МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
"Казанский (Приволжский) федеральный университет"
Институт вычислительной математики и информационных технологий

УТВЕРЖДАЮ
Проректор по образовательной деятельности КФУ
Проф. Д.А. Таюрский
_____ «___»_____20__г.

*подписано электронно-цифровой подписью*

# Программа дисциплины

Технологии и стандарты разработки программного обеспечения

Направление подготовки: 10.03.01 - Информационная безопасность
Профиль подготовки: Безопасность компьютерных систем
Квалификация выпускника: бакалавр
Форма обучения: очное
Язык обучения: русский
Год начала обучения по образовательной программе: 2020

**Содержание**





ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

Программу дисциплины разработал(а)(и) доцент, к.н. (доцент) Шаймухаметов Р.Р. (кафедра системного анализа и информационных технологий, отделение фундаментальной информатики и информационных технологий), Ramil.Shaimukhametov@kpfu.ru

**1. Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с планируемыми результатами освоения ОПОП ВО**

Обучающийся, освоивший дисциплину (модуль), должен обладать следующими компетенциями:

| Шифр компетенции | Расшифровка приобретаемой компетенции |
|---|---|
| ОПК-4 | способность понимать значение информации в развитии современного общества, применять информационные технологии для поиска и обработки информации |
| ОПК-7 | способность определять информационные ресурсы, подлежащие защите, угрозы безопасности информации и возможные пути их реализации на основе анализа структуры и содержания информационных процессов и особенностей функционирования объекта защиты |
| ПК-15 | способность организовать технологический процесс защиты информации ограниченного доступа в соответствии с нормативными правовыми актами и нормативными методическими документами Федеральной службы безопасности Российской Федерации, Федеральной службы по техническому и экспортному контролю |

Обучающийся, освоивший дисциплину (модуль):

Должен знать:

содержание действующих российских и международных стандартов в области создания программных средств,содержание действующих российских стандартов документирования программных средств, современное состояние развития CASE-средств и промышленных технологий проектирования ПО, современные методы проектирования ПО, принципы организации и методики тестирования при испытании сложных ПС и определения их надежности, методы управления разработкой сложных программных систем

Должен уметь:

составлять модель жизненного цикла для проектирования ПС,

формировать цели, задачи и требования к проектируемому ПС,

на основе построения и анализа моделей AS-IS и TO-BE,

строить семейство моделей проектируемой программной системы на основе выбранного метода проектирования,

применять инструментальные CASE-средства для разработки программного обеспечения,

выбирать и применять методы тестирования ПС,

составлять документацию, сопровождающую проектирование ПС на всех его этапах

Должен владеть:

-навыками создания документации по программному проекту

-навыками разработки ПО различного уровня сложности

Должен демонстрировать способность и готовность:

-применять полученные знания и навыки в своей будущей профессиональной деятельности

**2. Место дисциплины (модуля) в структуре ОПОП ВО**

Данная дисциплина (модуль) включена в раздел "Б1.В.ДВ.06.02 Дисциплины (модули)" основной профессиональной образовательной программы 10.03.01 "Информационная безопасность (Безопасность компьютерных систем)" и относится к дисциплинам по выбору.

Осваивается на 4 курсе в 7 семестре.

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



ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

Общая трудоемкость дисциплины составляет 3 зачетных(ые) единиц(ы) на 108 часа(ов).

Контактная работа - 54 часа(ов), в том числе лекции - 36 часа(ов), практические занятия - 0 часа(ов), лабораторные работы - 18 часа(ов), контроль самостоятельной работы - 0 часа(ов).

Самостоятельная работа - 54 часа(ов).

Контроль (зачёт / экзамен) - 0 часа(ов).

Форма промежуточного контроля дисциплины: зачет в 7 семестре.

**4. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий**

**4.1 Структура и тематический план контактной и самостоятельной работы по дисциплине (модулю)**

| N | Разделы дисциплины / модуля | Семестр | Виды и часы контактной работы, их трудоемкость (в часах) | | | Самостоятельная работа |
|---|---|---|---|---|---|---|
| | | | Лекции | Практические занятия | Лабораторные работы | |
| 1. | Тема 1. Программное обеспечение ЭВМ. | 7 | 2 | 0 | 1 | 4 |
| 2. | Тема 2. Пакеты прикладных программ. | 7 | 2 | 0 | 1 | 4 |
| 3. | Тема 3. Программные средства. | 7 | 4 | 0 | 1 | 4 |
| 4. | Тема 4. Жизненный цикл программного обеспечения. | 7 | 2 | 0 | 1 | 4 |
| 5. | Тема 5. Модели жизненного цикла программного обеспечения. | 7 | 4 | 0 | 1 | 4 |
| 6. | Тема 6. Разработка требований и внешнее проектирование ПО. | 7 | 2 | 0 | 1 | 4 |
| 7. | Тема 7. Структурный подход к проектированию программного обеспечения. | 7 | 2 | 0 | 1 | 4 |
| 8. | Тема 8. Проектирование и программирование модулей. | 7 | 2 | 0 | 1 | 4 |
| 9. | Тема 9. Объектно-ориентированный подход к проектированию программного обеспечения. | 7 | 2 | 0 | 1 | 4 |
| 10. | Тема 10. Проектирование и разработка интерфейса ПО. | 7 | 2 | 0 | 1 | 6 |
| 11. | Тема 11. Тестирование, отладка и сборка ПО. | 7 | 2 | 0 | 1 | 4 |
| 12. | Тема 12. Сопровождение ПО на стадии эксплуатации. | 7 | 2 | 0 | 1 | 2 |
| 13. | Тема 13. Управление разработкой ПО. | 7 | 2 | 0 | 2 | 2 |
| 14. | Тема 14. Документация ПО. | 7 | 2 | 0 | 2 | 2 |
| 15. | Тема 15. Разработка и стандартизация информационных технологий. | 7 | 4 | 0 | 2 | 2 |
| | Итого | | 36 | 0 | 18 | 54 |

**4.2 Содержание дисциплины (модуля)**

**Тема 1. Программное обеспечение ЭВМ.**

Технология программирования (ТП) - технология разработки программного средства (ПС), включающая все процессы, начиная с момента зарождения идеи этого средства. Результатом применения ТП является программа, действующая в заданной вычислительной среде, хорошо отлаженная и документированная, доступная для понимания и развития в процессе сопровождения.

Процесс разработки ПС и методы оценивания продуктов стандартизованы (ISO/IEC 12207, 9126 и др.). Все это способствует повышению эффективности проектирования, разработки, тестирования и оценки качества ПС.



ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

Архитектура ПС - это представление ПС как системы, состоящей из совокупности взаимодействующих подсистем. В качестве таких подсистем выступают отдельные программы. Разработка архитектуры является первым этапом упрощения создаваемого ПС путем выделения независимых компонент.

Основные задачи разработки архитектуры ПС [1]:

1. выделение программных подсистем и отображение на них внешних функций (заданных во внешнем описании) ПС;

2. определение способов взаимодействия между выделенными программными подсистемами.

С учетом принимаемых на этом этапе решений производится дальнейшая конкретизация функциональной спецификации.

Проектирование ПС ? это процесс, следующий за этапами анализа и формирования требований. Модели анализа поставляют этапу проектирования исходные сведения для работы. Информационная модель описывает информацию, которую должно обрабатывать ПС. Функциональная модель определяет перечень функций обработки. Поведенческая модель закрепляет динамику системы. На выходе этапа проектирования ? разработка данных, разработка архитектуры и процедурная разработка ПС

**Тема 2. Пакеты прикладных программ.**

В качестве модульной структуры программы принято использовать древовидную структуру. В узлах такого дерева размещаются программные модули, а направленные дуги показывают подчиненность модулей.

В процессе разработки программы ее модульная структура может формироваться и использоваться по-разному для определения порядка программирования и отладки модулей, указанных в этой структуре. Обычно рассматриваются два метода :

 метод восходящей разработки;

 метод нисходящей разработки.

Метод восходящей разработки заключается в следующем. Сначала строится модульная структура программы в виде дерева. Затем поочередно программируются модули программы, начиная с модулей самого нижнего уровня, в таком порядке, чтобы для каждого программируемого модуля были уже запрограммированы все модули, к которым он может обращаться. После того, как все модули программы запрограммированы, производится их тестирование и отладка в порядке, в каком велось их программирование.

Технологией программирования не рекомендуется восходящий порядок разработки программы по следующим причинам.

 Каждая программа подчиняется некоторым внутренним для нее, но глобальным для ее модулей соображениям. При восходящей разработке эта глобальная информация для модулей нижних уровней еще не ясна в полном объеме. Часто приходится перепрограммировать модули, когда уточняется эта информация.

 При восходящем тестировании для каждого модуля приходится создавать ведущую программу, которая подготавливает для тестируемого модуля необходимое состояние информационной среды и производит требуемое обращение к нему. Это приводит к большому объему отладочного программирования.

Метод нисходящей разработки заключается в следующем. Как и в предыдущем методе сначала строится модульная структура программы в виде дерева. Затем поочередно программируются модули программы, начиная с верхнего уровня. Переход к программированию следующего модуля происходит в том случае, если запрограммирован модуль, который к нему обращается. После того, как все модули программы запрограммированы, производится их поочередное тестирование и отладка в таком же порядке.

Первым тестируется головной модуль программы при том состоянии информационной среды, при котором начинает выполняться эта программа. Те модули, к которым может обращаться головной, заменяются их имитаторами. Имитатор модуля представляется программой, которая сигнализирует о факте обращения к имитируемому модулю.

После завершения тестирования и отладки головного и любого последующего модуля производится переход к тестированию модулей, которые представлены имитаторами. Для этого имитатор заменяется самим этим модулем и добавляются имитаторы тех модулей, к которым он может обращаться. При этом каждый такой модуль будет тестироваться при тех состояниях информационной среды, которые возникают к моменту обращения к этому модулю при выполнении тестируемой программы. Таким образом, большой объем отладочного программирования при восходящем тестировании заменяется программированием простых имитаторов.

Особенностью классических методов восходящей и нисходящей разработок является требование, чтобы модульная структура программы была разработана до начала программирования модулей. Это требование соответствует водопадному подходу к разработке ПС. Однако, не всегда до программирования модулей можно точно и содержательно разработать структуру программы. Конструктивный и архитектурный подходы к разработке программ предлагают формирование модульной структуры в процессе программирования модулей.



ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

Конструктивный подход к разработке программы представляет собой модификацию нисходящей разработки, при которой модульная древовидная структура программы формируется в процессе программирования модулей. Разработка программы при конструктивном подходе начинается с программирования головного модуля исходя из спецификации программы в целом. Если эта программа большая, выделяются подзадачи.

**Тема 3. Программные средства.**

Программу для ее упрощения разрабатывают по частям, которые называются программными модулями. Такой метод разработки программ называют модульным программированием.

Программный модуль ? фрагмент описания процесса, оформляемый как самостоятельный программный продукт, пригодный для использования в описаниях разных процессов. Программный модуль программируется, компилируется и отлаживается отдельно; может включаться в состав разных программ; является средством борьбы со сложностью программ; является средством борьбы с дублированием в программировании.

Основными характеристиками программного модуля являются [1, 7]:

- размер;
- связность;
- сцепление с другими модулями;
- рутинность.

Размер модуля измеряется числом содержащихся в нем операторов или строк. Модуль не должен быть слишком маленьким или слишком большим. Маленькие модули приводят к громоздкой модульной структуре программы. Большие модули неудобны для изучения и изменений, могут существенно увеличить суммарное время повторных трансляций программы при ее отладке.

Отладка модуля размером в одну страницу может быть в разы проще отладки модуля размером в одну страницу и еще 4-5 строк на другой странице. Это связано с принципами организации человеческой памяти. Есть сверхоперативная память, связанная, в основном, со зрением. Эта память имеет очень быстрый доступ, но очень мала ? 7-9 позиций. Существенно больше оперативная память, в которой и происходит вся основная мыслительная деятельность, но данные в ней не могут храниться долго. Наконец, самая большая ? долговременная память. Человеку непросто заложить туда данные, но хранятся они долго.

С устройством памяти связан принцип центрального зрения. Человек хорошо воспринимает какую-то точку и то, что ее окружает. Если при отладке программы автор должен обозревать больше, чем одну небольшую страницу текста, он не может полноценно воспринять программу - листать вредно.

Связность модуля - мера зависимости его частей, внутренняя характеристика. Чем выше связность модуля, тем больше связей он скрывает от внешней части программы и больший вклад в упрощение программы вносит. Для оценки степени связности модуля используется семь типов связности

**Тема 4. Жизненный цикл программного обеспечения.**

Под жизненным циклом программного средства (ЖЦПС) понимают весь период его разработки и эксплуатации, начиная от момента возникновения замысла ПС и кончая прекращением его использования. В настоящее время можно выделить пять основных подходов к организации процесса создания и использования ПС.

Водопадный подход состоит из цепочки этапов. На каждом этапе создаются документы, используемые на последующем этапе. В исходном документе фиксируются требования к ПС. В конце этой цепочки создаются программы, включаемые в ПС.

