Документ подписан простой электронной подписью
Информация о владельце:
ФИО: Дмитриев Николай Николаевич
Должность: Ректор
Дата подписания: 14.02.2025 08:09:33
Уникальный программный ключ:
f7c6227919e4cdbfb4d7b682991f8553b37cafbd

Министерство сельского хозяйства Российской Федерации
Иркутский государственный аграрный университет
имени А.А. Ежевского

Колледж автомобильного транспорта и агротехнологий

УТВЕРЖДАЮ:
Директор


_____ Н.Н. Бельков
«31» марта 2023 г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ПО ПРОФЕССИОНАЛЬНОМУ МОДУЛЮ

ПМ.05 Проектирование и разработка информационных систем

Специальность: 09.02.07 Информационные системы и программирование
(программа подготовки специалистов среднего звена)

Форма обучения: очная
3, 4 курсы; 5, 6, 7,8 семестры

Молодежный 2023

1. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

Фонд оценочных средств для промежуточной аттестации по профессиональному модулю ПМ.05 Проектирование и разработка информационных систем, включает:

- перечень компетенций с указанием этапов их формирования в процессе освоения образовательной программы;
- описание шкал оценивания;
- типовые контрольные задания или иные материалы, необходимые для оценки результатов обучения (промежуточной аттестации) по дисциплине, характеризующих этапы формирования компетенций и (или) для итогового контроля сформированности компетенций.

2.ПЕРЕЧЕНЬ КОМПЕТЕНЦИЙ С УКАЗАНИЕМ ЭТАПОВ ИХ ФОРМИРОВАНИЯ В ПРОЦЕССЕ ОСВОЕНИЯ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Рабочая программа профессионального модуля определяет перечень планируемых результатов обучения модулю, соотнесенных с планируемыми результатами освоения образовательной программы.

Код	**Наименование компетенции (планируемые результаты освоения ОП)**	**Планируемые результаты обучения по дисциплине, характеризующие этапы формирования компетенции**
ОК 01	Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам;	**Умения:** распознавать задачу и/или проблему в профессиональном и/или социальном контексте; анализировать задачу и/или проблему и выделять её составные части; определять этапы решения задачи; выявлять и эффективно искать информацию, необходимую для решения задачи и/или проблемы; составить план действия; определить необходимые ресурсы; владеть актуальными методами работы в профессиональной и смежных сферах; реализовать составленный план; оценивать результат и последствия своих действий (самостоятельно или с помощью наставника)
		Знания: актуальный профессиональный и социальный контекст, в котором приходится работать и жить; основные источники информации и ресурсы для решения задач и проблем в профессиональном и/или социальном контексте; алгоритмы выполнения работ в профессиональной и смежных областях; методы работы в профессиональной и смежных сферах; структуру плана для решения задач; порядок оценки результатов решения задач профессиональной деятельности

ОК 02	Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности;	**Умения:** определять задачи для поиска информации; определять необходимые источники информации; планировать процесс поиска; структурировать получаемую информацию; выделять наиболее значимое в перечне информации; оценивать практическую значимость результатов поиска; оформлять результаты поиска
		Знания: номенклатура информационных источников, применяемых в профессиональной деятельности; приемы структурирования информации; формат оформления результатов поиска информации
ОК 04	Эффективно взаимодействовать и работать в коллективе и команде;	**Умения:** организовывать работу коллектива и команды; взаимодействовать с коллегами, руководством, клиентами в ходе профессиональной деятельности
		Знания: психологические основы деятельности коллектива, психологические особенности личности; основы проектной деятельности
ОК 09	Пользоваться профессиональной документацией на государственном и иностранном языках	**Умения:** понимать общий смысл четко произнесенных высказываний на известные темы (профессиональные и бытовые), понимать тексты на базовые профессиональные темы; участвовать в диалогах на знакомые общие и профессиональные темы; строить простые высказывания о себе и о своей профессиональной деятельности; кратко обосновывать и объяснить свои действия (текущие и планируемые); писать простые связные сообщения на знакомые или интересующие профессиональные темы
		Знания: правила построения простых и сложных предложений на профессиональные темы; основные общеупотребительные глаголы (бытовая и профессиональная лексика); лексический минимум, относящийся к описанию предметов, средств и процессов профессиональной деятельности; особенности произношения; правила чтения текстов профессиональной направленности
Профессиональные компетенции		
ПК 5.1.	Собирать исходные данные для разработки проектной документации на информационную систему.	**Практический опыт:** Анализировать предметную область. Использовать инструментальные средства обработки информации. Обеспечивать сбор данных для анализа использования и функционирования информационной системы. Определять состав оборудования и программных средств разработки информационной системы. Выполнять работы предпроектной стадии.
		Умения: Осуществлять постановку задачи по обработке информации. Выполнять анализ предметной области. Использовать алгоритмы обработки информации для различных приложений. Работать с инструментальными средствами обработки информации. Осуществлять выбор модели построения информационной системы. Осуществлять выбор модели и средства построения информационной системы и программных средств.

		Знания: Основные виды и процедуры обработки информации, модели и методы решения задач обработки информации. Основные платформы для создания, исполнения и управления информационной системой. Основные модели построения информационных систем, их структуру, особенности и области применения. Платформы для создания, исполнения и управления информационной системой. Основные процессы управления проектом разработки. Методы и средства проектирования, разработки и тестирования информационных систем.
ПК 5.2.	Разрабатывать проектную документацию на разработку информационной системы в соответствии с требованиями заказчика.	**Практический опыт:** Разрабатывать проектную документацию на информационную систему.
		Умения: Осуществлять математическую и информационную постановку задач по обработке информации. Использовать алгоритмы обработки информации для различных приложений.
		Знания: Основные платформы для создания, исполнения и управления информационной системой. Национальную и международную систему стандартизации и сертификации и систему обеспечения качества продукции, методы контроля качества. Сервисно - ориентированные архитектуры. Важность рассмотрения всех возможных вариантов и получения наилучшего решения на основе анализа и интересов клиента. Методы и средства проектирования информационных систем. Основные понятия системного анализа.
ПК 5.3.	Разрабатывать подсистемы безопасности информационной системы в соответствии с техническим заданием.	**Практический опыт:** Управлять процессом разработки приложений с использованием инструментальных средств. Модифицировать отдельные модули информационной системы. Программировать в соответствии с требованиями технического задания.
		Умения: Создавать и управлять проектом по разработке приложения и формулировать его задачи. Использовать языки структурного, объектно-ориентированного программирования и языка сценариев для создания независимых программ. Разрабатывать графический интерфейс приложения.
		Знания: Национальной и международной системы стандартизации и сертификации и систему обеспечения качества продукции. Методы контроля качества объектно-ориентированного программирования. Объектно-ориентированное программирование. Спецификации языка программирования,

