МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«КОСТРОМСКАЯ ГОСУДАРСТВЕННАЯ СЕЛЬСКОХОЗЯЙСТВЕННАЯ АКАДЕМИЯ»

Утверждаю:
И.о. декана электроэнергетического факультета

Николай Александрович Климов
Подписано цифровой подписью: Николай Александрович Климов
Дата: 2024.09.11 16:12:19 +03'00'
_____ /Климов Н.А./
11 сентября 2024 года

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
по дисциплине

ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Специальность 09.02.07 Информационные системы и программирование

Квалификация выпускника программист

Форма обучения очная

Срок освоения ППССЗ 3 года 10 месяцев

На базе основного общего образования

Караваево 2024

Фонд оценочных средств предназначен для оценивания сформированности компетенций по дисциплине «Инструментальные средства разработки программного обеспечения».

Разработчик:
преподаватель кафедры информационных технологий в электроэнергетике А.С. Яблоков _____

Алексей Сергеевич Яблоков
Подписано цифровой подписью: Алексей Сергеевич Яблоков
Дата: 2024.09.05 14:47:13 +03'00'

Утвержден на заседании кафедры информационных технологий в электроэнергетике, протокол № 1 от 05.09.2024

Заведующий кафедрой Н.А. Климов _____

Николай Александрович Климов
Подписано цифровой подписью: Николай Александрович Климов
Дата: 2024.09.05 14:47:57 +03'00'

Согласовано:
председатель методической комиссии электроэнергетического факультета
протокол № 7 от 10.09.2024

А.С. Яблоков _____

Алексей Сергеевич Яблоков
Подписано цифровой подписью: Алексей Сергеевич Яблоков
Дата: 2024.09.10 15:14:06 +03'00'

Результаты освоения дисциплины:
«Инструментальные средства разработки программного обеспечения»
ППССЗ (СПО) по специальности:
09.02.07 Информационные системы и программирование

Таблица 1

Коды компетенций по ФГОС	Компетенции	Результат освоения
Общие компетенции		
ОК 02.	Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности	**Знать** номенклатуру информационных источников, применяемых в профессиональной деятельности; приемы структурирования информации; формат оформления результатов поиска информации **Уметь** 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 **Владеть** навыками использования современных средств поиска, анализа и интерпретации информации; информационными технологиями для выполнения профессиональной деятельности
Профессиональные компетенции		
ПК 2.1.	Разрабатывать требования к программным модулям на основе анализа проектной и технической документации на предмет взаимодействия компонент	**Знать** основные принципы процесса разработки программного обеспечения; методы и способы идентификации сбоев и ошибок при интеграции приложений; методы отладочных классов; встроенные и основные специализированные инструменты анализа качества программных продуктов; графические средства проектирования архитектуры программных продуктов; методы организации работы в команде разработчиков **Уметь** анализировать проектную и техническую документацию; использовать специализированные графические средства построения и анализа архитектуры программных продуктов; определять источники и приемники данных; проводить сравнительный анализ. Выполнять отладку, используя методы и инструменты условной компиляции (классы Debug и Trace); оценивать размер минимального набора тестов; разрабатывать тестовые пакеты и тестовые сценарии; выявлять ошибки в системных компонентах на основе спецификаций **Владеть** навыками разработки и оформления требования к программным модулям по предложенной документации; навыками разработки тестовых наборов (пакетов) для программного модуля; навыками разработки тестовых сценариев программного средства; навыками инспектирования разработанных программных модулей на предмет соответствия стандартам кодирования