? Исследовательское программирование предполагает быструю реализацию рабочих версий программ ПС, выполняющих в первом приближении требуемые функции. После экспериментального применения реализованных программ производится их модификация. Этот процесс повторяется до тех пор, пока ПС не будет достаточно приемлемо для пользователей. Такой подход применялся на ранних этапах развития программирования (интуитивная технология). В настоящее время этот подход применяется для разработки таких ПС, для которых пользователи не могут точно сформулировать требования (например, для разработки систем искусственного интеллекта).

? Прототипирование. Этот подход моделирует начальную фазу исследовательского программирования вплоть до создания рабочих версий программ, предназначенных для проведения экспериментов с целью установить требования к ПС. С самого начала разработчики пытаются выделить основные требования заказчика и реализовать их в виде работающего прототипа системы. Цикл разработки и показа прототипа повторяется несколько раз.

**Тема 5. Модели жизненного цикла программного обеспечения.**

Обобщением модели создания прототипов является спиральная модель, в которой разработка приложения выглядит как серия последовательных итераций. При большом числе итераций разработка по этой модели нуждается в автоматизации всех процессов, иначе она становится неэффективной.



ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

Формальные преобразования. Этот подход включает разработку формальных спецификаций ПС и превращение их в программы путем корректных преобразований. На этом подходе базируется компьютерная технология (CASE-технология) разработки ПС.

Сборочное программирование. Этот подход предполагает, что ПС конструируется из компонент, которые уже существуют. Должна быть библиотека таких компонент, каждая из которых может многократно использоваться в разных ПС. Процесс разработки ПС при данном подходе состоит скорее из сборки программ из компонент, чем из их программирования

Основное назначение моделей ЖЦ ПС

- Планирование и распределение работ между разработчиками, управление проектом.
- Обеспечение взаимодействия между разработчиками проекта и заказчиком.
- Контроль работ, оценивание промежуточных результатов заданным требованиям. Согласование промежуточных результатов с заказчиком.
- Проверка правильности конечного продукта путем его тестирования на запланированных и согласованных с заказчиком наборах тестов.
- Оценивание соответствия характеристик качества полученного продукта заданным требованиям.
- Определение направлений усовершенствования или модернизации продукта.

На сегодня основой формирования новой модели ЖЦ для конкретной прикладной системы является международный стандарт ISO/IEC 12207 ?Информационная технология. Процессы жизненного цикла программных средств?, который задает полный набор процессов, охватывающий все возможные виды работ и задач, связанных с построением ПС, начиная с анализа предметной области и кончая изготовлением соответствующего продукта. Данный стандарт содержит основные и вспомогательные процессы

**Тема 6. Разработка требований и внешнее проектирование ПО.**

Под моделью ПО в общем случае понимается формализованное описание системы ПО на определенном уровне абстракции. Каждая модель определяет конкретный аспект системы, использует набор диаграмм и документов заданного формата, а также отражает точку зрения и является объектом деятельности различных людей с конкретными интересами, ролями или задачами.

Графические (визуальные) модели представляют собой средства для визуализации, описания, проектирования и документирования архитектуры системы. Разработка модели системы ПО промышленного характера в такой же мере необходима, как и наличие проекта при строительстве большого здания. Это утверждение справедливо как в случае разработки новой системы, так и при адаптации типовых продуктов класса R/3 или BAAN, в составе которых также имеются собственные средства моделирования. Хорошие модели являются основой взаимодействия участников проекта и гарантируют корректность архитектуры. Поскольку сложность систем повышается, важно располагать хорошими методами моделирования. Хотя имеется много других факторов, от которых зависит успех проекта, но наличие строгого стандарта языка моделирования является весьма существенным.

Состав моделей, используемых в каждом конкретном проекте, и степень их детальности в общем случае зависят от следующих факторов:

сложности проектируемой системы;

необходимой полноты ее описания;

знаний и навыков участников проекта;

времени, отведенного на проектирование.

Визуальное моделирование оказало большое влияние на развитие ТС ПО вообще и CASE-средств в частности. Понятие CASE (Computer Aided Software Engineering) используется в настоящее время в весьма широком смысле. Первоначальное значение этого понятия, ограниченное только задачами автоматизации разработки ПО, в настоящее время приобрело новый смысл, охватывающий большинство процессов жизненного цикла ПО.

CASE-технология представляет собой совокупность методов проектирования ПО, а также набор инструментальных средств, позволяющих в наглядной форме моделировать предметную область, анализировать эту модель на всех стадиях разработки и сопровождения ПО и разрабатывать приложения в соответствии с информационными потребностями пользователей. Большинство существующих CASE-средств основано на методах структурного или объектно-ориентированного анализа и проектирования, использующих спецификации в виде диаграмм или текстов для описания внешних требований, связей между моделями системы, динамики поведения системы и архитектуры программных средств.

**Тема 7. Структурный подход к проектированию программного обеспечения.**



ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

Структурный подход к проектированию программного обеспечения. Характеристика и основные принципы структурного подхода. SADT (Structured Analysis and Design Technique), DFD (Data Flow Diagrams) и ERD (Entity-Relationship Diagrams) модели структурного подхода. Концепции функциональной модели SADT. Состав функциональной модели. Построение иерархии диаграмм моделей стандарта IDEF0. Типы связей между функциями.

**Тема 8. Проектирование и программирование модулей.**

Восходящее проектирование (или проектирование ?снизу вверх?) основано на выделении нескольких достаточно крупных модулей, реализующих некоторые функции в общей программе. При выделении модулей опираются на доступность реализуемых функций для понимания, простоту структурирования данных, существование готовых программ и модулей для реализации заданных функций, возможности переделки существующих программ для новых целей; имеет значение и размер будущего модуля. Каждый модуль при восходящем проектировании автономно программируется, тестируется и отлаживается. После этого отдельные модули объединяются в подсистемы с помощью управляющего модуля, в котором определяется последовательность вызовов модулей, ввод-вывод и контроль данных и результатов. В свою очередь, подсистемы затем объединяются в более сложные системы и в общий программный комплекс, который подвергается комплексной отладке с проверкой правильности межмодульных связей.

Рассмотренный подход можно рекомендовать при разработке не очень сложных программ. Если размеры подпрограмм невелики, то целесообразно выделить подпрограммы и начать программирование с их составления.

Основные недостатки восходящего проектирования программы проявляются в сложности объединения модулей в единую систему, в трудности выявления и исправления ошибок, допущенных на ранних стадиях разработки модулей. Кроме того, отдельные модули могут создаваться без общего представления о структуре всей системы, что затрудняет их объединение.