		принципы создания графического пользовательского интерфейса (GUI), файлового ввода-вывода, создания сетевого сервера и сетевого клиента. Файлового ввода-вывода. Создания сетевого сервера и сетевого клиента.
ПК 5.4.	Производить разработку модулей информационной системы в соответствии с техническим заданием.	**Практический опыт:** Разрабатывать документацию по эксплуатации информационной системы. Проводить оценку качества и экономической эффективности информационной системы в рамках своей компетенции. Модифицировать отдельные модули информационной системы.
		Умения: Использовать языки структурного, объектно-ориентированного программирования и языка сценариев для создания независимых программ. Решать прикладные вопросы программирования и языка сценариев для создания программ. Проектировать и разрабатывать систему по заданным требованиям и спецификациям. Разрабатывать графический интерфейс приложения. Создавать проект по разработке приложения и формулировать его задачи.
		Знания: Национальной и международной систему стандартизации и сертификации и систему обеспечения качества продукции, методы контроля качества. Объектно-ориентированное программирование. Спецификации языка программирования, принципы создания графического пользовательского интерфейса (GUI). Важность рассмотрения всех возможных вариантов и получения наилучшего решения на основе анализа и интересов клиента. Файлового ввода-вывода, создания сетевого сервера и сетевого клиента. Платформы для создания, исполнения и управления информационной системой.
ПК 5.5.	Осуществлять тестирование информационной системы на этапе опытной эксплуатации с фиксацией выявленных ошибок кодирования в разрабатываемых модулях информационной системы.	**Практический опыт:** Применять методики тестирования разрабатываемых приложений.
		Умения: Использовать методы тестирования в соответствии с техническим заданием.
		Знания: Особенности программных средств, используемых в разработке ИС.
ПК 5.6.	Разрабатывать техническую документацию на эксплуатацию информационной системы.	**Практический опыт:** Разрабатывать проектную документацию на информационную систему. Формировать отчетную документации по результатам работ. Использовать стандарты при оформлении программной документации.

		Умения: Разрабатывать проектную документацию на эксплуатацию информационной системы. Использовать стандарты при оформлении программной документации.
		Знания: Основные модели построения информационных систем, их структура. Использовать критерии оценки качества и надежности функционирования информационной системы. Реинжиниринг бизнес-процессов.
ПК 5.7.	Производить оценку информационной системы для выявления возможности ее модернизации.	**Практический опыт:** Проводить оценку качества и экономической эффективности информационной системы в рамках своей компетенции. Использовать критерии оценки качества и надежности функционирования информационной системы.
		Умения: Использовать методы и критерии оценивания предметной области и методы определения стратегии развития бизнес-процессов организации. Решать прикладные вопросы интеллектуальных систем с использованием статических экспертных систем, экспертных систем реального времени.
		Знания: Системы обеспечения качества продукции. Методы контроля качества в соответствии со стандартами.

3.ОПИСАНИЕ ШКАЛ ОЦЕНИВАНИЯ

При проведении промежуточной аттестации в колледже используются традиционные формы аттестации:

Элемент модуля	**Форма промежуточной аттестации**	**Шкала оценивания**
МДК.03.01 Моделирование и анализ программного обеспечения	Дифференцированный зачет	"отлично", "хорошо", "удовлетворительно", "неудовлетворительно"
МДК.03.02 Управление проектами	Экзамен	"отлично", "хорошо", "удовлетворительно", "неудовлетворительно"

4. ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ ИЛИ ИНЫЕ МАТЕРИАЛЫ, НЕОБХОДИМЫЕ ДЛЯ ОЦЕНКИ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ (ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ПО ПРОФЕССИОНАЛЬНОМУ МОДУЛЮ, ХАРАКТЕРИЗУЮЩИХ ЭТАПЫ ФОРМИРОВАНИЯ КОМПЕТЕНЦИЙ И (ИЛИ) ДЛЯ

ИТОГОВОГО КОНТРОЛЯ СФОРМИРОВАННОСТИ КОМПЕТЕНЦИИ

4.1. Примерный перечень вопросов к экзамену

МДК.05.01 Проектирование и дизайн информационных систем

1. Определение информационных систем.

Информационная система — взаимосвязанная совокупность средств, методов и персонала, используемых для хранения, обработки и выдачи информации в интересах достижения поставленной цели.

2. Информация и данные.

Информация — это результат преобразования и анализа данных. Отличие информации от данных состоит в том, что данные — это фиксированные сведения о событиях и явлениях, которые хранятся на определенных носителях, а информация появляется в результате обработки данных при решении конкретных задач.

3. Информационные системы и информационные технологии.

Информационные системы (ИС) — это система, построенная на базе компьютерной техники, предназначенная для хранения, поиска, обработки и передачи значительных объёмов информации, имеющая определённую практическую сферу применения.

Информационные технологии (ИТ) — это совокупность методов и программно-технических средств, объединённых в технологическую цепочку, обеспечивающую сбор, обработку, хранение, распределение и отображение информации с целью снижения трудоёмкости процессов использования информационных ресурсов.

4. Автоматизированная информационная система.

Автоматизированная информационная система (АИС) — это совокупность программных и аппаратных средств, предназначенных для хранения и (или) управления данными и информацией, а также для производства вычислений.

Основная цель АИС — хранение, обеспечение эффективного поиска и передачи информации по соответствующим запросам для наиболее полного удовлетворения информационных запросов большого числа пользователей

5. Виды обеспечения АИС.

Некоторые виды обеспечения автоматизированных информационных систем (АИС):

1. **Информационное обеспечение. Совокупность форм документов, классификаторов, нормативной базы и реализованных решений по объёмам, размещению и формам существования информации, применяемой в АИС.**
2. **Техническое обеспечение. Комплекс технических средств, предназначенных для работы информационной системы, а также соответствующая документация на эти средства и технологические процессы.**
3. **Программное обеспечение. Совокупность программ на носителях данных и программных документов, предназначенная для отладки, функционирования и проверки работоспособности АИС.**
4. **Организационное обеспечение. Совокупность документов, устанавливающих организационную структуру, права и обязанности пользователей, эксплуатирующего персонала АИС в условиях функционирования, проверки и обеспечения работоспособности системы.**

5. **Лингвистическое обеспечение. Совокупность средств и правил для формализации естественного языка, используемых при общении пользователей и эксплуатационного персонала АИС с комплексом средств автоматизации.**

6. **Математическое обеспечение. Совокупность математических методов, моделей и алгоритмов решения учётных задач.**

7. **Методическое обеспечение. Совокупность документов, описывающих технологию функционирования АИС, методы выбора и применения пользователями технологических приёмов для получения конкретных результатов.**

8. **Правовое обеспечение. Совокупность правовых норм, регламентирующих правовые отношения при функционировании АИС и юридический статус результатов её функционирования.**

6. Функции информационных систем.

Функции информационных систем можно разделить на функции управления и информационно-технологические.

Функции управления включают:

- **планирование и прогнозирование деятельности предприятия;**
- **нормирование производственной деятельности;**
- **учёт и отчётность;**
- **контроль производства;**
- **анализ производственной деятельности.**

Информационно-технологические функции информационной системы:

- **сбор сведений об управляемом объекте;**
- **регистрация данных;**
- **передача данных;**
- **ввод данных в ЭВМ;**
- **обработка данных;**
- **поиск данных;**
- **ведение баз данных;**
- **хранение данных;**
- **актуализация информации;**
- **копирование и тиражирование информации;**
- **предоставление информации (выдача производственных документов пользователю);**
- **защита данных. Эта функция обеспечивает сохранность и конфиденциальность информации.**

7. Стадии и этапы создания информационных систем.

