирование базового пути; тестирование условий; тестирование циклов; тестирование потоков данных.
- Выполнение функционального тестирования программ следующими методами: разбиение на классы эквивалентности; анализ граничных значений; анализ причинно-следственных связей; предположение об ошибке.
- Выполнение отладки программного обеспечения методами индукции и методами дедукции. Сравнительная характеристика данных методов отладки программного обеспечения.

Самостоятельная работа

Тема учебной дисциплины	Вид самостоятельной работы студента (реферат, расчетно-графическая работа, др.)	Всего часов
Тема 3. Тестирование и отладка программных продуктов при структурном подходе	Оформление и подготовка к защите лабораторных работ	4
	Подготовка к лекции	3
	Изучение дополнительного теоретического материала	8
	Подготовка к контрольной работе	4
	Курсовая работа	11
	ИТОГО:	30

Текущий контроль – устные опросы по лекции 3 и самостоятельно изученному материалу, выполнение контрольной работы.

Контрольная работа

Цель контрольной работы – закрепить материал лекции 3, а именно, методы тестирования и отладки программного обеспечения при структурном подходе. Определение ситуаций, когда следует применять различные методики тестирования.

Курсовая работа (12 часов)

Студенты приступают к программной реализации курсовой работы

Коды формируемых компетенций: ПК-2

Результаты освоения:

ПК-2: Освоить методы ручного, структурного и функционального тестирования.

Тема 4. Проектирование программного обеспечения при объектно-ориентированном подходе к программированию

Лекция 4 (2 часа)

UML- стандартный язык описания разработки программных продуктов с использованием объектного подхода. Диаграммы вариантов использования. Описание вариантов использования. Виды отношений между вариантами использования – ассоциация, расширение (extend), включение (include), обобщение.

Диаграмма классов. Построение концептуальной модели предметной области. Класс: имя класса, атрибуты класса, операции. Отношения между классами – отношение зависимости, ассоциации, агрегации, композиции, обобщения. Интерфейсы. Объекты. Параметризованные классы (шаблоны).

Диаграмма деятельности. Состояние действия. Переходы. Дорожки. Объекты.

Диаграмма последовательностей. Линия жизни объекта. Фокус управления. Сообщения. Ветвление потока управления. Стереотипы сообщений. Временные ограничения на диаграммах последовательностей.

Диаграмма кооперации. Объекты. Составные объекты. Связи. Сообщения.

Диаграмма пакетов.

Диаграммы состояний объекта. Состояние – имя состояния, список внутренних действий, начальное состояние, конечное состояние. Переход. Событие. Сторожевое условие. Выражение действия. Составное состояние и подсостояние. Последовательные подсостояния. Параллельные подсостояния. Историческое состояние. Сложные переходы.

Диаграмма компонентов. Имя компонента. Виды компонентов. Интерфейсы. Зависимости.

Диаграмма размещения. Узел. Соединения.

Лабораторная работа 4 (2 часа)

Работа с классами в C#. Наследование. Разработка UML – диаграмм.

Цель работы: Создать класс. Каждый разрабатываемый класс должен, как правило, содержать следующие элементы: скрытые поля; конструкторы с параметрами и без параметров, методы, свойства. Методы и свойства должны обеспечивать непротиворечивый, полный, минимальный и удобный интерфейс класса. При возникновении ошибок должны выбрасываться исключения. В программе должна выполняться проверка всех разработанных элементов класса.
Создать дочерний класс.

Самостоятельная работа

Тема учебной дисциплины	Вид самостоятельной работы студента (реферат, расчетно-графическая работа, др.)	Всего часов
Тема 4. Проектирование программного обеспечения при объектно-ориентированном подходе к программированию	Подготовка к лекции	3
	Оформление и подготовка к защите лабораторных работ	4
	Изучение дополнительного теоретического материала	8
	Подготовка к контрольной работе	4
	Курсовая работа	11
	ИТОГО:	30

Текущий контроль – устные опросы по лекции 4 и самостоятельно изученному материалу, выполнение контрольной работы.

Контрольная работа

Цель контрольной работы – закрепить материал лекции 4, а именно, построение UML- диаграмм на различных этапах проектирования объектно-ориентированного программного обеспечения.

Курсовая работа (12 часов)

Программная реализация курсовой работы. Тестирование и отладка приложения.

Коды формируемых компетенций: ОПК-2, ПК-4, ПК-5

Результаты освоения:

ОПК-2: Осваивать методики использования программных средств для решения практических задач. Освоить язык UML для проектирования объектно-ориентированных программных комплексов.

ПК-2 разрабатывать компоненты программных комплексов и баз данных, использовать современные инструментальные средства и технологии программирования.

Тема 5. Разработка пользовательских интерфейсов. Оценка качества программного обеспечения

Лекция 5 (2 часа)

Разработка пользовательского интерфейса. Типы пользовательских интерфейсов и этапы их разработки. Психофизические особенности человека, связанные с восприятием, запоминанием и обработкой информации. Пользовательская и программная модели интерфейса. Классификации диалогов и общие принципы их разработки. Основные компоненты графических пользовательских интерфейсов. Реализация диалогов в графическом пользовательском интерфейсе. Пользовательские интерфейсы прямого манипулирования и их проектирование. Интеллектуальные элементы пользовательских интерфейсов

Граф диалога с пользователем. Разработка графа абстрактного диалога управляемого системой. Разработка графа абстрактного диалога управляемого пользователем. Разработка графа абстрактного диалога комбинированного типа.

Оценка качества программного обеспечения по ГОСТ 28195-89. Факторы качества. Категории качества. Метрики качества. Формирование оценочных коэффициентов.

Лабораторная работа 5 (4 часа)

Оценка качества программного обеспечения.

Цель работы - разработать объектно-ориентированное, многооконное приложение с использование интерфейсов, делегатов и событий. Разработать UML – диаграммы. Обосновать тип пользовательского интерфейса и форму диалога. Разработать граф диалога пользователя. Разработать пользовательское меню. Выполнить оценку качества разработанного приложения по ГОСТ 28195-89.