Для создания сложных программ можно рекомендовать нисходящее проектирование, основанное на выделении в решаемой задаче иерархии уровней обобщения. Схема иерархии уровней обобщения позволяет программисту сначала сконцентрировать внимание на том, что нужно сделать, и лишь затем ? на том, как это сделать.

Ведущая программа записывается как программа верхнего уровня, управляющая вызовами модулей более низкого уровня. Каждый из модулей более низкого уровня, в свою очередь, управляет вызовами модулей еще более низкого уровня. Такой способ проектирования позволяет создавать сложные и громоздкие программы из небольших простых модулей: размер задачи отражается только в числе модулей и уровней обобщений.

При нисходящем проектировании появляется возможность использовать вертикальное управление в схеме иерархии с использованием таких правил: модуль возвращает управление вызвавшему; модуль вызывает только модули более низкого уровня; принятие основных решений возлагается на модули максимально высокого уровня.

**Тема 9. Объектно-ориентированный подход к проектированию программного обеспечения.**

Концептуальной основой объектно-ориентированного анализа и проектирования ПО (ООАП) является объектная модель. Ее основные принципы (абстрагирование, инкапсуляция, модульность и иерархия) и понятия (объект, класс, атрибут, операция, интерфейс и др.) наиболее четко сформулированы Гради Бучем в его фундаментальной книге и последующих работах.

Большинство современных методов ООАП основаны на использовании языка UML. Унифицированный язык моделирования UML (Unified Modeling Language) представляет собой язык для определения, представления, проектирования и документирования программных систем, организационно-экономических систем, технических систем и других систем различной природы. UML содержит стандартный набор диаграмм и нотаций самых разнообразных видов.

UML - это преемник того поколения методов ООАП, которые появились в конце 1980-х и начале 1990-х годов. Создание UML фактически началось в конце 1994 г., когда Гради Буч и Джеймс Рамбо начали работу по объединению их методов Booch и OMT (Object Modeling Technique) под эгидой компании Rational Software. К концу 1995 г. они создали первую спецификацию объединенного метода, названного ими Unified Method, версия 0.8. Тогда же в 1995 г. к ним присоединился создатель метода OOSE (Object-Oriented Software Engineering) Ивар Якобсон. Таким образом, UML является прямым объединением и унификацией методов Буча, Рамбо и Якобсона, однако дополняет их новыми возможностями. Главными в разработке UML были следующие цели:

предоставить пользователям готовый к использованию выразительный язык визуального моделирования, позволяющий им разрабатывать осмысленные модели и обмениваться ими;

предусмотреть механизмы расширяемости и специализации для расширения базовых концепций;

обеспечить независимость от конкретных языков программирования и процессов разработки.

обеспечить формальную основу для понимания этого языка моделирования (язык должен быть одновременно точным и доступным для понимания, без лишнего формализма);

стимулировать рост рынка объектно-ориентированных инструментальных средств;

интегрировать лучший практический опыт.



ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

**Тема 10. Проектирование и разработка интерфейса ПО.**

Интерфейсы являются основой взаимодействия всех современных информационных систем. Если интерфейс какого-либо объекта (персонального компьютера, программы, функции) не изменяется (стабилен, стандартизирован), это даёт возможность модифицировать сам объект, не перестраивая принципы его взаимодействия с другими объектами.

Например, научившись работать с одной программой под Windows, пользователь с легкостью освоит и другие ? потому, что они имеют одинаковый интерфейс.

В вычислительной системе взаимодействие может осуществляться на пользовательском, программном и аппаратном уровнях. В соответствии с этой классификацией можно выделить:

Интерфейс пользователя ? совокупность средств, при помощи которых пользователь общается с различными устройствами.

Интерфейс командной строки: инструкции компьютеру даются путём ввода с клавиатуры текстовых строк (команд).

Графический интерфейс пользователя: программные функции представляются графическими элементами экрана.

Диалоговый интерфейс

Естественно-языковой интерфейс: пользователь ?разговаривает? с программой на родном ему языке.

Физический интерфейс

Способ взаимодействия физических устройств. Чаще всего речь идёт о компьютерных портах.

Сетевой интерфейс

Шлюз (телекоммуникации) ? устройство, соединяющее локальную сеть с более крупной, например, Интернетом

Шина (компьютер)

Нейро-компьютерный интерфейс (англ. brain-computer interface): отвечает за обмен между нейронами и электронным устройством при помощи специальных имплантированных электродов.

Интерфейсы в программировании

Интерфейс функции

Интерфейс программирования приложений (API): набор стандартных библиотечных методов, который программист может использовать для доступа к функциональности другой программы.

Вызов удалённых процедур

COM-интерфейс

Интерфейс (ООП)

**Тема 11. Тестирование, отладка и сборка ПО.**

Учитывая разнообразие источников ошибок, при составлении плана тестирования классифицируют ошибки на два типа: 1 ? синтаксические; 2 ? семантические (смысловые).

Синтаксические ошибки ? это ошибки в записи конструкций языка программирования (чисел, переменных, функций, выражений, операторов, меток, подпрограмм).

Семантические ошибки ? это ошибки, связанные с неправильным содержанием действий и использованием недопустимых значений величин.

Обнаружение большинства синтаксических ошибок автоматизировано в основных системах программирования. Поиск же семантических ошибок гораздо менее формализован; часть их проявляется при исполнении программы в нарушениях процесса автоматических вычислений и индицируется либо выдачей диагностических сообщений рабочей программы, либо отсутствием печати результатов из-за бесконечного повторения одной и той же части программы (зацикливания), либо появлением непредусмотренной формы или содержания печати результатов.

В план тестирования обычно входят следующие этапы:

Сравнение программы со схемой алгоритма.

Визуальный контроль программы на экране дисплея или визуальное изучение распечатки программы и сравнение ее с оригиналом на программном бланке. Первые два этапа тестирования способны устранить больше количество ошибок, как синтаксических (что не так важно), так и семантических (что очень важно, так как позволяет исключить их трудоемкий поиск в процессе дальнейшей отладки).

Трансляция программы на машинных язык. На этом этапе выявляются синтаксические ошибки. Компиляторы с языков Си, Паскаль выдают диагностическое сообщение о синтаксических ошибках в листинге программы (листингом называется выходной документ транслятора, сопровождающий оттранслированную программу на машинном языке ? объектный модуль).



ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

Редактирование внешних связей и компоновка программы. На этапе редактирования внешних связей программных модуле программа-редактор внешних связей, или компоновщик задач, обнаруживает такие синтаксические ошибки, как несоответствие числа параметров в описании подпрограммы и обращении к ней, вызов несуществующей стандартной программы. например, 51 H вместо 51 N, различные длины общего блока памяти в вызывающем и вызываемом модуле и ряд других ошибок.