Стадии создания информационных систем (ИС):

1. **Формирование требований к ИС. Обследование объекта и обоснование необходимости создания ИС, формирование требований пользователей к ИС, оформление отчёта о выполненной работе и тактико-технического задания на разработку.**

2. **Разработка концепции ИС. Изучение объекта автоматизации, разработка вариантов концепции ИС, удовлетворяющих требованиям пользователей, оформление отчёта и утверждение концепции.**

3. **Техническое задание. Разработка и утверждение технического задания на создание ИС.**

4. **Эскизный проект. Разработка предварительных проектных решений по системе и её частям, эскизной документации на ИС и её части.**

5. **Технический проект. Разработка проектных решений по системе и её частям, документации на ИС и её части, разработка и оформление документации на поставку комплектующих изделий.**

6. Рабочая документация. Разработка и адаптация программного обеспечения, рабочей документации.

7. Ввод в действие. Подготовка объекта автоматизации, подготовка персонала, комплектация ИС поставляемыми изделиями, строительно-монтажные и пусконаладочные работы, проведение предварительных, опытной и приёмочной эксплуатации.

8. Сопровождение ИС. Выполнение работ в соответствии с гарантийными обязательствами, послегарантийное обслуживание.

Каждая стадия состоит из этапов, а этапы, в свою очередь, состоят из видов работ.

8. Жизненный цикл информационных систем.

Жизненный цикл информационной системы (ИС) — это непрерывный процесс, начинающийся с момента принятия решения о создании информационной системы и заканчивающийся в момент полного изъятия её из эксплуатации.

Основные стадии жизненного цикла ИС:

1. Планирование и анализ требований (предпроектная стадия). Исследование и анализ существующей информационной системы, определение требований к создаваемой ИС, оформление технико-экономического обоснования и технического задания на разработку ИС.

2. Проектирование (техническое проектирование, логическое проектирование). Разработка в соответствии со сформулированными требованиями состава автоматизируемых функций (функциональная архитектура) и состава обеспечивающих подсистем (системная архитектура), оформление технического проекта ИС.

3. Реализация (рабочее проектирование, физическое проектирование, программирование). Разработка и настройка программ, наполнение баз данных, создание рабочих инструкций для персонала, оформление рабочего проекта.

4. Внедрение (тестирование, опытная эксплуатация). Комплексная отладка подсистем ИС, обучение персонала, поэтапное внедрение ИС в эксплуатацию по подразделениям объекта, оформление акта о приёмо-сдаточных испытаниях ИС.

5. Эксплуатация ИС (сопровождение, модернизация). Сбор рекламаций и статистики о функционировании ИС, исправление ошибок и недоработок, оформление требований к модернизации ИС и её выполнение.

Завершается жизненный цикл информационной системы выводом её из эксплуатации.

9. CASE-технологии и CALS-технологии.

CASE-технологии — это методология проектирования информационных систем, а также набор инструментальных средств, позволяющих в наглядной форме моделировать предметную область, анализировать эту модель на всех этапах разработки и сопровождения информационных систем и разрабатывать приложения в соответствии с информационными потребностями пользователей.

CALS-технологии — это технологии комплексной компьютеризации сфер промышленного производства, цель которых — унификация и стандартизация спецификаций промышленной продукции на всех этапах её жизненного цикла. В CALS-системах предусмотрены хранение, обработка и передача информации в компьютерных средах, оперативный доступ к данным в нужное время и в нужном месте.

Таким образом, CASE-технологии направлены на разработку информационных систем, а CALS-технологии — на поддержку жизненного цикла продукции, включая унификацию и стандартизацию проектной, технологической, производственной, маркетинговой и эксплуатационной документации.

10.Программное обеспечение для рабочих групп (groupware).

Программное обеспечение для рабочих групп (groupware) — это программное обеспечение, созданное с целью поддержки взаимодействия между людьми, совместно работающими над решением общих задач.

Некоторые примеры такого программного обеспечения:

- **eGroupWare. Программное обеспечение с открытым исходным кодом, которое может быть развёрнуто в облаке или установлено локально. Содержит календарь, адресную книгу, управление задачами, электронную почту, файловый менеджер, систему управления инцидентами, табель учёта рабочего времени, управление проектами и другие функции.**
- **SOGo. Бесплатный сервер groupware с открытым исходным кодом для небольших и средних команд. Система помогает компаниям управлять командами, экономить время и деньги, решать проблемы групп.**
- **Group Office. Инструмент CRM и groupware, написанный на PHP и JavaScript. Имеет такие функции, как электронная почта, календарь, обмен файлами, адресная книга, задачи, служба поддержки, отслеживание времени, синхронизация мобильных устройств и другие.**
- **Tiki. Платформа groupware с открытым исходным кодом для Linux, Windows, Mac, BSD. Используется в бизнесе, правительстве, некоммерческих организациях и у частных лиц по всему миру.**

11.CASE-средства проектирования баз данных.

CASE-средства проектирования баз данных — это методы и технологии, которые позволяют проектировать различные информационные системы (в частности, базы данных) и автоматизировать их создание.

Некоторые примеры CASE-средств проектирования баз данных:

- **ERwin (Logic Works). CASE-инструмент для создания концептуальных и логических схем баз данных. Позволяет редактировать различные наборы данных, представляя их в виде электронных таблиц, разрабатывать структуры баз данных, синхронизировать модели, скрипты и БД, настраивать шаблоны, выводить рабочую информацию в виде отчётов, строить диаграммы, отображающие различные процессы в системе и взаимосвязи между ними.**
- **S-Designor (SDP). Графический CASE-инструмент для проектирования структуры реляционных БД. Создаёт модели баз данных в два этапа — выстраивая концептуальную модель и затем преобразуя её в физическую. Инструмент позволяет проектировать базы данных под различные СУБД, в том числе под Oracle и MySQL.**
- **DataBase Designer (ORACLE). Интегрированная CASE-среда, которая позволяет анализировать предметную область создания БД, выполнять программирование и проектирование, проводить оценку и тестирование, осуществлять сопровождение, обеспечивать качество, управлять конфигурацией и проектом, разрабатывать и анализировать требования к информационной системе.**

12.Стадии и этапы процесса проектирования ИС.

Стадии и этапы процесса проектирования информационной системы (ИС):

1. **Стадия 1. Формирование требований к ИС. На начальной стадии проектирования выделяют следующие этапы работ: обследование объекта и обоснование необходимости создания ИС, формирование требований пользователей к ИС, оформление отчёта о выполненной работе и тактико-технического задания на разработку.**
2. **Стадия 2. Разработка концепции ИС. Включает изучение объекта автоматизации, проведение необходимых научно-исследовательских работ, разработку вариантов концепции ИС, удовлетворяющих требованиям пользователей, оформление отчёта и утверждение концепции.**

3. Стадия 3. Техническое задание. На этой стадии разрабатывают и утверждают техническое задание на создание ИС.

4. Стадия 4. Эскизный проект. На этом этапе разрабатывают предварительные проектные решения по системе и её частям, эскизную документацию на ИС и её части.

5. Стадия 5. Технический проект. На этой стадии разрабатывают проектные решения по системе и её частям, документацию на ИС и её части, разрабатывают и оформляют документацию на поставку комплектующих изделий, разрабатывают задания на проектирование в смежных частях проекта.

6. Стадия 6. Рабочая документация. На этой стадии разрабатывают рабочую документацию на ИС и её части, разрабатывают и адаптируют программы.