ПК 2.2.	Выполнять интеграцию модулей в программное обеспечение	**Знать** модели процесса разработки программного обеспечения; основные принципы процесса разработки программного обеспечения; основные подходы к интегрированию программных модулей; основы верификации программного обеспечения; современные технологии и инструменты интеграции; основные протоколы доступа к данным; методы и способы идентификации сбоев и ошибок при интеграции приложений; основные методы отладки; методы и схемы обработки исключи-тельных ситуаций; основные методы и виды тестирования программных продуктов; стандарты качества программной документации; основы организации инспектирования и верификации; приемы работы с инструментальными средствами тестирования и отладки; методы организации работы в команде разработчиков **Уметь** использовать выбранную систему контроля версий; использовать методы для получения кода с заданной функциональностью и степенью качества; организовывать заданную интеграцию модулей в программные средства на базе имеющейся архитектуры и автоматизации бизнес-процессов; использовать различные транспортные протоколы и стандарты форматирования сообщений; выполнять тестирование интеграции; организовывать постобработку данных; создавать классы- исключения на основе базовых классов; выполнять ручное и автоматизированное тестирование программного модуля; выявлять ошибки в системных компонентах на основе спецификаций; использовать приемы работы в системах контроля версий **Владеть** навыками интегрирования модулей в программное обеспечение; навыками отладки программных модулей; навыками инспектирования разработанных программных модулей на предмет соответствия стандартам кодирования
ПК 2.3.	Выполнять отладку программного модуля с использованием специализированных программных средств	**Знать** модели процесса разработки программного обеспечения; основные принципы процесса разработки программного обеспечения; основные подходы к интегрированию программных модулей; основы верификации и аттестации программного обеспечения; методы и способы идентификации сбоев и ошибок при интеграции приложений; основные методы отладки; методы и схемы обработки исключи-тельных ситуаций; приемы работы с инструментальными средствами тестирования и отладки; стандарты качества программной документации; основы организации инспектирования и верификации; встроенные и основные специализированные инструменты анализа качества программных продуктов; методы организации работы в команде разработчиков

		Уметь использовать выбранную систему контроля версий; использовать методы для получения кода с заданной функциональностью и степенью качества; анализировать проектную и техническую документацию; использовать инструментальные средства отладки программных продуктов; определять источники и приемники данных; выполнять тестирование интеграции; организовывать постобработку данных; использовать приемы работы в системах контроля версий; выполнять отладку, используя методы и инструменты условной компиляции; выявлять ошибки в системных компонентах на основе спецификаций **Владеть** навыками отладки программных модулей; навыками инспектирования разработанных программных модулей на предмет соответствия стандартам кодирования.

Требования к результатам освоения дисциплины:

иметь навык:

$Н_1$ – разработки программного обеспечения с использованием инструментальных средств

уметь:

$У_1$ – использовать инструментальные средства разработки, контроля версий, тестирования отладки и документирования

знать:

$З_1$ – принцип работы систем контроля версий, интегрированных средств разработки, тестирования, отладки и документации программного обеспечения

Паспорт
фонда оценочных средств

Таблица 2

№ п/п	Контролируемые дидактические единицы	Контролируемые компетенции (или их части)	Наименование оценочных средств		
			Тесты, кол-во заданий	Другие оценочные средства	
				вид	кол-во заданий
1	Раздел 1. Разработка программного обеспечения	ОК 02 ПК 2.1 ПК 2.2	37	Опрос	35
2	Раздел 2. Тестирование и отладка программного обеспечения	ОК 02 ПК 2.1 ПК 2.2 ПК 2.3	19	Опрос	37
3	Раздел 3. Документирование программного обеспечения	ОК 02 ПК 2.1 ПК 2.2	15	Опрос	24
Всего:			**71**		**96**

Методика проведения контроля по проверке базовых знаний по дисциплине «Инструментальные средства разработки программного обеспечения»

Раздел 1. Разработка программного обеспечения

Контролируемые компетенции ОК 02; ПК 2.1; ПК 2.2

Вопросы для устного опроса

1. Сформулируйте определение интегрированной среды разработки программ.
2. Каковы основные компоненты интегрированной среды?
3. Назовите наиболее популярные интегрированные среды и их фирмы-разработчики.
4. Что такое текстовый редактор?
5. Какие дополнительный функции по синтаксической проверке вводимого исходного кода встроены в современные редакторы в интегрированной среде?
6. Что такое сборка программ?
7. Что такое отладчик и каковы его типовые команды?
8. Какую функциональность обеспечивает поддержка коллективной разработки программ?
9. Что такое Team Foundation Server?
10. Что такое рефакторинг?
11. Какие функции реализует поддержка моделирования программ на языке UML?
12. Что такое обфускация и с какой целью она выполняется?
13. Что такое моноязыковые и многоязыковые интегрированные среды?
14. Чем интегрированная среда разработки программного обеспечения отличается от редактора кода?
15. Что такое код-ревью?
16. Почему необходимо проводить код-ревью?
17. Какие преимущества дает использование автоматических инструментальных средств для код-ревью?
18. Перечислите несколько программ для код-ревью, опишите их возможности?
19. Что такое Git?
20. Что такое репозиторий в Git?
21. Для чего нужен файл .gitignore?
22. Что такое система контроля версий (VCS)?
23. В чем разница между git init и git clone?
24. Что такое статус git?
25. Что такое коммит в Git?
26. Что такое ‘конфликт’ в git?
27. В чем разница между командами git remote и git clone?
28. В чем преимущества Git перед SVN?
29. Как отменить коммит, который уже был отправлен и обнародован?
30. Что такое HEAD в Git?
31. Как Git хранит данные?
32. Что такое ветвление в Git?
33. Объясните разницу между git reset, git revert и git checkout?
34. Что такое "git diff`?
35. Объясните разницу между "git merge" и "git rebase" и когда вы бы использовали каждый из них?