Выполнение программы. После устранения обнаруженных транслятором и редактором внешних связей (компоновщиком задач) синтаксических ошибок переходят к следующему этапу ? выполнению программы на ЭВМ на машинном языке: программа загружается в оперативную память, в соответствие с программой вводятся исходные данные и начинается счет. Проявление ошибки в процессе вода исходных данных или в процессе счета приводит к прерыванию счета и выдаче диагностического сообщения рабочей программы. Проявление ошибки дает повод для выполнения отладочных действий; отсутствие же сообщений об ошибках не означает их отсутствия в программе. План тестирования включает при этом проверку правильности полученных результатов для каких-либо допустимых значений исходных данных.

Тестирование программы. Если программа выполняется успешно, желательно завершить ее испытания тестированием при задании исходных данных, принимающих предельные для программы значения. а также выходящие за допустимые пределы значения на входе.

Контрольные примеры (тесты) ? это специально подобранные задачи, результаты которых заранее известны или могут быть определены без существенных затрат.

**Тема 12. Сопровождение ПО на стадии эксплуатации.**

Фаза эксплуатация и сопровождение - практическое использование программного изделия. Процедуры сопровождения регламентируют соответствующими стандартами для снижения затрат на этот вид деятельности.

Цель сопровождения ? обеспечить удовлетворение реальных потребностей пользователя.

Деятельность - работы по внесению изменений в программы и документацию для развития и совершенствования функциональных возможностей программного изделия и повышения его качества, по поддерживанию изделия в рабочем состоянии и по повышению эффективности его использования.

Сопровождение программного изделия в результате всегда дает изменение программного продукта. Штат, занятый сопровождением, должен полностью понимать программный продукт, в который необходимо вносить изменения. В некоторых случаях требуется обучение специалистов по сопровождению. В процессе эксплуатации и сопровождения создается Документ, отражающий историю развития проекта.

На ранних стадиях эксплуатации существует гарантийный период, когда разработчик сохраняет ответственность за исправление ошибок в программном продукте.

Окончание гарантийного периода фиксируется окончательной приемкой, критерием которой служит успешное выполнение всех приемных тестов и подтверждение выполнения всех требований пользователя.

Момент окончательной приемки соответствует формальной передаче программного изделия от разработчика к пользователю (обычно организации).

Сопровождение программного обеспечения связано с внесением изменений в течение всего времени использования программного изделия.

Причины, определяющие необходимость внесения изменений в изделие:

- наличие ошибок,
- изменение требования пользователя,
- появление более совершенных общесистемных программных средств или технических устройств,
- изменение организационной структуры, условий и методов работы пользователя.

**Тема 13. Управление разработкой ПО.**

Структура стандарта ГОСТ ISO/IEC 12207

Первый раздел описывает область применения данного стандарта.

1. Назначение. Устанавливает общую структуру процессов ЖЦ ПС, на которую можно ориентироваться в программной индустрии. Определяет процессы, работы и задачи, которые используются: при приобретении системы, содержащей программные средства, или отдельно поставляемого программного продукта; при оказании программной услуги, а также при поставке, разработке, эксплуатации и сопровождении программных продуктов.

2. Область распространения. Применяется при приобретении систем, программных продуктов и оказании соответствующих услуг; а также при поставке, разработке, эксплуатации и сопровождении программных продуктов и программных компонентов программно-аппаратных средств.

3. Адаптация. Определяет набор процессов, работ и задач, предназначенных для адаптации к условиям конкретных программных проектов. Процесс адаптации заключается в исключении неприменяемых в условиях конкретного проекта процессов, работ и задач.



ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

4. Соответствие. Соответствие стандарту определяется как выполнение всех процессов, работ и задач, выбранных из стандарта в процессе адаптации, для конкретного программного проекта. Выполнение процесса или работы считается завершенным, когда выполнены все требуемые для них задачи в соответствии с предварительно установленными в договоре требованиями.

5. Ограничения. Описывает архитектуру процессов жизненного цикла программных средств, но не определяет детали реализации или выполнения работ и задач, входящих в данные процессы

Под качеством ПС принято понимать совокупность характеристик, относящуюся к его способности удовлетворять установленным потребностям.

Общепринятой моделью, лежащей в основе оценки качества ПС, является модель, регламентированная в стандарте ISO/IEC 9126-1:2001 ?Информационная технология. Оценка программного продукта. Характеристики качества и руководства по их применению?. В соответствии с данным стандартом модель качества ПС представляет собой иерархическую структуру, состоящую из трех уровней.

1. Характеристики качества (цели) - то, что мы хотим видеть в ПС.

2. Атрибуты качества - свойства ПС, показывающие приближение к цели.

3. Метрики - количественные характеристики степени наличия атрибутов.

Верхний уровень данной модели представлен шестью основными характеристиками качества ПС. Это функциональность, надежность, практичность, эффективность, сопровождаемость и мобильность.

Функциональность  это способность ПС выполнять набор функций, удовлетворяющих заданным потребностям пользователей. Набор указанных функций определяется во внешнем описании ПС.

Надежность - это способность ПС безотказно выполнять определённые функции при заданных условиях в течение заданного периода времени с достаточно большой вероятностью. Надёжное ПС не исключает наличия в нём ошибок, важно, чтобы эти ошибки при практическом применении этого ПС в заданных условиях проявлялись достаточно редко.

Легкость применения  это характеристики ПС, которые позволяют минимизировать усилия пользователя по подготовке исходных данных, применению ПС и оценке полученных результатов.

Эффективность  это отношение уровня услуг, предоставляемых ПС пользователю при заданных условиях, к объему используемых ресурсов.

Сопровождаемость  это характеристики ПС, которые позволяют минимизировать усилия по внесению изменений для устранения в нем ошибок и по его модификации в соответствии с изменяющимися потребностями пользователей.

Мобильность  это способность ПС быть перенесенным из одной среды (окружения) в другую, в частности, с одного компьютера на другой.

Функциональность и надежность являются обязательными характеристиками качества ПС.

Разработка спецификации качества сводится к построению модели качества ПС. В этой модели должен быть перечень всех свойств, которыми требуется обеспечить ПС, и которые в совокупности образуют приемлемое качество ПС.

**Тема 14. Документация ПО.**

Документация на программное обеспечение ? это документы, сопровождающие программное обеспечение (ПО) ? программу или программный продукт. Эти документы описывают то, как работает программа и/или то, как её использовать.

Документирование это важная часть в разработке программного обеспечения, но часто ей уделяется недостаточно внимания.