Самостоятельная работа

Тема учебной дисциплины	Вид самостоятельной работы студента (реферат, расчетно-графическая работа, др.)	Всего часов
Тема 5. Разработка пользовательских интерфейсов. Оценка качества программного обеспечения	Подготовка к лекции	3
	Изучение дополнительного теоретического материала	8
	Оформление и подготовка к защите лабораторных работ	4
	Подготовка к контрольной работе	4
	Курсовая работа	11
	ИТОГО:	30

Текущий контроль – устные опросы по лекции 5 и самостоятельно изученному материалу, выполнение контрольной работы.

Контрольная работа

Применение методики оценки качества разработанного программного обеспечения на практике.

Курсовая работа (12 часов)

Студенты показывают сделанную работу преподавателю, оформляют РПЗ и сдают на проверку преподавателю.

Коды формируемых компетенций: ОПК-2, ПК-1

Результаты освоения:

ОПК-2: Выполнять оценку качества разработанных программных продуктов по ГОСТ 28195-89.

ПК1: Разрабатывать интерфейсы "человек - электронно-вычислительная машина". Разработка многомодульных многооконных приложений, освоение методов межпроцессных взаимодействий. Использование при проектировании графа диалога с пользователем.

Промежуточная аттестация по дисциплине: экзамен.

Изучение дисциплины заканчивается экзаменом. Экзамен проводится в соответствии с Положением о зачетной и экзаменационной сессиях в НИУ МЭИ и инструктивным письмом от 14.05.2012 г. № 21-23.

5 Самостоятельная работа студента

Для обеспечения самостоятельной работы разработаны:

1. Конспект лекций по дисциплине (см. приложение З.РПД Б1.В.ОД.6 (лк));
2. Методические указания к выполнению лабораторных работ (см. приложение З.РПД Б1.В.ОД.6 (лб));

3. Методические указания к самостоятельной работе студентов (см. приложение З.РПД Б1.В.ОД.6 (срс));
4. Методические рекомендации к курсовой работе студентов (см. приложение З.РПД Б1.В.ОД.6 (кр)).

Примерная тематика курсовых работ (КР):
Требования курсовой работе:
1. Программа должна быть реализована на C#
2. Программа должна предусматривать две операции – шифрование и дешифрование
3. Пользователь должен иметь возможность смены ключа (ввод с клавиатуры или из файла)
4. Отсутствуют ограничения на размер шифруемого текста
5. Предусмотреть возможность ввода текста для шифрования и дешифрования из файла.
6. Приложение должно быть оконным (не консольным)
7. Разработать граф диалога пользователя
8. Выполнить тестирование разработанной системы
9. Разработать функциональные диаграммы, схемы алгоритмов, структурную и функциональную схемы.
10. Выполнить оценку качества разработанного приложения по ГОСТ 28195-89

Примерные темы КР:
Разработать криптографическую систему для следующих криптоалгоритмов:
1 Аффинная система подстановок Цезаря
2 Систем Цезаря с ключевым словом
3 Шифрующие таблицы Трисемуса
4 Биграммный шифр Плейфейра
5 Шифр Гронсфельда
6 Магические квадраты
7 Система шифрования Вижинера
8 Диск Альберти
9 Двойной квадрат Уитстона
10 Программная реализация роторной машины
11 Шифрующие таблицы с двойной перестановкой
12 Поворотная решетка Кардано
13 Шифр Ришелье

6 Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине

6.1 Перечень компетенций с указанием этапов их формирования

При освоении дисциплины формируются следующие компетенции:
- общепрофессиональные ОПК-1, ОПК-2;
- профессиональные ПК-1, ПК-2.

Указанные компетенции формируются в соответствии со следующими этапами:
1. Формирование и развитие теоретических знаний, предусмотренных указанными компетенциями (лекционные занятия, самостоятельная работа студентов).
2. Приобретение и развитие практических умений, предусмотренных компетенциями (практические занятия, лабораторные работы, выполнение расчетно-графической работы, самостоятельная работа студентов).
3. Закрепление теоретических знаний, умений и практических навыков, предусмотренных компетенциями, в ходе защит лабораторных работ, выполнения расчетно-графической работы, а

также решения конкретных технических задач на практических занятиях, успешной сдачи экзамена.

6.2 Описание показателей и критериев оценивания компетенций на различных этапах их формирования, описания шкал оценивания

Формы текущего контроля по темам дисциплины

№№пп	Наименование темы дисциплины	Формы текущего контроля
1.	Тема 1	Контрольная работа
2.	Тема 2	Тест
3.	Тема 3	Контрольная работа
4.	Тема 4	Контрольная работа
5.	Тема 5	Контрольная работа

Виды контроля самостоятельной работы студентов и оценочные средства

№ п/п	№ курса	Тема учебной дисциплины	Виды контроля	Оценочные средства
1	4	Тема 2	Тест	«50%»Пороговый уровень освоения компетенции «70%»- Продвинутый уровень освоения компетенции «90%»- Высокий уровень освоения компетенции
2	4	Тема 1,3,4,5	Контрольная работа	«3»- Пороговый уровень освоения компетенции «4»- Продвинутый уровень освоения компетенции «5»- Высокий уровень освоения компетенции

Образовательные технологии, обеспечивающие результаты освоения дисциплины в форме компетенций

Код компетенции	Компонентный состав компетенции (дескрипторы)	Технологии формирования	Средства оценки
ОПК-1	Знать: Основы языка программирования C#. Основы работы в среде Visual Studio	лекции, лабораторные занятия	Опрос, контрольные работы
	Уметь: Уметь инсталлировать среду программирования Visual Strudio и разрабатывать приложения на языке программирования C#	лабораторные занятия	Опрос, контрольные работы
	Владеть: навыками инсталляции различных программных сред, работы с массивами, коллекциями, строками, файлам и файловыми потоками на языке программирования C# при разработке конкретных прикладных задач	лабораторные занятия	Опрос, контрольные работы
ОПК-2	Знать: Принципы построения современных	лекции,	Опрос