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который прочно усвоил программный материал в полном объеме, исчерпывающе, грамотно и логически стройно его излагает, четко формулирует основные понятия, приводит соответствующие примеры, уверенно владеет материалом.

Оценка «хорошо» выставляется обучающемуся, который твердо усвоил программный материал, грамотно и по существу излагает его без существенных ошибок, правильно применяет теоретические положения при решении конкретных задач, с небольшими погрешностями приводит формулировки определений, по ходу изложения допускает небольшие пробелы, не искажающие содержания ответа.

Оценка «удовлетворительно» выставляется обучающемуся, который не совсем твердо владеет программным материалом, знает основные теоретические положения изучаемой темы, при ответах допускает малосущественные погрешности, искажения логической последовательности при изложении материала, неточную аргументацию теоретических положений, испытывает затруднения при ответе на дополнительные вопросы.

Оценка «неудовлетворительно» выставляется обучающемуся, имеющему серьезные пробелы в знании учебного материала, допускающему принципиальные ошибки при ответе на вопросы.

Тестовые задания

Выберите один правильный вариант ответа

GitHub – это …

программа для работы с Git
драйвер для Git
+веб-сервер для хостинга IT-проектов и их совместной разработки, основанный на Git
UI для работы с локальной версией Git

Git репозиторий – это …

любая директория в операционной системе
любая папка, находящаяся внутри Git
+каталог файловой системы, в котором находятся файлы конфигурации репозитория, файлы журналов, индекс и хранилище
папка .git/

Команда git status …

+показывает состояние проекта
показывает имя и email пользователя
показывает место, занимаемое репозиторием на жестком диске
не существует

Команда git add выполняет следующую функцию:

создает файл с указанным именем и сразу добавляет его в Git
добавляет локальный файл в удаленный репозиторий
это синоним команды git commit
+начинает отслеживать указанный файл или файлы

Статус файла untracked в выводе команды git status означает, что …

+система Git не отслеживает этот файл
файл был удален из Git
файл находится вне репозитория Git
файл добавлен в .gitignore

Статус файла new в выводе команды git status означает, что …

файл только что был создан и еще не отслеживается системой Git
+файл только что начал отслеживаться Git и пока не имеет истории
файл удаляли из Git и потом восстановили командой git return
такого статуса нет

Статус файла modified в выводе команды git status означает, что …
+файл имеет историю в системе Git и был изменен относительно его последнего состояния
такого статуса нет, есть только new и deleted
файл перестал отслеживаться системой Git
файл добавлен в commit

Коммит – это …
+единица состояния проекта в Git
результат вывода команды git diff
обобщенное название одного из статусов файла в выводе git status
ошибочный термин в Git

Чтобы сделать коммит нужно …
набрать команду git commit в любой момент времени
сделать изменения в файлах и перечислить их после git commit
+сделать изменения, собрать эти изменения командой «git add» или «git commit -a» и указать коммит-сообщение после ключа «-m»

В какой ситуации нужно делать git status?
чем чаще, тем лучше
всегда при создании коммита
всегда после команды git pull
+только если надо узнать, в каком статусе находится репозиторий

Ветка в репозитории Git – это …
то же самое, что и коммит
минимум два коммита с одинаковым коммит-сообщением
+разные пути развития проекта, по сути, разные последовательности коммитов
механизм изменения конкретного файла

Чем отличается master и origin master
это просто два разных названия одной ветви
+master принадлежит локальному репозиторию, а origin master – удаленному
это две разные ветки локального репозитория
ветки a origin master не существует

Отличие команды «git push» и «git pull» в том, что …
пишутся они по-разному, но выполняют одну и ту же функцию
команды «git pull» не существует, а команда «git push» нужна, чтобы выложить изменения в удаленный репозиторий
команды «git push» не существует, а команда «git pull» нужна, чтобы получить изменения из удаленного репозитория
+ а команда «git pull» нужна, чтобы получить изменения из удаленного репозитория, а команда «git push» нужна, чтобы выложить изменения в удаленный репозиторий

Команда git log …
пишет указанный после файла лог
не существует, есть только команда git look