Существует четыре основных типа документации на ПО:

архитектурная/проектная ? обзор программного обеспечения, включающий описание рабочей среды и принципов, которые должны быть использованы при создании ПО

техническая ? документация на код, алгоритмы, интерфейсы, API

пользовательская ? руководства для конечных пользователей, администраторов системы и другого персонала

маркетинговая

Архитектурная/проектная документация

Проектная документация обычно описывает продукт в общих чертах. Не описывая того, как что-либо будет использоваться, она скорее отвечает на вопрос ?почему именно так?? Например, в проектном документе программист может описать обоснование того, почему структуры данных организованы именно таким образом. Описываются причины, почему какой-либо класс сконструирован определённым образом, выделяются паттерны, в некоторых случаях даже даются идеи как можно будет выполнить улучшения в дальнейшем. Ничего из этого не входит в техническую или пользовательскую документацию, но всё это действительно важно для проекта.

Техническая документация



ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

Это именно то, что подразумевают под термином документация большинство программистов. При создании программы, одного лишь кода, как правило, недостаточно. Должен быть предоставлен некоторый текст, описывающий различные аспекты того, что именно делает код. Такая документация часто включается непосредственно в исходный код или предоставляется вместе с ним.

Подобная документация имеет сильно выраженный технических характер и в основном используется для определения и описания API, структур данных и алгоритмов.

Часто при составлении технической документации используются автоматизированные средства ? генераторы документации. Они получают информацию из специальным образом оформленных комментариев в исходном коде, и создают справочные руководства в каком-либо формате, например, в виде текста или HTML. Использование генераторов документации и документирующих комментариев многими программистами признаётся удобным средством, по различным причинам. В частности, при таком подходе документация является частью исходного кода, и одни и те же инструменты могут использоваться для сборки программы и одновременной сборки документации к ней. Это также упрощает поддержку документации в актуальном состоянии.

**Тема 15. Разработка и стандартизация информационных технологий.**

Стандарт в ИТ определяют как общепринятые требования, предъявляемые к техническому, программному, информационному и иному обеспечению, которые обеспечивают возможность стыковки и совместной работы систем. Различают:

стандарты де-юре (объявленные и принятые официально);

стандарты де-факто (не оформленные в виде документа, но применяемые на практике).

В области традиционного ?материального? производства давно сложилась система поддержки и согласования стандартов, а в области информационных технологий многое ещё предстоит сделать.

Популярное программное обеспечение не знает границ территорий и достаточно быстро распространяется по всему миру. Поэтому на национальном, межкорпоративном и международном уровнях всё чаще требуется использование общих (унифицированных) международных стандартов.

Важно отметить активное использование Интернета при разработке стандартов, в которой принимают участие многие организации и специалисты их различных стран. Это телеконференции с дискуссиями по наиболее важным вопросам; электронное голосование по утверждению проектов стандартов на разных стадиях разработки вплоть до статуса международного стандарта; организация очных семинаров и конференций; организация полного электронного архива, доступного по сети.

Развитие информационных технологий связано с национальными и международными стандартами. Международные стандарты создаются на основе шести принципов, определенных Всемирной торговой организацией (ВТО): открытость, прозрачность, непредвзятость и соблюдение консенсуса, эффективность и целесообразность, согласованность и нацеленность на развитие.

В России создаётся отечественная нормативная база в области информационных технологий. Для стандартизации информационных технологий, информационно-телекоммуникационных систем и проектирования информационных систем в стране создаются национальные стандарты и другие нормативные документы. Они определяют фундаментальные общие процедуры, положения и требования, которые могут быть использованы в различных предметных областях деятельности. Существуют специализированные организации: ВНИИстандарт, Гостехкомиссии России и др.

## 5. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине (модулю)

Самостоятельная работа обучающихся выполняется по заданию и при методическом руководстве преподавателя, но без его непосредственного участия. Самостоятельная работа подразделяется на самостоятельную работу на аудиторных занятиях и на внеаудиторную самостоятельную работу. Самостоятельная работа обучающихся включает как полностью самостоятельное освоение отдельных тем (разделов) дисциплины, так и проработку тем (разделов), осваиваемых во время аудиторной работы. Во время самостоятельной работы обучающиеся читают и конспектируют учебную, научную и справочную литературу, выполняют задания, направленные на закрепление знаний и отработку умений и навыков, готовятся к текущему и промежуточному контролю по дисциплине.

Организация самостоятельной работы обучающихся регламентируется нормативными документами, учебно-методической литературой и электронными образовательными ресурсами, включая:

Порядок организации и осуществления образовательной деятельности по образовательным программам высшего образования - программам бакалавриата, программам специалитета, программам магистратуры (утвержден приказом Министерства образования и науки Российской Федерации от 5 апреля 2017 года №301)

Письмо Министерства образования Российской Федерации №14-55-996ин/15 от 27 ноября 2002 г. "Об активизации самостоятельной работы студентов высших учебных заведений"



ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

Устав федерального государственного автономного образовательного учреждения "Казанский (Приволжский) федеральный университет"

Правила внутреннего распорядка федерального государственного автономного образовательного учреждения высшего профессионального образования "Казанский (Приволжский) федеральный университет"

Локальные нормативные акты Казанского (Приволжского) федерального университета

Цифровой образовательный ресурс "Программная инженерия" - https://stepik.org/course/65880

**6. Фонд оценочных средств по дисциплине (модулю)**

Фонд оценочных средств по дисциплине (модулю) включает оценочные материалы, направленные на проверку освоения компетенций, в том числе знаний, умений и навыков. Фонд оценочных средств включает оценочные средства текущего контроля и оценочные средства промежуточной аттестации.

В фонде оценочных средств содержится следующая информация:

- соответствие компетенций планируемым результатам обучения по дисциплине (модулю);

- критерии оценивания сформированности компетенций;

- механизм формирования оценки по дисциплине (модулю);

- описание порядка применения и процедуры оценивания для каждого оценочного средства;

- критерии оценивания для каждого оценочного средства;

- содержание оценочных средств, включая требования, предъявляемые к действиям обучающихся, демонстрируемым результатам, задания различных типов.

Фонд оценочных средств по дисциплине находится в Приложении 1 к программе дисциплины (модулю).

**7. Перечень литературы, необходимой для освоения дисциплины (модуля)**

Освоение дисциплины (модуля) предполагает изучение основной и дополнительной учебной литературы. Литература может быть доступна обучающимся в одном из двух вариантов (либо в обоих из них):

- в электронном виде - через электронные библиотечные системы на основании заключенных КФУ договоров с правообладателями;

- в печатном виде - в Научной библиотеке им. Н.И. Лобачевского. Обучающиеся получают учебную литературу на абонементе по читательским билетам в соответствии с правилами пользования Научной библиотекой.