	многомодульных, многооконных приложений, методы межпроцессного взаимодействия и передачи сообщений между приложениями. Методику оценки качества разработанного программного обеспечения по ГОСТ 28195.89	лабораторные занятия	
	Уметь: Разрабатывать приложения в среде Visual Studio Подбирать оценочные элементы при оценке качества по ГОСТ 28195.89	лабораторные занятия	Опрос
	Владеть: Навыками тестирования и отладки разработанных приложений в среде Visual Studio Практическими навыками оценки качества разработанного программного обеспечения по ГОСТ 28195.89, сравнивать полученные данные с базовыми образцами.	лабораторные занятия	Опрос
ПК-1	Знать: Методы структурного программирования и проектирования. Методы тестирования и отладки приложений. Основы объектно-ориентированного программирования и языка UML. Основы работы с CASE средствами для структурного и объектно-ориентированного проектирования (BPWin, Microsoft Office Visio). Знать основы работы с CASE-средствами для создания баз данных (ERWin). Знать основные пункты технического задания	лекции, лабораторные занятия	Опрос
	Уметь: Создавать UML-диаграммы. Уметь проводить предпроектные исследования и составлять техническое задание. Работать в среде ERWin, BPWin, Microsoft Office Visio. Проектировать функциональные диаграммы, диаграммы потоков данных, диаграммы переходов состояний, структурные и функциональные схемы.	лабораторные занятия	Опрос
	Владеть: Методами использования UML-языка для разработки практических задач. CASE- средствами при проектировании структурных и объектно-ориентированных приложений, а также CASE- средствами для работы с базами данных.	лабораторные занятия	Опрос
ПК-2	Знать: требования к разработке технологичных продуктов, типы сцепления и связности модулей. Требования к оформлению программных продуктов	лекции, лабораторные занятия	Опрос, контрольная работа
	Уметь: Определять тип сцепления и связность модулей, разрабатывать приложения,	лабораторные занятия	Опрос, контрольная работа

	удовлетворяющие требованиям эффективности по времени выполнения и требуемой памяти		
	Владеть: Методикой оценки степени технологичности программных продуктов	лабораторные занятия	Опрос, контрольная работа

Оценка уровней сформированности компетенций в результате освоения учебной дисциплины

Коды компетенций	Уровни сформированности компетенции	Основные признаки уровня
Общепрофессиональные компетенции - ОПК		
ОПК-1	Пороговый уровень освоения компетенции	Знает: Основы языка программирования C#. Основы работы в среде Visual Studio
	Продвинутый уровень освоения компетенции	Дополнительно умеет: Инсталлировать среду программирования Visual Strudio и разрабатывать приложения на языке программирования C#
	Высокий уровень освоения компетенции	Дополнительно владеет: Навыками инсталляции различных программных сред, работы с массивами, коллекциями, строками, файлам и файловыми потоками на языке программирования C# при разработке конкретных прикладных задач
ОПК-2	Пороговый уровень освоения компетенции	Знает: Принципы построения современных многомодульных, многооконных приложений, методы межпроцессного взаимодействия и передачи сообщений между приложениями. Методику оценки качества разработанного программного обеспечения по ГОСТ 28195.89
	Продвинутый уровень освоения компетенции	Дополнительно умеет: Разрабатывать приложения в среде Visual Studio. Подбирать оценочные элементы при оценке качества по ГОСТ 28195.89
	Высокий уровень освоения компетенции	Дополнительно владеет: Навыками тестирования и отладки разработанных приложений в среде Visual Studio Практическими навыками оценки качества разработанного программного обеспечения по ГОСТ 28195.89, сравнивать полученные данные с базовыми образцами.
Профессиональные компетенции - ПК		
ПК-1	Пороговый уровень освоения компетенции	Знает: Методы структурного программирования и проектирования. Методы тестирования и отладки приложений. Основы объектно-ориентированного программирования и языка UML. Основы работы с CASE средствами для структурного и объектно-ориентированного проектирования (BPWin, Microsoft Office Visio). Знает основы работы с CASE-средствами для создания баз данных (ERWin). Знает основные

		пункты технического задания
	Продвинутый уровень освоения компетенции	Дополнительно умеет: Создавать UML-диаграммы, проводить предпроектные исследования и составлять техническое задание. Работать в среде ERWin, BPWin, Microsoft Office Visio. Проектировать функциональные диаграммы, диаграммы потоков данных, диаграммы переходов состояний, структурные и функциональные схемы.
	Высокий уровень освоения компетенции	Дополнительно владеет: Методами использования UML-языка для разработки практических задач. CASE- средствами при проектировании структурных и объектно-ориентированных приложений, а также CASE- средствами для работы с базами данных.
ПК-2	Пороговый уровень освоения компетенции	Знает: Требования к разработке технологичных продуктов, типы сцепления и связности модулей. Требования к оформлению программных продуктов
	Продвинутый уровень освоения компетенции	Дополнительно умеет: Определять тип сцепления и связность модулей, разрабатывать приложения, удовлетворяющие требованиям эффективности по времени выполнения и требуемой памяти.
	Высокий уровень освоения компетенции	Дополнительно владеет: Методикой оценки степени технологичности программных продуктов

6.3 Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций в процессе освоения образовательной программы

Вопросы к экзамену

1. Технология программирования. Основные понятия. Этапы развития технологии программирования.
2. Жизненный цикл и этапы разработки программного обеспечения
3. Модели жизненного цикла программного обеспечения
4. Понятия эффективности и технологичности программного обеспечения. Модули и их свойства (сцепление и связность модулей)
5. Структурное программирование
6. Средства описания структурных алгоритмов (псевдокоды, схемы алгоритмов)
7. Средства описания структурных алгоритмов (Flow-формы, диаграммы Насси-Шнейдермана)
8. Правила оформления программ
9. Разработка технического задания
10. Классификация моделей разрабатываемого программного обеспечения
11. Структурный подход. Диаграммы переходов состояний