+показывает историю коммитов
удаляет файл из репозитория

Команда git show …
+показывает изменения, сделанные в указанном коммите
показывает содержимое файла
показывает состояние проекта
показывает время

Чтобы узнать автора строчки в файле, в системе Git используется команда …
git show --author
git commit --author
+git blame
git status

Чтобы узнать сделанные локально изменения относительно последнего состояния репозитория, используется команда…
git show
+git diff
git izmeneniya
git commit

Чтобы отменить действие команды «git add» на файл, нужно выполнить …
git abort
git stash
git not-add
+git reset

Чтобы решить конфликт слияния в Git нужно …
+вручную поправить изменения там, где Git не смог это сделать автоматически и затем собрать все коммиты и выложить
создать новый репозиторий
выполнить команду git commit merge please
удалить файлы, для которых Git не знает, как слить изменения

Чтобы привести файл в начальное состояние (до изменения), нужно выполнить команду …
git abort /путь/к/файлу
+git checkout /путь/к/файлу
git pull /путь/к/файлу
git commit /путь/к/файлу

Команда git stash …
отменяет все изменения
+сохраняет все изменения в буфер
удаляет все изменения
не существует

Отменить слияние веток при конфликте можно командой…
git stash pop
+git merge --abort
git remove repository
git clean

Применить патч в Git можно командой …
+git apply /путь/к/файлу
git patch /путь/к/файлу
git add /путь/к/файлу
git checkout /путь/к/файлу

В репозитории Git …
+может быть любое количество веток
число веток настраивается в конфигурациях
может быть не больше двух веток
столько же веток, сколько участников в проекте

В Git ветку с названием my_branch можно создать командой …
+git branch my_branch
git create branch my_branch
git commit origin my_branch
git checkout my_branch

Команда «git branch» без каких-либо параметров …
переключает на последнюю используемую ветку
выдает ошибку
+выдает список локальных веток
выдает список удаленных (remote) веток

Чтобы сделать коммит для ветки my_branch …
+нужно переключиться на нее и дальше сделать коммит по тем же правилам, что и для ветки master
нужно в commit-сообщении прописать название ветки
необходимо выполнить команду git fetch origin my_branch
нужно создать отдельный репозиторий

Для удаления локальной ветки my_branch используется команда …
git delete branch my_branch
+git branch -D my_branch
git reset my_branch
git fetch origin my_branch

Как исправить ошибку «fatal: The current branch my_branch has upstream branch», возникающую при вводе git push?
никак, придется создавать репозиторий заново
+ввести команду git push -u my_branch
ошибка означает, что проблема в коде, так что нужно сначала исправить код проекта
переустановить Git или обновить на новую версию

Конфликты при слиянии веток возникают, потому что …
ветки были созданы в разное время
ветки были созданы от разных коммитов
+в обоих ветках есть изменения один и тех же строк
используется версия Git ниже 1.2

Скачать ветку their_branch, если ее нет в локальном она уже есть в удаленном (remote) репозитории, можно командой …

git clone their_branch
git get origin their_branch
+git fetch origin their_branch
git clone origin their_branch

Удалить все файлы untracked поможет команда …
+git clean -f
git delete
git stash
git reset --hard

Команда «git clean -fd» …
не существует
выдаст ошибку, так как ключа -d нет у этой команды
+удалит не только untracked файлы, но и папки
удалит не только untracked файлы, но и весь репозиторий

Для чего добавляют файлы в .gitignore?
Чтобы Git удалил их история, храня только последнюю версию
Чтобы Git при работе с ними переспрашивал «Are you sure you want interact with this file?»
+Чтобы Git не замечал их и на них не действовали любые команды Git
Файл gitignore не несет никакой смысловой нагрузки, так что не надо добавлять в него имена файлов

Чтобы добавить новую директорию в Git нужно …
добавить каждый файл в директорию Git
+добавить хотя бы один файл из этой директории в Git
выполнить команду git add -d
выполнить команду git clone

Директория с репозиторием от любой другой директории …
ничем не отличается
отличается правами доступа
+отличается наличием папки .git с настройками репозитория
отличается тем, что прописывается в реестр операционной системы

Команда «git abuse» …
удаляет лишние файлы из репозитория и очищает буфер
находит и показывает все файлы с неправильным форматом
не существует, есть только ключ --abuse
+не существует, есть команда git reset

Методика проведения контроля

Параметры методики	Значение параметра
Предел длительности всего контроля	10 минут
Последовательность выбора вопросов	Случайная
Предлагаемое количество вопросов	10