7. Стадия 7. Ввод в действие. На этом этапе подготавливают объект автоматизации, готовят персонал, комплектуют ИС поставляемыми изделиями, проводят строительно-монтажные и пусконаладочные работы, предварительные испытания, опытную эксплуатацию и приёмочные испытания.

8. Стадия 8. Сопровождение ИС. На этой стадии выполняют работы в соответствии с гарантийными обязательствами и послегарантийное обслуживание.

13.Объектно-ориентированная технология.

Объектно-ориентированная технология (ООТ) — это совокупность методик программирования, основанная на представлении программы в форме набора объектов, являющихся экземплярами определённых классов, образующих иерархию наследования.

В основе ООТ лежат две ключевые концепции — классы и объекты:

1. Класс — это шаблон или чертёж, который определяет свойства (характеристики) и методы (функции) будущих объектов. Например, класс «Собака» может содержать свойства «порода», «возраст» и методы «лаять» или «бегать».

2. Объект — это конкретный экземпляр класса, представляющий реальную сущность с уникальными значениями свойств. Например, объект «мой пёс» является конкретной собакой с породой «Лабрадор» и возрастом 5 лет.

ООТ используется, чтобы:

- **структурировать информацию и не допускать путаницы;**
- **точно определять взаимодействие одних элементов с другими;**
- **повышать управляемость программы;**
- **быстрее масштабировать код под различные задачи;**
- **лучше понимать написанное;**
- **эффективнее поддерживать готовые программы;**
- **внедрять изменения без необходимости переписывать весь код.**

Возможности ООТ поддерживает большинство популярных языков программирования, включая JavaScript, PHP, Python и другие.

14.Объектно-ориентированный анализ.

Объектно-ориентированный анализ (ООА) — это методология, при которой требования к системе формируются на основе понятий классов и объектов, составляющих словарь предметной области.

Главная идея — рассмотрение предметной области и логическое решение задач с точки зрения объектов (понятий и сущностей).

Основные методы объектно-ориентированного анализа:

1. Объектное моделирование. Развивает статическую структуру программной системы с точки зрения объектов. Определяет объекты, классы, в которые они могут быть сгруппированы, и отношения между объектами. Также определяет основные атрибуты и операции, характеризующие каждый класс.

2. Динамическое моделирование. После анализа статического поведения системы необходимо изучить её поведение во времени и внешних изменениях.

Динамическое моделирование описывает, как отдельный объект реагирует на события: внутренние, запускаемые другими объектами, или внешние, запускаемые внешним миром.

3. Функциональное моделирование. Заключительный компонент объектно-ориентированного анализа. Функциональная модель показывает процессы, которые выполняются внутри объекта, и то, как данные изменяются при перемещении между методами.

15.Объектно-ориентированное проектирование.

Объектно-ориентированное проектирование — это методология проектирования, которая решает задачи проектирования программной системы с использованием объектов и взаимодействий между ними.

В основе объектно-ориентированного проектирования лежит представление о том, что программную систему необходимо проектировать как совокупность взаимодействующих друг с другом объектов, рассматривая каждый объект как экземпляр определённого класса, причём классы образуют иерархию.

Некоторые концепции объектно-ориентированного проектирования:

- **Инкапсуляция. Скрытие информации от внешних сущностей.**
- **Наследование. Повторное использование методов работы с данными в различных условиях.**
- **Интерфейсы. Формальное описание методов и данных, используемое для взаимодействия между объектами.**
- **Полиморфизм. Возможность использования наследованных от объекта потомков в том же контексте, что и сам объект.**

Объектно-ориентированное проектирование применяется не только в программировании, но и в проектировании интерфейса пользователя, баз данных, баз знаний и даже компьютерной архитектуры.

16.Объектно-ориентированная реализация.

Объектно-ориентированное программирование (ООП) — это подход, при котором программа рассматривается как набор объектов, взаимодействующих друг с другом. У каждого есть свойства и поведение.

Основные элементы ООП:

- **Объекты. Это куски программного кода, которые описывают элемент с определённым набором характеристик и функций. Например, в видеоигре главный герой (персонаж) может выражаться отдельным объектом с различными характеристиками: здоровье, сила, выносливость, ловкость, урон.**
- **Методы. Это функции, описанные внутри того или иного класса или объекта. Они имеют прямое отношение к конкретному элементу и позволяют с ним взаимодействовать.**
- **Классы. Это «шаблон» для объекта, который описывает его свойства. Несколько похожих между собой объектов, например профили разных пользователей, будут иметь одинаковую структуру, а значит, принадлежать к одному классу.**

Основные принципы ООП:

- **Абстракция. У объекта должны быть определённые характеристики, которые отличают его от объектов иного типа. Так объект будет работать только с теми данными, которые ему нужны.**
- **Инкапсуляция. Объединяет данные и методы в классе, но скрывает их детали и делает невозможными неожиданные и случайные изменения в коде.**
- **Наследование. Позволяет объектам забирать свойства у своего шаблона. Классы также могут наследоваться от других классов.**
- **Полиморфизм. Позволяет применять для разных объектов одни и те же методы, которые ведут себя по-разному.**

17.Язык UML.

UML (Unified Modeling Language) — унифицированный язык моделирования. Это графический язык, который с помощью диаграмм и схем описывает разнообразные процессы и структуры.

UML применяется во многих областях:

- **В программировании. Чтобы наглядно видеть связи между классами и другими частями приложения или построить карту поведения пользователя на сайте.**
- **В дизайне. Чтобы создавать интерфейсы и понимать, как пользователи будут взаимодействовать с ними.**
- **В бизнесе. Чтобы визуально представлять, как работают бизнес-процессы или ведётся документооборот в организации.**

Особенность языка UML в том, что одинаковые связи и обозначения будут означать одно и то же в разных диаграммах. Это значит, что любой знающий UML человек легко поймёт любую схему, созданную на этом языке.

18. Современное состояние и тенденции развития ИС.

Некоторые современное состояние и тенденции развития информационных систем (ИС):

- **Наличие большого количества промышленно функционирующих баз данных большого объёма, содержащих информацию практически по всем видам деятельности общества.**
- **Создание технологий, обеспечивающих интерактивный доступ массового пользователя к этим информационным ресурсам через системы связи и передачи данных, объединённых в национальные, региональные и глобальные информационные сети.**
- **Расширение функциональных возможностей ИС. Реализация технологий создания и ведения гипертекстовых баз данных, включение в информационные системы экспертных систем, систем поддержки принятия решений и других технологических средств.**

Некоторые тенденции развития ИС:

- **Развитие искусственного интеллекта (ИИ). Машинное обучение, глубокое обучение и нейросети становятся всё более распространёнными и находят применение в разных областях.**
- **Интернет вещей (IoT). Подключение к сети различных устройств — от бытовой техники и домашней автоматики до промышленного оборудования и умных городов — позволяет собирать большие объёмы данных и управлять устройствами удалённо.**
- **Кибербезопасность. В условиях растущей угрозы кибератак и утечек данных компании и государства активно инвестируют в разработку и внедрение защитных механизмов, включая алгоритмы машинного обучения для обнаружения угроз, биометрическую аутентификацию и блокчейн-технологии для обеспечения целостности данных.**
- **Облачные технологии. Облачные сервисы предоставляют компаниям гибкие и масштабируемые решения для хранения данных, развёртывания приложений и вычислений, что позволяет сократить затраты на ИТ-инфраструктуру и повысить эффективность бизнес-процессов.**
- **Развитие квантовых вычислений. Эта технология обещает революционизировать область вычислений, обрабатывая большие объёмы данных в разы быстрее, чем современные суперкомпьютеры, и открывая новые возможности в области криптографии, моделирования искусственного интеллекта и многих других областях.**

4.2. Примерный перечень вопросов к зачету с оценкой

МДК.05.02 Разработка кода информационных систем

1. Структура CASE-средства. Структура среды разработки. Основные возможности.