Электронные издания доступны дистанционно из любой точки при введении обучающимся своего логина и пароля от личного кабинета в системе "Электронный университет". При использовании печатных изданий библиотечный фонд должен быть укомплектован ими из расчета не менее 0,5 экземпляра (для обучающихся по ФГОС 3++ - не менее 0,25 экземпляра) каждого из изданий основной литературы и не менее 0,25 экземпляра дополнительной литературы на каждого обучающегося из числа лиц, одновременно осваивающих данную дисциплину.

Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины (модуля), находится в Приложении 2 к рабочей программе дисциплины. Он подлежит обновлению при изменении условий договоров КФУ с правообладателями электронных изданий и при изменении комплектования фондов Научной библиотеки КФУ.

**8. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)**

Интернет-журнал по ИТ - http://www.rsdn.ru

Интернет-издание о высоких технологиях - http://www.cnews.ru/

Интернет-портал образовательных ресурсов по ИТ - http://www.intuit.ru

Команда MS Teams для курса - https://teams.microsoft.com/l/team/19%3a75845a58fc2b402db5b8ba1e93e9b268%40thread.tacv2/conversations?groupId=781

Компьютерная энциклопедия - http://www.computer-encyclopedia.ru

**9. Методические указания для обучающихся по освоению дисциплины (модуля)**



ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

| Вид работ | Методические рекомендации |
|---|---|
| лекции | В ходе лекционных занятий вести конспектирование учебного материала. Обращать внимание на категории, формулировки, раскрывающие содержание тех или иных явлений и процессов, научные выводы и практические рекомендации, положительный опыт в ораторском искусстве. Желательно оставить в рабочих конспектах поля, на которых делать пометки из рекомендованной литературы, дополняющие материал прослушанной лекции, а также подчеркивающие особую важность тех или иных теоретических положений. Задавать преподавателю уточняющие вопросы с целью уяснения теоретических положений, разрешения спорных ситуаций. |
| лабораторные работы | Целью проведения лабораторных работ является:<br>- систематизация, закрепление и расширение теоретических знаний и практических умений студента;<br>- приобретение опыта работы с литературой и другими источниками информации, умение обобщать и анализировать научную информацию, вырабатывать собственное отношение к проблеме;<br>-выработка умения применять информационные и компьютерные технологии для решения прикладных медицинских задач;<br>-развитие навыков овладения специализированным программным обеспечением;<br>- проведение глубокого анализа результатов собственных исследований и формирование содержательных выводов относительно качества полученных результатов |
| самостоя-тельная работа | Самостоятельная работа студентов (далее, СРС) ? планируемая учебная, научно-исследовательская работа студентов, выполняемая во внеаудиторное/(аудиторное) время по заданию и при методическом руководстве преподавателя, но без его непосредственного участия (возможно частичное непосредственное участие преподавателя при сохранении ведущей роли студентов).<br>Целью СРС является овладение фундаментальными знаниями, профессиональными умениями и навыками по профилю будущей специальности, опытом творческой, исследовательской деятельности, развитие самостоятельности, ответственности и организованности, творческого подхода к решению проблем учебного и профессионального уровней.<br>Задачи СРС:<br>систематизация и закрепление полученных теоретических знаний и практических умений студентов;<br>углубление и расширение теоретической подготовки;<br>формирование умений использовать нормативную, правовую, справочную документацию и специальную литературу;<br>развитие познавательных способностей и активности студентов: творческой инициативы, самостоятельности, ответственности и организованности;<br>формирование самостоятельности мышления, способностей к саморазвитию, самосовершенствованию и самореализации;<br>развитие исследовательских умений;<br>использование материала, собранного и полученного в ходе самостоятельных занятий на практических занятиях, при написании курсовых и выпускной квалификационной работ. |
| зачет | Готовиться к зачету необходимо последовательно, с учетом контрольных вопросов. Работу над темой можно считать завершенной, если вы сможете ответить на все контрольные вопросы и дать определение понятий по изучаемой теме. При подготовке необходимо выявлять наиболее сложные, дискуссионные вопросы, с тем, чтобы обсудить их с преподавателем на обзорных лекциях и консультациях. Нельзя ограничивать подготовку к зачету простым повторением изученного материала. Необходимо углубить и расширить ранее приобретенные знания за счет новых идей и положений. Результат по сдаче зачета объявляется студентам, вносится в экзаменационную ведомость. |

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



ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень программного обеспечения и информационных справочных систем, представлен в Приложении 3 к рабочей программе дисциплины (модуля).

## 11. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)

Материально-техническое обеспечение образовательного процесса по дисциплине (модулю) включает в себя следующие компоненты:

Помещения для самостоятельной работы обучающихся, укомплектованные специализированной мебелью (столы и стулья) и оснащенные компьютерной техникой с возможностью подключения к сети "Интернет" и обеспечением доступа в электронную информационно-образовательную среду КФУ.

Учебные аудитории для контактной работы с преподавателем, укомплектованные специализированной мебелью (столы и стулья).

Компьютер и принтер для распечатки раздаточных материалов.

Мультимедийная аудитория.

Компьютерный класс.

## 12. Средства адаптации преподавания дисциплины к потребностям обучающихся инвалидов и лиц с ограниченными возможностями здоровья

При необходимости в образовательном процессе применяются следующие методы и технологии, облегчающие восприятие информации обучающимися инвалидами и лицами с ограниченными возможностями здоровья:

- создание текстовой версии любого нетекстового контента для его возможного преобразования в альтернативные формы, удобные для различных пользователей;

- создание контента, который можно представить в различных видах без потери данных или структуры, предусмотреть возможность масштабирования текста и изображений без потери качества, предусмотреть доступность управления контентом с клавиатуры;

- создание возможностей для обучающихся воспринимать одну и ту же информацию из разных источников - например, так, чтобы лица с нарушениями слуха получали информацию визуально, с нарушениями зрения - аудиально;

- применение программных средств, обеспечивающих возможность освоения навыков и умений, формируемых дисциплиной, за счёт альтернативных способов, в том числе виртуальных лабораторий и симуляционных технологий;

- применение дистанционных образовательных технологий для передачи информации, организации различных форм интерактивной контактной работы обучающегося с преподавателем, в том числе вебинаров, которые могут быть использованы для проведения виртуальных лекций с возможностью взаимодействия всех участников дистанционного обучения, проведения семинаров, выступления с докладами и защиты выполненных работ, проведения тренингов, организации коллективной работы;

- применение дистанционных образовательных технологий для организации форм текущего и промежуточного контроля;

- увеличение продолжительности сдачи обучающимся инвалидом или лицом с ограниченными возможностями здоровья форм промежуточной аттестации по отношению к установленной продолжительности их сдачи:

- продолжительности сдачи зачёта или экзамена, проводимого в письменной форме, - не более чем на 90 минут;

- продолжительности подготовки обучающегося к ответу на зачёте или экзамене, проводимом в устной форме, - не более чем на 20 минут;

- продолжительности выступления обучающегося при защите курсовой работы - не более чем на 15 минут.