12. Структурный подход. Функциональные диаграммы
13. Структурный подход. Диаграммы потоков данных
14. Структурный подход. Структуры данных и диаграммы отношений компонентов данных
15. Структурный подход. Сетевая модель данных (Диаграммы «сущность-связь»)
16. Проектирование программного обеспечения при структурном подходе. Структурная и функциональная схемы
17. Структурный подход. Структурные карты Константайна
18. Проектирование структур данных. Методика Джексона
19. UML- стандартный язык описания разработки программных продуктов с использованием объектного подхода
20. Диаграммы вариантов использования
21. Диаграмма классов. Отношения между классами
22. Диаграмма последовательностей
23. Диаграммы деятельности
24. Диаграмма пакетов
25. Диаграммы состояний объекта
26. Диаграмма кооперации
27. Диаграмма компонентов
28. Диаграмма размещения
29. Структурное тестирование. Тестирование базового пути
30. Структурное тестирование. Тестирование условий
31. Структурное тестирование. Тестирование циклов
32. Структурное тестирование. Тестирование потоков данных
33. Функциональное тестирование. Разбиение на классы эквивалентности и анализ граничных значений
34. Функциональное тестирование. Анализ причинно-следственных связей
35. Классификация ошибок
36. Методы отладки программного обеспечения
37. Разработка пользовательского интерфейса. Классификация диалогов и общие принципы их работы
38. Разработка пользовательского интерфейса. Граф диалога с пользователем
39. Оценка качества программного обеспечения по ГОСТ 28195-89

Примеры экзаменационных задач

ВАРИАНТ 1

Класс Деньги для работы с денежными суммами.
Число должно быть представлено двумя полями: для рублей и для копеек.
Дробная часть (копейки) при выводе на экран должна быть отделена от целой части запятой. Реализовать сложение и вычитание.
Создать дочерний класс, который выполняет деление, умножение и операцию сравнения.
Проверить работу созданных методов.

ВАРИАНТ 2

Класс Равнобочная трапеция, члены класса: координаты 4-х точек. Предусмотреть в классе конструктор и методы: проверка, является ли фигура равнобочной трапецией.
Дочерний класс выполняет вычисления и вывод сведений о фигуре: длины сторон, периметр, площадь.
Продемонстрировать работу с классом: дано N трапеций, найти количество трапеций, у которых площадь больше средней площади.

ВАРИАНТ 3

Построить описание класса, содержащего информацию о почтовом адресе организации. Предусмотреть возможность раздельного изменения составных частей адреса, создания и уничтожения объектов этого класса. Дочерний класс выполняет сравнение двух адресов.
Продемонстрировать работу с классом: дано N адресов, найти одинаковые адреса.

ВАРИАНТ 4

Составить регулярное выражение, является ли заданная строка IP адресом версии 4, записанным в десятичном виде.
Пример правильных выражений:
127.0.0.1
255.255.255.0
192.168.0.1
Пример неправильных выражений:
1300.6.7.8
abc.def.gha.bcd

ВАРИАНТ 6

С помощью регулярных выражений проверить, надежно ли составлен пароль. Пароль считается надежным, если он состоит из 8 или более символов. Где символом может быть английская буква, цифра и знак подчеркивания. Пароль должен содержать хотя бы одну заглавную букву, одну маленькую букву и одну цифру.
Пример правильных выражений:
C00l_Pass
SupperPas1
Пример неправильных выражений:
Cool_pass
C00l

Примеры тестов для промежуточной аттестации
Тема 2. Основы программирования в среде C#

1. Для чего служит строки, начинающиеся с двух или трех косых черт:
С таких строк начинается описание класса; Служат для обозначения ключевых слов; Такие строки являются комментариями и служат для документирования текста программы; Такие строки ограничивают блок операторов, выполняющихся в циклах или в операторах ветвления
2. Чему будет равно значение переменных a, b и c после выполнения следующего фрагмента программы?

```
{  int a = 0; int b = 0; int c = 0;
  a = (b = 2 + 3) / 2 - 4 + (c = 5 % 2);
  Console.WriteLine("a={0} b={1}   c={2}", a, b, c);}
```

3. Сколько раз будет напечатано слово «Привет!»

```
    static void Main(string[] args)
    {   const int LIMIT = 10;
        int i = 0;
        while (i < LIMIT)
        Console.WriteLine("Привет!");
         i++; }
```

4. Сколько раз будет напечатано слово «Привет!»

```
    static void Main(string[] args)
```

```
{    const int LIMIT = 10;
     int i = 0;
      while (i < LIMIT){
      Console.WriteLine("Привет!");
      i++;}}
```

5. Сколько раз будет напечатано слово «Привет!»

```
static void Main(string[] args)
     { const int LIMIT = 10;
       uint i = 0;
       while (i < LIMIT)
       {  Console.WriteLine("Привет!");
          i=i-1;}}
```

6. Сколько раз будет напечатано слово «Привет!»

```
static void Main(string[] args)
   {   const int LIMIT = 10;
       for (uint i = 0; i <= LIMIT; ++i)
           Console.WriteLine("Привет!");}
```

7. Сколько раз будет напечатано слово «Привет!»

```
    static void Main(string[] args)
    {   const int LIMIT = 10;
        uint i = 1;
        do
        {Console.WriteLine("Привет!");
                i=i+1;
        } while (i>LIMIT);}
```

8. Чему будет равна переменная i после завершения работы цикла?

```
static void Main(string[] args)
   {   const int LIMIT = 10;
       uint i = 0;
       for (i = 0; i <= LIMIT; ++i)
           Console.WriteLine("Привет!");
       Console.WriteLine("i={0}",i);}
```

9. Чему будет равна переменная a после завершения работы программы?

```
static void Main(string[] args)
   {   bool i = false;
       int a = 0;
       if (i = true) a = 3;
       else a = -3;
       Console.WriteLine("a={0}", a);}
```