Ответ: Структура CASE-средства включает следующие компоненты: Репозиторий. Является основой CASE-средства. Обеспечивает хранение версий проекта и его отдельных компонентов, синхронизацию поступления информации от различных разработчиков при групповой разработке, контроль метаданных на полноту и непротиворечивость.

Графические средства анализа и проектирования. 14 Обеспечивают создание и редактирование иерархически связанных диаграмм (DFD, ERD и др.), образующих модели ИС. Средства разработки приложений, включая языки 4GL и генераторы кодов. Средства конфигурационного управления.
Средства документирования.
Средства тестирования.
Средства управления проектом.
Средства реинжиниринга.
Структура среды разработки обычно включает в себя:

текстовый редактор;
компилятор и/или интерпретатор;
средства автоматизации сборки;
отладчик.
Основные возможности среды разработки:
использование встроенного многофайлового текстового редактора для работы с исходными текстами программ;
автоматическая диагностика выявленных при компиляции ошибок;
возможность параллельной работы над несколькими проектами;
минимум перекомпиляции: ей подвергаются только редактировавшиеся модули;
возможность загрузки отлаживаемой программы в имеющиеся средства отладки и работы с ними без выхода из оболочки;
возможность подключения к оболочке практически любых программных средств.

2. Основные инструменты среды для создания, исполнения и управления информационной системой. Выбор средств обработки информации

Ответ: Основные инструменты среды для создания, исполнения и управления информационной системой включают:
языки программирования;
системы управления базами данных (СУБД);
операционные системы;
средства разработки приложений (IDE);
средства управления версиями кода.
Выбор средств обработки информации зависит от требований проекта и целей разработки.

Некоторые инструменты для обработки данных:

Astera. Инструмент интеграции данных без кода, предназначенный для пользователей с любым уровнем технических возможностей. Предлагает комплексное управление данными: от извлечения до интеграции, хранения данных и даже управления API.

Инструменты для обработки информации в информационных технологиях. К ним относятся электронно-вычислительные машины (ЭВМ) различных классов и типов. Например, проблемно-ориентированные ЭВМ служат для решения задач, связанных с управлением технологическими объектами, регистрацией, накоплением и обработкой относительно небольших объёмов данных, выполнением расчётов по относительно несложным алгоритмам.
Также для обработки информации могут использоваться интегрированные среды разработки, которые предоставляют средства для создания, отладки и управления кодом в одном интерфейсе. Они облегчают разработку независимых программ путём предоставления инструментов, таких как редакторы кода, отладчики и компиляторы.

3. Организация работы в команде разработчиков. Система контроля версий: совместимость, установка, настройка

Ответ: Организация работы в команде разработчиков включает распределение ролей, включая руководителя проекта, архитектора, разработчика и тестировщика.

Система контроля версий помогает сотрудникам одновременно работать над проектом, отслеживать, кто какие изменения внёс, и избегать конфликтов. Существует три основных разновидности архитектур систем контроля версий: локальная, централизованная и распределённая. 3 В локальной системе файлы хранятся на одном устройстве, в централизованной — на общем сервере, а в распределённой — в общем облачном хранилище и на локальных устройствах участников команды.

Выбор подходящей системы контроля версий зависит от специфики проекта и команды. Некоторые популярные системы:

Git. Распределённая система, подходит для проектов любого масштаба.
Subversion (SVN). Централизованная система, удобна для небольших команд и проектов.
Mercurial. Распределённая система, схожая с Git, но с более простым интерфейсом.
Установка и настройка системы контроля версий зависят от используемой программы. Например, для установки Git нужно перейти на сайт git-scm.com и выбрать способ установки под свою операционную систему. Для Windows можно скачать загрузочный файл и поставить Git как обычную программу. На macOS и Linux программу удобно устанавливать через программу «Терминал».
Организация работы команды в системах контроля версий требует тщательного планирования и соблюдения лучших практик. Выбор подходящей системы, настройка рабочего процесса и автоматизация задач помогут команде работать эффективно и продуктивно.

4. Обеспечение кроссплатформенности информационной системы

Ответ: Для обеспечения кроссплатформенности информационной системы можно предпринять следующие шаги:
Использовать кроссплатформенные языки программирования. Например, C++, Free Pascal, FreeBASIC, PureBasic обеспечивают кроссплатформенность на уровне

компиляции, предлагая компиляторы для разных платформ. При их использовании нет необходимости переписывать весь код — достаточно скорректировать отдельные куски.
Сделать кроссплатформенный интерфейс. Чтобы избежать искажений стандартных элементов, можно либо создать интерфейс на основе стандартной схемы, либо сделать его самоадаптирующимся.
Вести разработку в кроссплатформенных средах. Специальные программы-редакторы облегчают этот процесс, позволяют свести к минимуму количество ошибок, дают подсказки, тестируют и компилируют код.
Использовать протестированные фреймворки, CSS-хаки и универсальные элементы кода.
Также можно построить систему с многоуровневой архитектурой, которая предполагает использование серверов приложений. В этом случае пользователи не будут устанавливать на своих устройствах дополнительное программное обеспечение, а будут получать доступ к системе через браузер.

5. Сервисно-ориентированные архитектуры.

Ответ: Се́рвис-ориенти́рованная архитекту́ра (СОА, англ. service-oriented architecture - SOA) — модульный подход к разработке программного обеспечения, базирующийся на обеспечении удаленного по стандартизированным протоколам использования распределённых, слабо связанных [en] легко заменяемых компонентов (сервисов) со стандартизированными интерфейсами.

6. Интегрированные среды разработки для создания независимых программ.

Ответ: Некоторые интегрированные среды разработки для создания независимых программ:

Универсальные: Visual Studio, NetBeans, Eclipse, KDevelop, Xcode, Geany, MonoDevelop, Aptana, Open Watcom, Komodo, Kylix.

Для C/C++: Anjuta, Borland C++, C++ Builder, Code::Blocks, CodeLite, wxDev-C++, Pelles C, Oracle Solaris Studio, Qt Creator, Ultimate++, Microsoft QuickC.

Для Python: Boa Constructor, Eric Python IDE, Geany, NetBeans IDE, PyScripter.

Для Java: WebLogic, BlueJ, DrJava, Greenfoot, JCreator, JDeveloper, IntelliJ IDEA, JBuilder, JGRASP.

Для PHP: Aptana, Delphi for PHP, Eclipse PDT, Zend Studio, NuSphere PhpED, PHP expert editor.

Для Flash: Flash IDE, Flash Builder, FDT, IntelliJ IDEA, PrimalScript.
Также есть среды визуальной разработки, которые включают в себя возможность наглядного редактирования интерфейса программы.

7. Особенности объектно-ориентированных и структурных языков программирования.