Раздел 2. Тестирование и отладка программного обеспечения

Контролируемые компетенции ОК 02; ПК 2.1; ПК 2.2; ПК 2.3

Вопросы для устного опроса

1. Объясните, что такое «обеспечение качества» при разработке программных продуктов (Quality Assurance).
2. Объясните, что такое «контроль качества» при разработке программных продуктов (Quality Control).
3. Объясните, что такое тестирование ПО.
4. Какие основные цели тестирования ПО?
5. Когда следует начинать тестировать ПО?
6. Когда следует заканчивать тестирование ПО?
7. Какие основные уровни тестирования ПО?
8. Приведите несколько примеров, которые объясняют критерии входа для тестирования ПО.
9. Приведите несколько примеров, которые объясняют критерии выхода для тестирования ПО.
10. Приведите несколько инструментов, которые могут использоваться для автоматизации тестирования.
11. Как вы объясните Bug/Defect/Error в ПО?
12. Опишите различные мероприятия процесса верификации.
13. Приведите примеры верификации в зависимости от уровней тестирования.
14. Приведите несколько причин, которые приводят к багам в ПО.
15. Что такое процедура тестирования (Test Procedure)?
16. Что такое программный компонент?
17. Что такое «покрытие кода»?
18. Что такое «инспекция кода»?
19. Что такое тестирование стабильности?
20. Расскажите про критичность (серьезность) бага и общепринятые уровни такой критичности.
21. Расскажите про приоритет бага.
22. Можно ли начинать тестирование без рабочей сборки?
23. Что такое тестовый драйвер и тестовая обвязка?
24. Что такое тестирование End-to-End?
25. Что такое функциональное тестирование? Какие основные типы функционального тестирования? Какие виды функциональных тестов вы знаете?
26. Что такое нефункциональное тестирование?
27. Что такое «тестирование белого ящика»?
28. Что такое «тестирование чёрного ящика»?
29. Что такое «конверсионное тестирование»?
30. Что такое «конформационное тестирование»?
31. Что такое отладка ПО?
32. Какие существуют основные методы отладки?
33. Может ли отладка помочь в оптимизации программы?
34. Что такое отладчик и как он работает?
35. В чём преимущества использования логирования при отладке?
36. Какие существуют инструменты для автоматизации отладки?
37. Можно ли отлаживать код без использования специальных инструментов?

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который прочно усвоил программный материал в полном объеме, исчерпывающе, грамотно и логически стройно его излагает, четко формулирует основные понятия, приводит соответствующие примеры, уверенно владеет материалом.

Оценка «хорошо» выставляется обучающемуся, который твердо усвоил программный материал, грамотно и по существу излагает его без существенных ошибок, правильно применяет теоретические положения при решении конкретных задач, с небольшими погрешностями приводит формулировки определений, по ходу изложения допускает небольшие пробелы, не искажающие содержания ответа.

Оценка «удовлетворительно» выставляется обучающемуся, который не совсем твердо владеет программным материалом, знает основные теоретические положения изучаемой темы, при ответах допускает малосущественные погрешности, искажения логической последовательности при изложении материала, неточную аргументацию теоретических положений, испытывает затруднения при ответе на дополнительные вопросы.

Оценка «неудовлетворительно» выставляется обучающемуся, имеющему серьезные пробелы в знании учебного материала, допускающему принципиальные ошибки при ответе на вопросы.

Тестовые задания

Выберите один правильный вариант ответа

Тестирование – это:

фаза тестирования, которая осуществляется конечными пользователями непосредственно перед официальным выпуском программного обеспечения

направление на поиск отсутствующей или неверно работающей функциональности, ошибок в доступе к базе данных, ошибки инициализации, проблемы с производительностью, ошибки интерфейса

+проверка соответствия программного обеспечения требованиям, осуществляемая с помощью наблюдения за его работой в специальных, искусственно построенных ситуациях.

К основной задаче тестирования относят:

+ построение такого набора ситуаций, который был бы достаточно представителен и позволял бы завершить тестирование с достаточной степенью уверенности в правильности программного обеспечения вообще и убедиться, что в конкретной ситуации ПО работает правильно, в соответствии с требованиями.

экономия времени команды тестировщиков в случае, если релиз имеет серьезные проблемы со своей готовностью к полному циклу тестирования.

снижение вероятности наличия дефектов, находящихся в программном обеспечении

Стратегия тестирования – это …

когда подаются некоторые данные на вход и проверятся результаты, в надежде найти несоответствия.