10. Чему будет равна переменная i после завершения работы программы?

```
static void Main(string[] args)
     {   int i = -1;
         if (i <0 && (i!=-1)) i = 3;
         else i = -3;
         Console.WriteLine("i={0}", i);}
```

11. Чему будет равна переменная i после завершения работы программы?

```
static void Main(string[] args)
     {   int i = 0;
         if ((i==i) || (i==2)) i = 3;
         else i = -3;
         Console.WriteLine("i={0}", i);}
```

12. Чему будет равна переменная i после завершения работы программы?

```
static void Main(string[] args)
      {   int a=8;  int b=12;  int i =0;
          i=((2*b-2*8)+2)/3;
          if (i-3>0)
              if (i==3) i=1;
              else i = 2;
          Console.WriteLine("i={0}", i);}
```

13. Чему будет равно значение переменной a после завершения работы программы?

```
static void Main(string[] args)
      {   int a=8;  int b=12;  bool i =true;
          if (i == a > b)  a = 1;
          else
            if (i == !(a != b))  a = 2;
            else if (i ==(a< b)) a = 3;
                 else a = 4;
          Console.WriteLine("a={0}", a);}
```

14. Чему будет равно значение переменной a после завершения работы программы?

```
   static void Main(string[] args)
      {   int a=-10;
          a = (a + 1 > 0) ? a : -a;
          Console.WriteLine("a={0}", a);}
```

15. Чему будет равно значение переменной b после завершения работы программы?

```
static void Main(string[] args)
      {   int b=-9;
          b = (b + 1 >= 0) ? b : -b;
          Console.WriteLine("b={0}", b);}
```

16. Чему будет равно значение переменной c после завершения работы программы?

```
static void Main(string[] args)
      {   int c = -10; int a = 0;
          a= (c + 1 >= 0) ? c : -c;
          Console.WriteLine("c={0}", c);}
```

17. Чему будет равно значение переменной d после завершения работы программы?

```
static void Main(string[] args)
      {   int d = -10;
          d+= (d + 1 > 0) ? d : -d;
          Console.WriteLine("d={0}", d);}
```

18. Чему будет равно значение переменной e после завершения работы программы?

```
static void Main(string[] args)
      {   int e = -10;
          e= (--e> 0) ? e : -e;
          Console.WriteLine("e={0}", e);}
```

19. Чему будет равно значение переменной s после завершения работы программы?

```
static void Main(string[] args)
      {   int i = 5;
          float s = 1;
          s*=(i+=5)/4;
          Console.WriteLine("s={0}", s);}
```

20. Чему будет равно значение переменной s после завершения работы программы?

```
static void Main(string[] args)
      {   int i = 5;
```

```
            float s = 1;
            s*=(float)(i+=5)/4;
            Console.WriteLine("s={0}", s);}
```

21. Чему будет равно значение переменной i после завершения работы программы?

```
static void Main(string[] args)
      {   int i = 0; int s = 10;
          for (i = 0; s > 0; i++)
              s /= 2;
          Console.WriteLine("i={0}", i);}
```

22. Чему будет равно значение переменной j после завершения работы программы?

```
static void Main(string[] args)
      {   int j = 0; int s = 10;
          for (j = 0; s > 0 && (s /= 2) != 0; j++);
          Console.WriteLine("j={0}", j);}
```

23. Чему будет равно значение переменной j после завершения работы программы?

```
static void Main(string[] args)
      {   int j = 0; int s = 10;
          for (j = 0; s > 0 && (s %= 2) == 0; j++);
          Console.WriteLine("j={0}", j);}
```

24. Чему будет равно значение переменной k (в шестнадцатеричном формате) после завершения работы программы?

```
static void Main(string[] args)
      {   int i = 0x1234;  int k = 0;
          k=i<<4;
          Console.WriteLine("k={0:x}", k);}
```

25. Чему будет равно значение переменной j (в шестнадцатеричном формате) после завершения работы программы?

```
static void Main(string[] args)
      {   int i = 0x1234; int j = 0;
          j=i>>8;
          Console.WriteLine("j={0:x}", j); }
```

26. Чему будет равно значение переменной r (в шестнадцатеричном формате) после завершения работы программы?

```
static void Main(string[] args)
      {   int i = 0x45ff; int j = 0x00ff;
          int r =0;
          r=i^j;
          Console.WriteLine("r={0:x}", r);}
```

27. Чему будет равно значение переменной r (в шестнадцатеричном формате) после завершения работы программы?

```
static void Main(string[] args)
      {   int i = 0x45ff; int j = 0x00ff;
          int r =0;
          r=i|j;
          Console.WriteLine("r={0:x}", r);}
```

28. Чему будет равно значение переменной r (в шестнадцатеричном формате) после завершения работы программы?

```
static void Main(string[] args)
      {   int i = 0x45ff; int j = 0x00ff;
          int r =0;
          r=i&j;
```

```
            Console.WriteLine("r={0:x}", r);}
```

29. Чему будет равно значение переменной a после завершения работы программы?

```
static void Main(string[] args)
   {  int a = 5; int b = 2;
       switch  (b++)
       {   case 1: a += 3; break;
           case 2: a *= 3; break;
           case 3: a = a * 2 + 10; break;
           case 4: a %= 6; break; }
       Console.WriteLine("a={0}", a);}
```

30. Чему будет равно значение переменных a и b после завершения работы программы?

```
static void Main(string[] args)
    {  int a = 5; int b = 2; int c = 30; int d = 7;
       a += ++b+c;
       c -= d++ + ++b;
       Console.WriteLine("a={0}    c={1}", a,c);}
```

Примеры контрольных работ

Контрольная работа по Теме 1

Основные понятия и подходы. Приемы обеспечения технологичности программных продуктов

Вопросы:

