АННОТАЦИЯ К РАБОЧЕЙ ПРОГРАММЕ ДИСЦИПЛИНЫ

Направление подготовки **09.03.01 «Информатика и вычислительная техника»**
Профиль подготовки **«Программное обеспечение средств вычислительной техники и автоматизированных систем»**

Б1.В.03 ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ

Индекс	Наименование	Семестр 4										Итого за курс									
		Контроль	Академических часов								з.е.	Контроль	Академических часов								з.е.
			Всего	Контакт.	Лек	Лаб	Пр	КРП	СР	Контроль			Всего	Контакт.	Лек	Лаб	Пр	КРП	СР	Контроль	Всего
Б1.В.03	Технология программирования	Экз КР	**252**	76	34	34		8	140	36	7	Экз КР	**252**	76	34	34		8	140	36	5

Формируемые компетенции: ПК-3, ПК-6.

№	Наименование видов занятий и тематик, содержание
1	лекционные занятия 17 шт. по 2 часа: *1.1. Основы программирования в среде C#* Язык программирования C#. Структура программы. Операции уменьшения, увеличения, операции условия. Логические операции. Типы данных. Приведение типов. Массивы в C#. Многомерные массивы. Ступенчатые массивы. Функции класса System.Array. Операторы условия и выбора. Операторы циклов. Циклы for и foreach. Классы для работы со строками System.String и System.StringBuilder. Регулярные выражения. Классы в C#. Наследование. Интерфейсы, делегаты, события. Объявление делегатов. Использование делегатов. Простые делегаты. Групповые делегаты. Анонимные методы. Определение событий. Коллекции. *1.2. 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. *1.3. Приемы обеспечения технологичности программных продуктов*

Год начала подготовки (по учебному плану) 2024
Образовательный стандарт № 929 от 19.09.2017

	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. *1.4. Классификация программного обеспечения* Классификация программных продуктов по функциональному признаку. Основные эксплуатационные требования к программным продуктам. Предпроектные исследования предметной области. Разработка технического задания. *1.5. Разработка технического задания* Определение технического задания. Принципиальные решения начальных этапов проектирования. Классификация моделей разрабатываемого программного обеспечения. *1.6. Проектирование программного обеспечения при структурном подходе к программированию* Анализ требований и определение спецификаций при структурном подходе. Спецификации процессов. Словарь терминов. *1.7. Проектирование программного обеспечения при структурном подходе к программированию* 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»). *1.8. Методики структурного проектирования программных средств* Структурная схема разрабатываемого программного обеспечения. Функциональная схема. Метод пошаговой детализации при составлении алгоритмов. Структурные карты Константайна. Структурные карты Джексона. *1.9. Тестирование и отладка программных продуктов при структурном подходе* 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. *1.10. Функциональное тестирование программного обеспечения.* Особенности функционального тестирования. Разбиение на классы эквивалентности. Анализ граничных значений. Диаграммы причинно-следственных связей. *1.11. Отладка программного обеспечения.*

	Классификация ошибок. Методы отладки программного обеспечения. Методы и средства получения дополнительной информации. Общая методика отладки программного обеспечения. *1.12 Разработка программных продуктов с использованием объектного подхода* 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). *1.13 Разработка программных продуктов с использованием объектного подхода* 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. *1.14 Разработка программных продуктов с использованием объектного подхода* 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. *1.15 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. *1.16 Типы и виды пользовательского интерфейса* 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. *1.17 Оценка качества разработанных программных продуктов по ГОСТ 28195-89.* Общие положения стандарта. Классификация показателей качества программных средств. Методика оценки качества программных средств.
2	лабораторные работы 9 шт. по 4 (2) часа: *2.1. Линейные и циклические программы*