Ответ: Особенности объектно-ориентированных языков программирования (ООП):

Модульность. Программа разбивается на отдельные модули, которые могут взаимодействовать друг с другом. Каждый модуль представляет собой отдельный объект со своим состоянием и поведением.
Абстракция. Позволяет абстрагироваться от деталей реализации и концентрироваться на основных понятиях и взаимодействии объектов.
Наследование. Создаётся иерархия классов, где классы-наследники могут наследовать свойства и методы от родительских классов. Это позволяет избежать дублирования кода и делает программу более гибкой и расширяемой.
Полиморфизм. Позволяет использовать один и тот же интерфейс для работы с разными типами объектов.
Инкапсуляция. Детали реализации объектов скрываются, а предоставляются только необходимые интерфейсы для их взаимодействия. Это увеличивает безопасность программы и упрощает её использование и поддержку.
Особенности структурных языков программирования: в их основе лежит представление программы в виде иерархической структуры блоков. Любая программа состоит из трёх базовых управляющих структур: последовательность, ветвление, цикл.

8. Разработка сценариев с помощью специализированных языков.

Ответ: Разработка сценариев с помощью специализированных языков предполагает создание последовательности команд (алгоритма), необходимой для выполнения тех или иных задач. Для этого используются различные языки программирования, например:
JavaScript. Один из наиболее популярных языков сценариев. На нём создают веб- или мобильные приложения, инструменты командной строки, сетевые приложения реального времени.
Python. Разработчики используют скрипты Python для автоматизации ежедневных задач, создания отчётов, обеспечения безопасности и т. д..
PHP. 13 Язык поддерживает несколько типов баз данных, обеспечивает эффективную производительность веб-сайтов с интенсивным трафиком.
R. Язык сценариев используется для статистических вычислений и графики.
Также существуют командно-сценарные языки, которые применяются для выполнения различных действий в пределах операционных систем, например, работы с консолью.
Для создания сценариев необходимо выучить выбранный язык программирования, определиться с целью и написать программный код.

9. Обоснование и осуществление выбора модели построения или модификации информационной системы.

Ответ: Обоснование и осуществление выбора модели построения или модификации информационной системы включает в себя анализ требований к проекту и целей разработки.
Некоторые модели, которые могут быть выбраны:
Водопадная модель. Применяется, когда требования к проекту чётко определены и не предполагают изменений в ходе разработки. Например, создание простого веб-сайта.
Инкрементальная модель. Используется, если требования могут быть выделены на этапы и разработка может вестись частями. Например, создание пошагового обновления программного обеспечения.
Спиральная модель. Подходит для проектов, требующих постоянного обновления и добавления новых функциональностей. Например, разработка сложных систем безопасности.

10. Обоснование и осуществление выбора средства построения информационной системы и программных средств.

Ответ: Обоснование и осуществление выбора средства построения информационной системы и программных средств включает в себя выбор следующих элементов:
Язык программирования. Например, Python для быстрой разработки прототипов или Java для кроссплатформенных приложений.
Фреймворк. Например, Django для веб-разработки на Python или Spring для разработки приложений на Java.
Библиотеки. Например, библиотека TensorFlow для машинного обучения в Python или jQuery для упрощения веб-разработки.
Инструменты. Например, среда разработки Visual Studio Code для написания и отладки кода.
Выбор средств зависит от требований проекта и целей разработк

11. Построение архитектуры проекта. Шаблон проекта.

Ответ: Многоуровневая архитектура;

«Клиент-сервер»;

«Каналы и фильтры»;

«SOA»;

«Издатель-подписчик»;

«Общие данные»;

«Одноранговая сеть»;

«Брокер сервисов»

12. Определение конфигурации информационной системы. Выбор технических средств.

Ответ: Определение конфигурации информационной системы включает настройку её под особенности области внедрения. Некоторые действия, которые для этого выполняются:
изменение объектной модели;
определение авторизаций пользователей;
настройка интерфейса;
создание типовых объектов данных — справочников, шаблонов, отчётов, процессов и т. д.;
настройка вариантов развёртывания и взаимодействия программных компонентов, определение параметров системных служб, сервисов и т. д..
Выбор технических средств для информационной системы включает определение технического обеспечения — комплекса технических средств, предназначенных для работы системы, а также соответствующей документации на эти средства и технологические процессы.

В состав технического обеспечения входят:
компьютеры любых моделей;
устройства сбора, накопления, обработки, передачи и вывода информации;
устройства передачи данных и линий связи;
оргтехника и устройства автоматического съёма информации;
эксплуатационные материалы и др..
Сложились две основные формы организации технического обеспечения: централизованная и частично или полностью децентрализованная. Централизованное техническое обеспечение базируется на использовании в информационной системе больших ЭВМ и вычислительных центров. Децентрализация технических средств предполагает реализацию функциональных подсистем на персональных компьютерах непосредственно на рабочих местах.

13.Формирование репозитория проекта, определение уровня доступа в системе контроля версий. Распределение ролей.

Ответ: Формирование репозитория проекта, определение уровня доступа в системе контроля версий и распределение ролей включают следующие аспекты:
Репозиторий — хранилище информации, связанной с проектом разработки программного продукта в течение всего его жизненного цикла. Выделяют три класса уровней репозиториев: модельный, программного интерфейса и окружения.
Системы управления версиями — программное обеспечение, предназначенное для отслеживания изменений между различными версиями файловых документов и разделения доступа к ним. При передаче файлов под управление такой системой она создаёт для них собственный репозиторий, с помощью которого отслеживает и хранит все изменения.
Определение уровня доступа. Системы контроля версий позволяют хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение. Некоторые системы управления версиями дают возможность заблокировать файл в хранилище, что обеспечивает исключительный доступ только тому пользователю, который работает с документом.
Распределение ролей. В системе контроля версий можно распределить роли пользователей, разрешив или запретив чтение или изменение информации в зависимости от того, кто запрашивает это действие.

14.Настройки среды разработки

Ответ: Некоторые настройки среды разработки:
Сохранять все файлы при выходе без подтверждения. При выходе из среды разработки все произведённые изменения будут автоматически сохранены.
Закрывать все окна при закрытии проекта. При закрытии проекта будут закрыты все окна.
Отображать папку проекта в заголовке окна. Позволяет отображать относительный путь к папке текущего проекта в заголовке главного окна.
Отображать имя функции в строке состояния. Позволяет отображать имя функции или метода, в пределах которой находится каретка текстового редактора.
Открывать новое окно среды разработки. Опции устанавливают необходимость открывать новое окно среды разработки при создании нового проекта, открытии другого проекта или открытии проекта из списка последних.
Автоматическое сохранение настроек каждые N секунд. Позволяет в фоновом режиме выполнять сохранение изменений настроек среды разработки и редактора форм.

Автосохранение списка открытых окон среды разработки. Позволяет в фоновом режиме запоминать проекты и файлы, открытые в окнах среды разработки. Запомненные окна могут быть автоматически открыты при старте программы. Сохранять расположение стыкующихся окон при выходе. При выходе из среды разработки запоминается расположение стыкующихся окон, которое будет использовано при последующих открытиях окна среды разработки.