1. Дайте определение технологичности программного обеспечения
2. Что понимают под связностью модуля, типы связности
3. Что понимают под сцеплением модуля, типы сцепления
4. Нисходящая и восходящая разработка программного обеспечения. Перечислите достоинства и недостатки данных методов.
5. Постройте диаграмму Насси-Шнейдермана для программы перевода числа из десятичной формы в шестнадцатеричную
6. Перечислите формы описания структурных алгоритмов, опишите особенности данных форм, их достоинства и недостатки
7. Постройте flow-форму для программы определения типа треугольника по трем его сторонам
8. Опишите метод оценки характеристик разработанных программ с помощью метрик Холстеда
9. Опишите метод оценки характеристик разработанных программ с помощью метрик Джилба
10. Опишите метод оценки надежности программных средств с помощью модели Джелински – Моранды
11. Опишите метод оценки надежности программных средств с помощью модели Нельсона
12. Опишите метод оценки надежности программных средств с помощью эвристической модели
13. Назовите основные эксплуатационные требования к программным продуктам. Какими средствами и приемами обеспечивается каждый из них? Для каких типов программных систем целесообразно указывать каждый из них?
14. В каких ситуациях необходимы предпроектные исследования? Какие вопросы при этом решают? Что получают в результате таких исследований?
15. Назовите, какой раздел технического задания можно считать основным и почему? Какую информацию должны содержать остальные разделы? В чем основная сложность разработки технического задания?
16. Составьте техническое задание на разработку «калькулятора» по типу, предлагаемого Windows.
17. Разработать программный модуль «Учет успеваемости стуков». Программный модуль предназначен для оперативного успеваемости студентов в сессию деканом, заместителями декана и

сотрудниками деканата. Сведения об успеваемости студентов должны храниться в течение всего срока их обучения и использоваться при составлении справок о прослушанных курсах и приложений к диплому.

18. Разработать программный модуль «Личные дела студентов. Программный модуль предназначен для получения сведений о студентах сотрудниками деканата, профкома и отдела кадров. Сведения должны храниться в течение всего срока обучения студентов и использоваться при составлении справок и отчетов.
19. Разработать программный модуль «Решение комбинаторно-оптимизационных задач». Модуль должен содержать алгоритмы поиска цикла минимальной длины (задача коммивояжера), поиска кратчайшего пути и поиска минимального связывающего дерева.
20. Разработать приложение Windows «Органайзер». Приложение предназначено для записи, хранения и поиска адресов и телефонов физических лиц и организаций, а также расписания, встреч и др. Приложение предназначено для любых пользователей компьютера.
21. Разработать приложение Windows «Калькулятор». Приложение предназначено для любых пользователей и должно содержать все арифметические операции (с соблюдением приоритетов) и желательно (но не обязательно) несколько математических функций.
22. Разработать программный модуль «Кафедра», содержащий сведения о сотрудниках кафедры (ФИО, должность, ученая степень, дисциплины, нагрузка, общественная работа, совместительство и др.). Модуль предназначен для использования сотрудниками отдела кадров и деканата.
23. Разработать программный модуль «Лаборатория», содержащий сведения о сотрудниках лаборатории (ФИО, пол, возраст, семейное положение, наличие детей, должность, ученая степень). Модуль предназначен для использования сотрудниками профкома и отдела кадров.
24. Разработать программный модуль «Автосервис». При записи на обслуживание заполняется заявка, в которой указываются ФИО владельца, марка автомобиля, вид работы, дата приема заказа и стоимость ремонта. После выполнения работ распечатывается квитанция.
25. Разработать программный модуль «Учет нарушений правил дорожного движения». Для каждой автомашины (и ее владельца) в базе хранится список нарушений. Для каждого нарушения фиксируется дата, время, вид нарушения и размер штрафа. При оплате всех штрафов машина удаляется из базы.
26. Разработать программный модуль «Картотека агентства недвижимости», предназначенный для использования работниками агентства. В базе содержатся сведения о квартирах (количество комнат, этаж, метраж и др.). При поступлении заявки на обмен (куплю, продажу) производится поиск подходящего варианта. Если такого нет, клиент заносится в клиентскую базу и оповещается, когда вариант появляется.
27. Разработать программный модуль «Картотека абонентов АТС». Картотека содержит сведения о телефонах и их владельцах. Фиксирует задолженности по оплате (абонентской и повременной). Считается, что повременная оплата местных телефонных разговоров уже введена.
28. Разработать программный модуль «Авиакасса», содержащий сведения о наличии свободных мест на авиамарщруты. В базе должны содержаться сведения о номере рейса, экипаже, типе самолета, дате и времени вылета, а также стоимости авиабилетов (разного класса). При поступлении заявки на билеты программа производит поиск подходящего рейса.
29. Разработать программный модуль «Книжный магазин», содержащий сведения о книгах (автор, название, издательство, год издания, цена). Покупатель оформляет заявку на нужные ему книги, если таковых нет, он заносится в базу и оповещается, когда нужные книги поступают в магазин.
30. Разработать программный модуль «Автостоянка». В программе содержится информация о марке автомобиля, его владельце, дате и времени въезда, стоимости стоянки, скидках, задолженности по оплате и др.

31. Разработать программный модуль «Кадровое агентство», содержащий сведения о вакансиях и резюме. Программный модуль предназначен как для поиска сотрудника, отвечающего требованиям руководителей фирмы, так и для поиска подходящей работы.