Программа составлена в соответствии с требованиями ФГОС ВО и учебным планом по направлению 10.03.01 "Информационная безопасность" и профилю подготовки "Безопасность компьютерных систем".



ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

*Приложение 2*
*к рабочей программе дисциплины (модуля)*
*Б1.В.ДВ.06.02 Технологии и стандарты разработки программного обеспечения*

**Перечень литературы, необходимой для освоения дисциплины (модуля)**

Направление подготовки: 10.03.01 - Информационная безопасность
Профиль подготовки: Безопасность компьютерных систем
Квалификация выпускника: бакалавр
Форма обучения: очное
Язык обучения: русский
Год начала обучения по образовательной программе: 2020

**Основная литература:**

1. Шишов, О. В. Современные технологии и технические средства информатизации : учебник / О.В. Шишов. - М. : ИНФРА-М, 2019. - 462 с. + Доп. материалы [Электронный ресурс; Режим доступа http://www.znanium.com]. - (Высшее образование: Бакалавриат). - ISBN 978-5-16-011776-8. - Текст : электронный. - URL: https://znanium.com/catalog/product/1002745 (дата обращения: 03.03.2020). - Режим доступа: по подписке.

2. Черников, Б. В. Управление качеством программного обеспечения : учебник / Б.В. Черников. - Москва : ИД 'ФОРУМ' : ИНФРА-М, 2019. - 240 с. - (Высшее образование: Бакалавриат). - ISBN 978-5-8199-0499-2. - Текст : электронный. - URL: https://znanium.com/catalog/product/1018037 (дата обращения: 03.03.2020). - Режим доступа: по подписке.

3. Гвоздева, В. А. Основы построения автоматизированных информационных систем : учебник / В. А. Гвоздева, И. Ю. Лаврентьева. - Москва : ФОРУМ : ИНФРА-М, 2020. - 318 с. - (Среднее профессиональное образование). - ISBN 978-5-8199-0705-4. - Текст : электронный. - URL: https://znanium.com/catalog/product/1066509 (дата обращения: 03.03.2020). - Режим доступа: по подписке.

**Дополнительная литература:**

1. Гагарина, Л. Г. Технология разработки программного обеспечения : учеб. пособие / Л.Г. Гагарина, Е.В. Кокорева, Б.Д. Сидорова-Виснадул ; под ред. Л.Г. Гагариной. - Москва : ИД 'ФОРУМ' : ИНФРА-М, 2019. - 400 с. - (Высшее образование: Бакалавриат). - ISBN 978-5-8199-0707-8. - Текст : электронный. - URL: https://znanium.com/catalog/product/1011120 (дата обращения: 03.03.2020). - Режим доступа: по подписке.

2. Кузнецов, А. С. Многоэтапный анализ архитектурной надежности и синтез отказоустойчивого программного обеспечения сложных систем : монография / А. С. Кузнецов, С. В. Ченцов, Р. Ю. Царев. - Красноярск: Сиб. федер. ун-т, 2013. - 143 с. - ISBN 978-5-7638-2730-9. - Текст : электронный. - URL: https://znanium.com/catalog/product/492347 (дата обращения: 03.03.2020). - Режим доступа: по подписке.

3. Царев, Р. Ю. Мультиверсионное программное обеспечение. Алгоритмы голосования и оценка надёжности [Электронный ресурс] : монография / Р. Ю. Царев, А. В. Штарик, Е. Н. Штарик. - Красноярск: Сиб. федер. ун-т, 2013. - 120 с. - ISBN 978-5-7638-2749-1. - Текст : электронный. - URL: https://znanium.com/catalog/product/492377 (дата обращения: 03.03.2020). - Режим доступа: по подписке.

4. Антамошкин, О. А. Программная инженерия. Теория и практика [Электронный ресурс] : учебник / О. А. Антамошкин. - Красноярск: Сиб. Федер. ун-т, 2012. - 247 с. - ISBN 978-5-7638-2511-4. - Текст : электронный. - URL: https://znanium.com/catalog/product/492527 (дата обращения: 03.03.2020). - Режим доступа: по подписке.



ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ

*Приложение 3*
*к рабочей программе дисциплины (модуля)*
*Б1.В.ДВ.06.02 Технологии и стандарты разработки программного обеспечения*

**Перечень информационных технологий, используемых для освоения дисциплины (модуля), включая перечень программного обеспечения и информационных справочных систем**

Направление подготовки: 10.03.01 - Информационная безопасность
Профиль подготовки: Безопасность компьютерных систем
Квалификация выпускника: бакалавр
Форма обучения: очное
Язык обучения: русский
Год начала обучения по образовательной программе: 2020

Освоение дисциплины (модуля) предполагает использование следующего программного обеспечения и информационно-справочных систем:
Операционная система Microsoft Windows 7 Профессиональная или Windows XP (Volume License)
Пакет офисного программного обеспечения Microsoft Office 365 или Microsoft Office Professional plus 2010
Браузер Mozilla Firefox
Браузер Google Chrome
Adobe Reader XI или Adobe Acrobat Reader DC
Kaspersky Endpoint Security для Windows
Учебно-методическая литература для данной дисциплины имеется в наличии в электронно-библиотечной системе "ZNANIUM.COM", доступ к которой предоставлен обучающимся. ЭБС "ZNANIUM.COM" содержит произведения крупнейших российских учёных, руководителей государственных органов, преподавателей ведущих вузов страны, высококвалифицированных специалистов в различных сферах бизнеса. Фонд библиотеки сформирован с учетом всех изменений образовательных стандартов и включает учебники, учебные пособия, учебно-методические комплексы, монографии, авторефераты, диссертации, энциклопедии, словари и справочники, законодательно-нормативные документы, специальные периодические издания и издания, выпускаемые издательствами вузов. В настоящее время ЭБС ZNANIUM.COM соответствует всем требованиям федеральных государственных образовательных стандартов высшего образования (ФГОС ВО) нового поколения.


ЭЛЕКТРОННЫЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННО-АНАЛИТИЧЕСКАЯ СИСТЕМА КФУ