15.Мониторинг разработки проекта. Сохранение версий проекта

Ответ: Мониторинг разработки проекта позволяет получать фактические данные о ходе выполнения проекта, сопоставлять их с плановыми характеристиками и выявлять отклонения, на основании которых принимаются управленческие решения.
Для сохранения версий проекта используются системы контроля версий (СКВ). 24 Они хранят все версии проекта и обеспечивают к ним доступ.
Некоторые задачи СКВ:
Отслеживание изменений. СКВ позволяют точно отслеживать каждое изменение, вносимое в исходный код или другие файлы. Это помогает понять, какие изменения были сделаны, кто их сделал и когда.
Восстановление предыдущих версий. Система контроля версий сохраняет историю изменений, что позволяет разработчикам легко возвращаться к предыдущим версиям кода. Если что-то пошло не так или новые изменения оказались нежелательными, можно легко откатиться к предыдущей стабильной версии.
Слияние изменений. Когда несколько разработчиков работают над одним проектом, СКВ позволяет сливать их изменения в одну общую версию. Это упрощает совместную работу, поскольку каждый разработчик может работать над своей частью проекта, а затем объединить изменения без конфликтов.
Контроль доступа. СКВ позволяет определить, кто имеет доступ к проекту и какие права у разных пользователей. Это обеспечивает безопасность и конфиденциальность проекта, поскольку только авторизованные лица могут вносить изменения и просматривать код.
Одной из популярных систем контроля версий является Git.

Критерии экзамена

Отметка «5 (отлично)» ставится в случае:

знания, понимания, глубины усвоения обучающимся всего объема программного материала;

творчески применять полученные знания в незнакомой ситуации;

отсутствия ошибок и недочётов при воспроизведении изученного материала, при устных ответах, устранения отдельных неточностей с помощью дополнительных вопросов педагога;

соблюдения культуры письменной и устной речи, правил оформления письменных работ.

Отметка «4 (хорошо)» ставится в случае:

знания всего изученного материала;

умения выделять главные положения в изученном материале, на основании фактов и примеров обобщать, делать выводы, устанавливать межпредметные и внутрипредметные связи, применять полученные знания на практике;

наличие незначительных (негрубых) ошибок при воспроизведении изученного материала;

соблюдения основных правил культуры письменной и устной речи, правил оформления письменных работ.

Отметка «3 (удовлетворительно)» ставится в случае:

- знания и усвоения материала на уровне минимальных требований программы, затруднения при самостоятельном воспроизведении, необходимости незначительной помощи учителя;

умения работать на уровне воспроизведения, затруднения при ответах на видоизменённые вопросы;

наличия 1-2 грубых ошибок, нескольких негрубых при воспроизведении изученного материла;

незначительного несоблюдения основных правил культуры письменной и устной речи, правил оформления письменных работ.

Отметка «2 (неудовлетворительно)» ставится в случае:

знания и усвоения учебного материала на уровне ниже минимальных требований программы;

отсутствия умения работать на уровне воспроизведения, затруднения при ответах на стандартные вопросы;

наличия нескольких грубых ошибок, большого числа негрубых при воспроизведении изученного материала;

- значительного несоблюдения основных правил культуры письменной и устной речи, правил оформления письменных работ.

Отметка «1 (неудовлетворительно)» ставится в случае:

- отказ обучающегося от ответа, выполнения работы, теста, отсутствие выполненного (в том числе, домашнего) задания.

При выставлении отметок необходимо учитывать классификацию ошибок и их количество:

грубые ошибки;

однотипные ошибки;

негрубые ошибки;

недочеты.

К грубым ошибкам следует относить:

незнание определения основных понятий, правил,

неумение выделять главное в ответе;

неумение делать выводы и обобщения;

неумение пользоваться первоисточниками, учебником и справочником.

К **однотипным ошибкам** относятся ошибки на одно и то же правило.

К **негрубым ошибкам** следует относить:

неточность формулировок, определений, понятий, правил, вызванная неполнотой охвата основных признаков определяемого понятия или замена 1-2 из этих признаков второстепенными;

нерациональные методы работы с учебной и справочной литературой

4.2. Примерный перечень вопросов к зачету

МДК.05.03 Тестирование информационных систем

1. Организация тестирования в команде разработчиков. Виды и методы тестирования (в том числе автоматизированные).

Ответ: Организация тестирования в команде разработчиков включает разработку стратегии и плана тестирования. В него входят выбор методов (ручное, автоматизированное, тестирование на реальных устройствах и другие), анализ потенциальных рисков и планирование ресурсов (кто будет тестировать продукт, какое оборудование и инструменты можно использовать, сколько времени займёт тестирование, к какому сроку оно должно быть закончено).

Виды тестирования:

По характеру сценариев. Тестирование позитивных сценариев проверяет, как должна работать программа в нормальных условиях. Тестирование негативных сценариев проверяет, как программа будет вести себя в неожиданных ситуациях.

По степени автоматизации. Ручное тестирование — это проверка программного обеспечения вручную, без использования автоматизированных инструментов. Тестировщик взаимодействует с программой как обычный пользователь. Автоматизированное тестирование — это проверка программного обеспечения с использованием специальных программных инструментов, которые выполняют тесты автоматически, без участия человека.

По объектам тестирования. Функциональное тестирование проверяет соответствие программы или системы заранее определённым функциональным требованиям и ожиданиям. Нефункциональное тестирование проверяет нефункциональные аспекты программы — производительность, безопасность, надёжность, масштабируемость и совместимость.

Некоторые методы тестирования:

Исследовательское тестирование. Тестировщик не имеет заранее определённых тестовых сценариев и пытается интуитивно исследовать возможности программного продукта и обнаружить и зафиксировать неизвестные ошибки.

Интеграционное тестирование. Используется для проверки корректности совместной работы компонентов программного продукта.

Нагрузочное тестирование. Предназначено для проверки работоспособности программного продукта при предельной входной нагрузке.

Регрессионное тестирование. Применяется при внесении изменений в программное обеспечение с целью проверки корректности работы компонентов системы, которые потенциально могут взаимодействовать с изменённым компонентом.

Приёмочное тестирование. Представляет собой функциональные испытания, которые должны подтвердить то, что программный продукт соответствует требованиям и ожиданиям пользователей и заказчиков.

2. Тестовые сценарии, тестовые варианты. Оформление результатов тестирования.

Ответ: Для разработки тестовых сценариев и выполнения тестов используются системы
управления тестированием, существенно повышающие производительность тест-дизайнеров и тестировщиков, а также обеспечивающие видимость уровня качества приложений среди всех участников проекта.
Тестовые сценарии неразрывно связаны с требованиями, изменения в которых

должны своевременно отражаться в тестовой документации, что позволяет сделать система управления жизненным циклом разработки приложений, при помощи механизма трассировок.
При выполнении теста тестировщик отмечает результат прохождения одного шага или всего тестового сценария, прикрепляет обнаруженные ошибки и другую вспомогательную информацию: скриншоты, дампы, логи и т.п.
Тестовые сценарии удобно объединять в тест-планыпо назначению:
· тестирование релиза, то есть очередной версии продукта;
· тестирование развертывания;
· тестирование удобства использования;
· конфигурационное тестирование;
· тестирование безопасности и т.п.
Зачастую ручное тестирование превращается в рутину и занимает значительное время, что отрицательно сказывается на скорости выпуска релизов.
Автоматизация тестирования позволяет:
· высвободить ресурсы для проведения более сложных видов тестирования;
· снизить количество дефектов, доходящих до стадии контроля качества;
· ускорить выпуск релизов.
Сведение результатов автоматических и ручных тестов в системе управления качеством, позволяет всем участникам проекта видеть уровень качества очередного релиза, контролировать его изменение и опираться на эту информацию при планировании своей работы.
Получение результатов тестирования и их анализ.
Получение результатов тестирования и их анализ.
Получение результатов тестирования напрямую зависит от средств тестирования. В моем случае это был встроенный в 1С механизм отладки и система контроля ошибок.