Контрольная работа по Теме 3. Тестирование и отладка программных продуктов при структурном подходе

1. Определите понятие тестирования.
2. Что такое тест? Поясните содержание процесса тестирования.
3. Что такое исчерпывающее тестирование?
4. Какие задачи решает тестирование?
5. Каких задач не решает тестирование?
6. Какие принципы тестирования вы знаете? В чем их отличие друг от друга?
7. В чем состоит суть тестирования «черного ящика»?
8. В чем состоит суть тестирования «белого ящика»?
9. Каковы особенности тестирования «белого ящика»?
10. Какие недостатки имеет тестирование «белого ящика»?
11. Какие достоинства имеет тестирование «белого ящика»?
12. Дайте характеристику способа тестирования базового пути.
13. Какие особенности имеет потоковый граф?
14. Поясните понятие независимого пути.
15. Поясните понятие цикломатической сложности.
16. Что такое базовое множество?
17. Какие свойства имеет базовое множество?
18. Какие способы вычисления цикломатической сложности вы знаете?
19. Поясните шаги способа тестирования базового пути.
20. Поясните достоинства, недостатки и область применения способа тестирования базового пути.
21. Дайте общую характеристику способов тестирования условий.
22. Какие типы ошибок в условиях вы знаете?
23. Какие методики тестирования условий вы знаете?
24. Поясните суть способа тестирования ветвей и операторов отношений. Какие он имеет
25. ограничения?
26. Каковы особенности тестирования методом «черного ящика»?
27. Какие категории ошибок выявляет тестирование методом «черного ящика»?
28. Какие достоинства имеет тестирование методом «черного ящика»?
29. Поясните суть способа разбиения по эквивалентности.
30. Что такое класс эквивалентности?
31. Какие правила формирования классов эквивалентности вы знаете?
32. Как выбирается тестовый вариант при тестировании по способу разбиения по эквивалентности?
33. Поясните суть способа анализа граничных значений.
34. Чем способ анализа граничных значений отличается от разбиения по эквивалентности?
35. Поясните правила анализа граничных значений.
36. то такое дерево разбиений? Каковы его особенности?
37. В чем суть способа диаграмм причин-следствий?
38. Что такое причина?
39. Что такое следствие?
40. Дайте общую характеристику графа причинно-следственных связей.
41. Какие функции используются в графе причин и следствий?
42. Какие ограничения используются в графе причин и следствий?
43. Поясните шаги способа диаграмм причин-следствий.

44. Какую структуру имеет таблица решений в способе диаграмм причин-следствий?
45. Как таблица решений преобразуется в тестовые варианты?

Контрольная работа по Теме 4
Проектирование программного обеспечения при объектно-ориентированном подходе к программированию

1. Какие элементы определяются в составе класса?
2. Приведите синтаксис описания класса в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
3. Какие модификаторы типа доступа Вам известны?
4. В чем заключаются особенности доступа членов класса с модификатором public, private, protected, internal?
5. Приведите синтаксис создания объекта в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
6. Что понимается под термином «конструктор», В чем состоит назначение конструктора, Каждый ли класс языка C# имеет конструктор, Какие умолчания для конструкторов приняты в языке C#? Приведите синтаксис конструктора класса в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
7. Что понимается под термином «деструктор», В чем состоит назначение деструктора? Приведите синтаксис деструктора класса в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.
8. Что понимается под термином «наследование»? Какая классификация объектов соответствует наследованию? Что общего имеет дочерний класс с родительским? В чем состоит различие между дочерним и родительским классами? Приведите синтаксис описания наследования классов в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.

Контрольная работа по Теме 5
Оценка качества программного обеспечения

1. Этапы жизненного цикла программного обеспечения в соответствии с ГОСТ 28195-89
2. Опишите четырехуровневую систему показателей оценки качества программного обеспечения
3. Опишите факторы качества, используемые для оценки качества программного обеспечения
4. Опишите алгоритм оценки качества программного обеспечения
5. Опишите процесс определения конкретных значений оценочных элементов

6.4 Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций

Процедуры оценивания знаний, умений, навыков, характеризующих этапы формирования компетенций , изложены в:

1. Конспект лекций по дисциплине (см. приложение З.РПД Б1.В.ОД.6 (лк));
2. Методические указания к выполнению лабораторных работ (см. приложение З.РПД Б1.В.ОД.6 (лб));
3. Методические указания к самостоятельной работе студентов (см. приложение З.РПД Б1.В.ОД.6 (срс));
4. Методические рекомендации к курсовой работе студентов (см. приложение З.РПД Б1.В.ОД.6(кр)).

7 Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины

Основная учебная литература

1. Буч Г. Язык UML. Руководство пользователя [Электронный ресурс] : / Буч Г., Рамбо Д., Якобсон И. — Электрон. дан. — М. : ДМК Пресс, 2008. — 494 с. — Режим доступа: http://e.lanbook.com/books/element.php?pl1_id=1246
2. Вишневская, Т.И. Технология программирования. Часть 1 [Электронный ресурс] : учебно-методическое пособие / Т.И. Вишневская, Т.Н. Романова. — Электрон. дан. — М. : МГТУ им. Н.Э. Баумана (Московский государственный технический университет имени Н.Э. Баумана), 2007. — 59 с. — Режим доступа: http://e.lanbook.com/books/element.php?pl1_id=52381
3. Вишневская, Т.И. Технология программирования. Часть 2 [Электронный ресурс] : учебно-методическое пособие / Т.И. Вишневская, Т.Н. Романова. — Электрон. дан. — М. : МГТУ им. Н.Э. Баумана (Московский государственный технический университет имени Н.Э. Баумана), 2010. — 52 с. — Режим доступа: http://e.lanbook.com/books/element.php?pl1_id=52411
4. ОСНОВЫ ПРОГРАММИРОВАНИЯ В СРЕДЕ C#. Методические указания к лабораторным работам по дисциплине «Технологии программирования» [Текст]: методические указания / А.И. Гаврилов, В.В. Малахов, И.В. Малашенкова, Е.А. Панкратова, О.В. Семенова – Смоленск: РИО филиала МЭИ в г.Смоленске, 2014. – 110 с.
5. Панкратова Е.А. Проектирование программного обеспечения.[Текст]: методические рекомендации/ Е.А. Панкратова, О.В. Семенова, В.В. Малахов - Смоленск: РИО филиала ГОУВПО «МЭИ(ТУ)» в г. Смоленске, 2010 . – 36 с.
6. Панкратова Е.А. Тестирование программного обеспечения [Текст]: методические рекомендации / Е.А. Панкратова, О.В. Семенова - Смоленск: РИО филиала МЭИ в г. Смоленске, 2011. – 24 с.