+система методов отбора и создания тестов для тестового набора

начало тестирования с терминальных классов (т.е. классов, не использующих методы других классов)

Процесс локализации и исправления ошибок, обнаруженных при тестировании ПО называют …

+отладкой

локализацией

инициализацией

Ошибки, обнаруженные компоновщиком при объединении модулей программы, называют

ошибками компиляции

+ошибками компоновки
ошибками выполнения

Ошибки, обнаруженные ОС, аппаратными средствами или пользователем при выполнении программы называют …
+ошибками выполнения
ошибками компиляции
ошибками компоновки

Ошибки, фиксируемые компилятором при выполнении синтаксического и частично семантического анализа программы, называют …
+ошибками компиляции
ошибками компоновки
ошибками выполнения

Что относится к ошибкам кодирования:
ошибки выполнения
+ошибки некорректного использования переменных, ошибки вычислений, ошибки взаимодействия модулей, игнорирование особенностей конкретного языка программирования
логические ошибки

Какой метод отладки программ описан в тексте: «Самый простой и естественный способ отладки программы. Метод эффективен, но не применим для программ со сложными вычислениями, для больших программ, а также в случаях, когда ошибка связана с неверным представлением программиста о выполнении операций»:
метод индукции
+метод ручного тестирования
метод обратного прослеживания

Какой метод отладки программ описан в тексте: «Сначала формируют множество причин, которые могли бы вызвать данное проявление ошибки. Затем, анализируя причины, исключают те, которые противоречат имеющим данным.»:
метод индукции
метод ручного тестирования
+метод дедукции

Какой метод отладки программ описан в тексте: «Метод основан на тщательном анализе симптомов ошибки, которые могут проявляться как неверные результаты вычислений или как сообщение об ошибке.»
+метод индукции
метод ручного тестирования
метод дедукции

Какой метод отладки программ описан в тексте: «Начинается проверка с точки вывода неправильного результата. Для этой точки строится гипотеза о значениях основных переменных, которые могли бы привести к получению имеющегося результата»
метод индукции
метод ручного тестирования
+метод обратного прослеживания

Метод тестирования функционального поведения объекта с точки зрения внешнего мира:
тестирование «белого ящика»
тестирование «серого ящика»
+тестирование «черного ящика»

Метод, который позволяет исследовать внутреннюю структуру программы:
+тестирование «белого ящика»
тестирование «серого ящика»
«тестирование «черного ящика»

Структурный подход к формированию тестовых наборов:
основывается на том, что структура ПО не известна
+базируется на том, что известна структура тестируемого ПО, в том числе его алгоритмы соответствует основным критериям тестирования

Функциональный подход к формированию тестовых наборов:
+основывается на том, что структура ПО не известна
базируется на том, что известна структура тестируемого ПО, в том числе его алгоритмы соответствует основным критериям тестирования

При статическом подходе к ручному контролю ПО:
анализируют внешние связи
+анализируют структуру, управляющие и информационные связи программы, ее входные и выходные данные.
анализируют только структуру

При динамическом подходе к ручному контролю ПО:
+моделируют процесс выполнения программы на заданных исходных данных
анализируют структуру, управляющие и информационные связи программы, ее входные и выходные данные.
анализируют только структуру

К основным методам ручного контроля относят:
контроль вычислений, соответствие заданным требованиям
контроль обращения к данным
+инспекции исходного текста, сквозные просмотры, проверка за столом, оценки программ

Методика проведения контроля

Параметры методики	Значение параметра
Предел длительности всего контроля	10 минут
Последовательность выбора вопросов	Случайная
Предлагаемое количество вопросов	10