	Изучение основ языка C# и знакомство с элементами управления C#. Составление линейных и циклических программ. Работа с математическими функциями C#. Особенности работы с математическими функциями в C# (библиотека Math); операторы ветвления (if) и выбора (switch); операторы циклов (for, foreach, while и do…while). *2.2. Работа с массивами в C#* Особенности работы с массивами в языке c#, свойства и методы класса system.Array. Решение задач с одномерными, двумерными прямоугольными и двумерными ступенчатыми массивами. *2.3. Работа со строками и регулярными выражениями* Работа с элементами класса String. Работа со строками класса StringBuilder. Регулярные выражения. Оценка характеристик разработанных программ с помощью метрик Холстеда; метрик Джилба. Оценка надежности программных средств с помощью модели Джелински – Моранды, эвристической модели и модели Нельсона. *2.4. Работа с файлами* Оценка степени отлаженности разрабатываемых приложений с помощью модели Миллса. Построение диаграммы переходов состояния, функциональной диаграммы. Построение структурной, функциональной схемы разрабатываемого программного обеспечения. *2.5. Ручное тестирование программного обеспечения* Структурное тестирование программного обеспечения. Функциональное тестирование программного обеспечения. Отладка программного обеспечения. Изучение методов тестирования программного обеспечения. Выполнение ручного тестирования программ. Выполнение структурного тестирования программ следующими методами: тестирование базового пути; тестирование условий; тестирование циклов; тестирование потоков данных. *2.6. Изучение методов тестирования и отладки программного обеспечения* 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. *2.7. 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. *2.8. Разработка пользовательских интерфейсов*

	Цель работы - разработать объектно-ориентированное, многооконное приложение с использованием интерфейсов, делегатов и событий, реализующее автоматизацию процессов в заданной предметной области. 2.9. *Разработка пользовательских интерфейсов* Разработать UML – диаграммы для созданного в предыдущей работе многооконного приложения. Обосновать тип пользовательского интерфейса и форму диалога. Разработать граф диалога пользователя. Разработать пользовательское меню.
3	практические занятия не предусмотрены в структуре дисциплины
4	курсовая работа *Проектирование, тестирование, отладка и оценка качества программных средств* *Цель курсовой работы по дисциплине «Технология программирования» – закрепление соответствующего лекционного материала дисциплины, приобретение практических навыков проектирования программных систем с использованием структурного подхода, формирование компетенций, связанных с обеспечением требуемых технологических свойств разрабатываемого программного обеспечения, овладение современными технологиями проектирования приложений, методами расчета качества разрабатываемого программного обеспечения.* Примеры тем курсовых работ: 1. Разработать программный модуль «Авиакасса», содержащий сведения о наличии свободных мест на авиамаршруты. 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. 2. 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. 3. 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. 4. Система «Участковые». В системе должны поддерживаться режимы учета участковых (Ф.И.О., телефоны и т.д.) и их обслуживаемой территории (названия улиц, номера домов и т.д.), происшествий на обслуживаемой территории (дата происшествия, его вид, принятые меры и т.д.), поиска участкового по названию улицы, номера дома, анализа происшествий на обслуживаемой территории 5. Система «Утраченное оружие». В системе должны поддерживаться режимы учета утраченного оружия (похищенного, утерянного), поиска оружия по индивидуальному номеру, формирования списка разыскиваемого оружия по видам (боевое, ручное, стрелковое, служебное, гражданское, газовое оружие и т.д.), анализ утраченного оружия по времени года, по районам города. 6. 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. 7. 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. 8. «Федеральный розыск». В системе должны поддерживаться режимы учета лиц, объявленных в федеральный и

	межгосударственный розыск, поиска лиц по различным реквизитам (фамилия, имя, отчество, кличка, дата рождения, номер паспорта и т.д.), формирование списка лиц пропавших без вести. 9. Система «Паспорт». В системе должны поддерживаться режимы ввода информации о гражданине, результатах его проверки по спец. учетам; поиска гражданина по реквизитам его паспорта; поиска паспорта по ФИО гражданина, формирование списка паспортов с истекшим сроком действия, проверка паспорта на его подлинность. 10. Система «Штрафы за нарушение правил дорожного движения пешеходами». В системе должны поддерживаться режимы учета штрафов за нарушение правил дорожного движения, а также поиск неоплаченных штрафов.
5	расчетно-графическая работа не предусмотрена в структуре дисциплины
6	Самостоятельная работа студентов: 6.1. 4 контрольных опроса после 6-й, 10-й , 14-й и 17-й лекций; 6.2. Закрепление материала по тематике лекционных занятий: закрепление изучения материалов лекций 1.1-1.17 – основы программирования на языке высокого уровня; классификация программного обеспечения; проектирование программного обеспечения при структурном и объектно-ориентированном подходе к программи-рованию; тестирование и отладка программных продуктов; оценка качества разработанных программных средств. 6.3. Подготовка к экзамену по дисциплине (оценочные материалы приведены в разделе 6 настоящей РПД).