3. Инструментарии анализа качества программных продуктов в среде разработке.

Ответ: Некоторые инструменты для анализа качества программных продуктов в среде разработки:

Метрики. С их помощью можно дать количественную или качественную оценку качества программного обеспечения. Различают три типа метрик: интервальная шкала с относительными величинами физических показателей (время наработки на отказ, вероятность ошибки, объём информации и другие), порядковая шкала, позволяющая ранжировать характеристики путём сравнения с опорными значениями, и номинальная, или категорированная шкала, определяющая наличие рассматриваемого свойства или признака у объекта без учёта градаций по этому признаку.
MicrosoftTestManager. Инструментарий Microsoft, предназначенный для управления жизненным циклом тестирования программного обеспечения.
LabManagement. Диспетчер виртуальной среды тестирования.
Также к инструментам анализа качества программных продуктов относят тестирование. 23 Оно позволяет определить отклонения в реализации функциональных требований, обнаружить ошибки в выполнении программ и исправить их как можно раньше в процессе выполнения проекта. На протяжении всего жизненного цикла разработки ПО применяются различные типы тестирования:

Модульное тестирование. Предназначено для проверки правильности функционирования методов классов ПО.

Исследовательское тестирование. Тестировщик не имеет заранее определённых тестовых сценариев и пытается интуитивно исследовать возможности программного продукта и обнаружить и зафиксировать неизвестные ошибки.

Интеграционное тестирование. Используется для проверки корректности совместной работы компонентов программного продукта.

Функциональное тестирование. Предполагает проверку конкретных требований к ПО и проводится после добавления к системе новых функций.

Нагрузочное тестирование. Предназначено для проверки работоспособности программного продукта при предельной входной нагрузке.

Регрессионное тестирование. Применяется при внесении изменений в программное обеспечение с целью проверки корректности работы компонентов системы, которые потенциально могут взаимодействовать с изменённым компонентом.

Комплексное тестирование. Предназначено для тестирования функциональных и нефункциональных требований всей системы программного продукта.

Приёмочное тестирование. Представляет собой функциональные испытания, которые должны подтвердить то, что программный продукт соответствует требованиям и ожиданиям пользователей и заказчиков.

4. Обработка исключительных ситуаций. Методы и способы идентификации сбоев и ошибок.

Ответ: Обработка исключительных ситуаций — механизм языков программирования, предназначенный для описания реакции программы на ошибки времени выполнения и другие возможные проблемы (исключения), которые могут возникнуть при выполнении программы и приводят к невозможности дальнейшей отработки программой её базового алгоритма.

Исключительные ситуации можно разделить на два основных типа:

Синхронные. Могут возникнуть только в определённых, заранее известных точках программы. Например, ошибка чтения файла или коммуникационного канала, нехватка памяти.

Асинхронные. Могут возникать в любой момент времени и не зависеть от того, какую конкретно инструкцию программы выполняет система. Типичные примеры таких исключений: аварийный отказ питания или поступление новых данных. Обработка исключительных ситуаций самой программой заключается в том, что при возникновении исключительной ситуации управление передаётся заранее определённому обработчику — блоку кода, процедуре, функции, которые выполняют необходимые действия.

Для идентификации сбоев и ошибок могут использоваться различные методы и способы, например:

Детерминированное тестирование. Наиболее трудоёмкий и детализированный метод, который требует многократного выполнения программы на компьютере с использованием определённых, специально подобранных тестовых наборов данных.

Стохастическое тестирование. Применяется для комплексного тестирования, когда невозможно перебрать все комбинации исходных данных и проконтролировать результаты функционирования каждого из них.

Тестирование в реальном масштабе времени. Применяется к программам, которые предназначены для работы в системах реального времени. В процессе этого тестирования проверяются результаты обработки исходных данных с учётом времени

их поступления, длительности и приоритетности обработки, динамики использования памяти и взаимодействия с другими программами.

5. Выявление ошибок системных компонентов.

Ответ: Выявление ошибок системных компонентов — важный этап в процессе разработки программного обеспечения. Ошибки могут возникать на разных стадиях разработки, начиная от определения требований и проектирования, до написания кода и подготовки документации.

Некоторые методы диагностики ошибок системных компонентов:

Статический анализ. Предполагает изучение исходных или проектных спецификаций, текстов кодов программ, эксплуатационной документации и т.п..

Динамический анализ. В рамках этого метода выявляются дефекты и ошибки, которые отражаются на функциональном назначении, рисках и качестве применения версии программных средств.

Использование автоматизированных инструментов тестирования. Они помогают обнаруживать проблемы, анализируя контролируемое выполнение программы на конечном множестве тестовых данных и результаты этого выполнения.

Некоторые виды тестирования для выявления ошибок системных компонентов:

Интеграционное тестирование. Используется для проверки корректности совместной работы компонентов программного продукта.

Функциональное тестирование. Предполагает проверку конкретных требований к ПО и проводится после добавления к системе новых функций.

Нагрузочное тестирование. Предназначено для проверки работоспособности программного продукта при предельной входной нагрузке.

Регрессионное тестирование. Применяется при внесении изменений в программное обеспечение с целью проверки корректности работы компонентов системы, которые потенциально могут взаимодействовать с изменённым компонентом.

6. Реинжиниринг бизнес-процессов в информационных системах.

Ответ: Реинжиниринг бизнес-процессов в информационных системах — это целостное и системное моделирование и реорганизация материальных, финансовых и информационных потоков. Направлен на упрощение организационной структуры, перераспределение и минимизацию использования различных ресурсов, сокращение сроков реализации потребностей клиентов, повышение качества их обслуживания.

Применение информационных технологий (ИТ) — важнейшая часть реинжиниринга. Знание достижений в области ИТ позволяет реорганизовать процессы на новых принципах с использованием компьютерных систем. Кроме того, сам процесс реинжиниринга, как правило, осуществляется с применением компьютерных средств поддержки, что позволяет существенно снизить риск неудачи.

Обычно в реинжиниринге бизнес-процессов используются современные средства автоматизации проектирования (CASE-технологии), например, CASE Oracle Designer2000, SilverRun, Natural Engineering Workbench и другие, или комплексные системы управления ресурсами предприятия (ERP), например, R/3, BAAN IV. В этих системах в специальном репозитории автоматизировано поддерживается модель бизнеса, используемая при создании информационной системы.

Разработчик: доцент, к.т.н. П.Г.Асалханов

ФОС одобрен на заседании предметно-цикловой комиссии технических дисциплин № 8 от «25» марта 2023 г.

Председатель ПЦК Е.А.Хуснудинова

(подпись)

СОГЛАСОВАНО:
Внешний эксперт:
к.т.н., доцент Ильин С.Н.
(должность, звание, квалификационная категория) *(подпись)* (Ф.И.О.)