Раздел 3. Документирование программного обеспечения

Контролируемые компетенции ОК 02; ПК 2.1; ПК 2.2

Вопросы для устного опроса

1. Каковы плюсы и минусы ЕСПД?
2. Что такое ЕСКД?
3. Как принято обозначать стандарт ЕСПД?
4. Какие виды программ выделяют?
5. Какие виды программных документов существуют?
6. Каково содержание программных документов?
7. Наряду с видами программ и программных документов по ГОСТ 19.101-77 ЕСПД какие ещё существуют подходы к систематизации программных документов и соответствующие им разновидности?
8. Перечислите стадии разработки программ.
9. Каковы обязательные стадии разработки программ?
10. В каких случаях этапы не являются обязательными при разработке программ?
11. Каковы структурные части программных документов?
12. Что содержит и как может использоваться техническое задание?
13. Каковы состав и требования к содержанию программного документа «Описание программы»?
14. Каковы состав и требования к содержанию программного документа «Пояснительная записка»?
15. Каковы состав и требования к содержанию программных документов «Руководство системного программиста» и «Руководство программиста»? В чем их отличие?
16. Что такое «Документация пользователя»? Каким характеристикам она должна отвечать.
17. Приведите список ключевых слов-синонимов для характеристик «Документации пользователя».
18. Как можно измерить характеристики «Документации пользователя»?
19. Какие функции предоставляет Doxygen?
20. Что такое Read the Docs?
21. Что такое «перекрёстные ссылки» в контексте Doxygen?
22. Какие преимущества предоставляют инструментальные средства документирования разработчикам и пользователям программного обеспечения?
23. Какие функции предоставляют современные инструменты для создания технической документации?
24. Какие инструменты документирования наиболее популярны среди разработчиков и какие факторы влияют на их выбор?

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который прочно усвоил программный материал в полном объеме, исчерпывающе, грамотно и логически стройно его излагает, четко формулирует основные понятия, приводит соответствующие примеры, уверенно владеет материалом.

Оценка «хорошо» выставляется обучающемуся, который твердо усвоил программный материал, грамотно и по существу излагает его без существенных ошибок, правильно применяет теоретические положения при решении конкретных задач, с небольшими погрешностями приводит формулировки определений, по ходу изложения допускает небольшие пробелы, не искажающие содержания ответа.

Оценка «удовлетворительно» выставляется обучающемуся, который не совсем твердо владеет программным материалом, знает основные теоретические положения изучаемой темы, при ответах допускает малосущественные погрешности, искажения логической последовательности при изложении материала, неточную аргументацию теоретических положений, испытывает затруднения при ответе на дополнительные вопросы.

Оценка «неудовлетворительно» выставляется обучающемуся, имеющему серьезные пробелы в знании учебного материала, допускающему принципиальные ошибки при ответе на вопросы.

Тестовые задания

Выберите один правильный вариант ответа

Документирование программного обеспечения - это…

+процесс создания и поддержания документации для программного обеспечения
набор инструментов для автоматизации процессов разработки программного обеспечения
метод функционального моделирования, используемый для описания бизнес-процессов.
расширение UML с добавлением специфических элементов и нотаций для конкретной предметной области

Какие виды документации существуют?

Техническая документация
Административная документация
Пользовательская документация
+Все перечисленные

Пользовательская документация включает в себя:

описание интерфейса пользователя и примеры использования
примеры использования и требования к аппаратному обеспечению
требования к аппаратному обеспечению и описание интерфейса
+ описание интерфейса пользователя, примеры использования, требования к аппаратному обеспечению

Техническая документация описывает:

архитектуру программного обеспечения
реализацию программного обеспечения
функционирование программного обеспечения
+архитектуру, реализацию и функционирование программного обеспечения

UML (Unified Modeling Language) – это …

визуальный язык программирования
+визуальный язык моделирования для описания поведения программного обеспечения
набор инструментов для автоматизации процессов разработки программного обеспечения
расширение браузера

Диаграмма классов UML описывает …

+ классы и их отношения в системе
порядок выполнения действий и взаимодействия объектов в системе
состояния и переходы между ними для каждого объекта в системе
структуру и зависимости между компонентами системы

Диаграмма последовательности UML описывает …

+ порядок выполнения действий и взаимодействия объектов в системе
классы и их отношения в системе
структуру и зависимости между компонентами системы
состояния и переходы между ними для каждого объекта в системе

Диаграмма состояний UML описывает …

классы и их отношения в системе
порядок выполнения действий и взаимодействия объектов в системе
+состояния и переходы между ними для каждого объекта в системе

структуру и зависимости между компонентами системы

Диаграмма компонентов UML описывает …
классы и их отношения в системе
порядок выполнения действий и взаимодействия объектов в системе
состояния и переходы между ними для каждого объекта в системе
+структуру и зависимости между компонентами системы

DFD (Data Flow Diagram) – это …
+диаграмма потоков данных, используемая для визуализации и анализа информационных потоков в системе
диаграмма компонентов UML
диаграмма развёртывания UML
диаграмма состояний и переходов между ними

К категории программ для документирования программного обеспечения относится …
Turbo Pascal
+Doxygen
Git
Sublime Text

Программы для документирования программного обеспечения поддерживают формат …
JPG
+Markdown
RTF
PDF