Дополнительная учебная литература

7. Вендров А.М. Проектирование программного обеспечения экономических информационных систем. Учебник для студентов ВУЗов 2-е издание, переработанное и дополненное / А.М. Вендров –М.: «Финансы и статистика», 2006 г. -534 с.:ил
8. Вендров А.М. Практикум по проектированию программного обеспечения экономических информационных систем. Учебник для студентов ВУЗов / А.М. Вендров –М.: «Финансы и статистика»,_ 2006 г. -190 с. :ил.
9. Вилле К. Представляем C# [Электронный ресурс] : . — Электрон. дан. — М. : ДМК Пресс, 2008. — 186 с. — Режим доступа: http://e.lanbook.com/books/element.php?pl1_id=1225
10. Гагарина Л.Г. Технология разработки программного обеспечения: учебное пособие / под ред. Л.Г. Гагариной –М: ИД «ФОРУМ»: ИНФРА-М, 2008. –400с.: ил. – (Высшее образование)
11. Иванова Г.С. Технология программирования: учебник / Г.С. Иванова –М. КНОРУС, 2011. - 336 с.
12. Орлов С.А. Технологии разработки программного обеспечения: Учебник /С.Орлов –СПб. Питер, 2012. -464 с.: ил.
13. Робисон У. C# без лишних слов [Электронный ресурс] : . — Электрон. дан. — М. : ДМК Пресс, 2008. — 342 с. — Режим доступа: http://e.lanbook.com/books/element.php?pl1_id=1240
14. Черников Б.В. Оценка качества программного обеспечения: Практикум; учебное пособие / Б.В. Черников, Б.Е. Поклонов /Под ред. Б.В. Черникова –М: ИД «ФОРУМ»: ИНФРА-М, 2012, -400 с.: ил.(Высшее образование)

8 Перечень ресурсов информационно-телекоммуникационной сети «Интернет» необходимых для освоения дисциплины

1. http://www.rugost.com/index.php - ГОСТ 19.xxx Единая система программной документации (ЕСПД)
2. http://standartgost.ru/0/2870/2880-edinaya_sistema_programmnoy_dokumentatsii - ГОСТ 19.xxx Единая система программной документации (ЕСПД)

9 Методические указания для обучающихся по освоению дисциплины

Дисциплина предусматривает 10 часов лекций и 12 часов лабораторных работ. Изучение курса завершается экзаменом.

Успешное изучение курса требует посещения лекций, активной работы на лабораторных работах, выполнения всех учебных заданий преподавателя, ознакомления с основной и дополнительной литературой.

Во время **лекции** студент должен вести краткий конспект.

Работа с конспектом лекций предполагает просмотр конспекта в тот же день после занятий. При этом необходимо пометить материалы конспекта, которые вызывают затруднения для понимания. При этом обучающийся должен стараться найти ответы на затруднительные вопросы, используя рекомендуемую литературу. Если ему самостоятельно не удалось разобраться в материале, необходимо сформулировать вопросы и обратится за помощью к преподавателю на консультации или на ближайшей лекции.

Обучающемуся необходимо регулярно отводить время для повторения пройденного материала, проверяя свои знания, умения и навыки по контрольным вопросам.

Лабораторные работы составляют важную часть профессиональной подготовки студентов. Они направлены на экспериментальное подтверждение теоретических положений и формирование учебных и профессиональных практических умений.

Выполнение студентами лабораторных работ направлено на:

обобщение, систематизацию, углубление, закрепление полученных теоретических знаний по конкретным темам дисциплин;

формирование необходимых профессиональных умений и навыков;

Содержание лабораторных работ фиксируется в РПД в разделе 4 настоящей программы.

При планировании лабораторных работ следует учитывать, что наряду с ведущей целью - подтверждением теоретических положений - в ходе выполнения заданий у студентов формируются практические умения и навыки обращения с лабораторным оборудованием, аппаратурой и пр., которые могут составлять часть профессиональной практической подготовки, а также исследовательские умения (наблюдать, сравнивать, анализировать, устанавливать зависимости, делать выводы и обобщения, самостоятельно вести исследование, оформлять результаты).

Состав заданий для лабораторной работы +спланирован с таким расчетом, чтобы за отведенное время они могли быть качественно выполнены большинством студентов.

Выполнению лабораторных работ предшествует проверка знаний студентов – их теоретической готовности к выполнению задания.

Помимо собственно выполнения работы для каждой лабораторной работы предусмотрена процедура защиты, в ходе которой преподаватель проводит устный или письменный опрос студентов для контроля понимания выполненных ими измерений, правильной интерпретации полученных результатов и усвоения ими основных теоретических и практических знаний по теме занятия.

При подготовке к **экзамену** в дополнение к изучению конспектов лекций и учебных пособий, необходимо пользоваться учебной литературой, рекомендованной к настоящей программе. При подготовке к экзамену нужно изучить теорию: определения всех понятий и подходы к оцениванию до состояния понимания материала и самостоятельно решить по нескольку типовых задач

из каждой темы. При решении задач всегда необходимо уметь качественно интерпретировать итог решения.

Самостоятельная работа студентов (СРС) по дисциплине играет важную роль в ходе всего учебного процесса. Методические материалы и рекомендации для обеспечения СРС готовятся преподавателем и являются неотъемлемой частью программы.

10 Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень программного обеспечения и информационных справочных систем (при необходимости)

При проведении **лабораторных работ** предусматривается использование персональных компьютеров, оснащенных необходимым комплектом лицензионного программного обеспечения. – Viusual Studio 2010 по подписке Dream Spark и Microsoft Visio по подписке Dream Spark.

11 Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине

Лекционные занятия:
Аудитория.
Лабораторные работы по данной дисциплине проводятся в компьютерных классах, оснащенных необходимым комплектом программного обеспечения.

Автор
канд. техн. наук, доцент  Е.А. Панкратова

Зав. кафедрой ВТ
д-р техн. наук, профессор  А.С. Федулов

Программа одобрена на заседании кафедры 31 августа 2016 года, протокол № 01.