Какие программы для документирования программного обеспечения поддерживают создание онлайн-документации?
Sphinx
Doxygen
JSDoc
+ReadTheDocs

Какие программы для документирования программного обеспечения поддерживают создание интерактивных документов?
+ReadTheDocs
Doxygen
JSDoc
Sphinx

Read the Docs поддерживает форматы разметки:
+Sphinx, Markdown и MkDocs
HTML, LaTeX, RTF и JPG
C++, C, Java, Python
line, circle, rectangle

Методика проведения контроля

Параметры методики	Значение параметра
Предел длительности всего контроля	10 минут
Последовательность выбора вопросов	Случайная
Предлагаемое количество вопросов	10

Билеты к экзамену

Контролируемые компетенции ОК 02; ПК 2.1; ПК 2.2; ПК 2.3

Вариант 1

1. Назовите наиболее популярные интегрированные среды и их фирмы-разработчики. Какие дополнительный функции по синтаксической проверке вводимого исходного кода встроены в современные редакторы в интегрированной среде?
2. Объясните, что такое тестирование ПО. Какие основные цели тестирования ПО? Когда следует начинать тестировать ПО? Когда следует заканчивать тестирование ПО?
3. Единая система программной документации. Плюсы и минусы ЕСПД.

Вариант 2

1. Сформулируйте определение интегрированной среды разработки программ. Каковы основные компоненты интегрированной среды?
2. Опишите различные мероприятия процесса верификации. Приведите примеры верификации в зависимости от уровней тестирования.
3. Назовите структурные части программных документов. Что содержит и как может использоваться техническое задание?

Вариант 3

1. Что такое моноязыковые и многоязыковые интегрированные среды? Чем интегрированная среда разработки программного обеспечения отличается от редактора кода?
2. Приведите несколько причин, которые приводят к багам в ПО. Поясните термины «покрытие кода» и «инспекция кода».
3. Наряду с видами программ и программных документов по ГОСТ 19.101-77 ЕСПД какие ещё существуют подходы к систематизации программных документов и соответствующие им разновидности?

Вариант 4

1. Почему необходимо проводить код-ревью? Какие преимущества дает использование автоматических инструментальных средств для код-ревью? Перечислите несколько программ для код-ревью, опишите их возможности?
2. Что такое тестирование стабильности? Расскажите про критичность (серьезность) бага и общепринятые уровни такой критичности.
3. Каковы состав и требования к содержанию программного документа «Описание программы»?

Вариант 5

1. Что такое Git? Что такое репозиторий в Git? Для чего нужен файл .gitignore?
2. Какие основные уровни тестирования ПО? Приведите несколько примеров, которые объясняют критерии входа для тестирования ПО.
3. Какие функции предоставляют современные инструменты для создания технической документации?

Вариант 6

1. В чем разница между git init и git clone? Что такое статус git? Что такое коммит в Git?
2. Расскажите про приоритет бага. Можно ли начинать тестирование без рабочей сборки? Что такое тестовый драйвер и тестовая обвязка?
3. Что такое «Документация пользователя»? Каким характеристикам она должна отвечать.

Вариант 7

1. Что такое HEAD в Git? Как Git хранит данные? Что такое ветвление в Git?

2. Что такое функциональное тестирование? Какие основные типы функционального тестирования? Какие виды функциональных тестов вы знаете?
3. Какие инструменты документирования наиболее популярны среди разработчиков и какие факторы влияют на их выбор?

Вариант 8

1. Какую функциональность обеспечивает поддержка коллективной разработки программ? Что такое Team Foundation Server?
2. Что такое нефункциональное тестирование? Что такое «тестирование белого ящика»? Что такое «тестирование чёрного ящика»?
3. Какие функции предоставляет Doxygen?

Вариант 9

1. Какие функции реализует поддержка моделирования программ на языке UML?
2. Какие существуют основные методы отладки? Может ли отладка помочь в оптимизации программы?
3. Что такое «перекрёстные ссылки» в контексте Doxygen?

Вариант 10

1. Что такое рефакторинг? Что такое обфускация и с какой целью она выполняется?
2. Что такое отладчик и как он работает? В чём преимущества использования логирования при отладке?
3. Какие преимущества предоставляют инструментальные средства документирования разработчикам и пользователям программного обеспечения?

Критерии оценки:

Оценка «отлично» выставляется обучающемуся, который прочно усвоил программный материал в полном объеме, исчерпывающе, грамотно и логически стройно его излагает, четко формулирует основные понятия, приводит соответствующие примеры, уверенно владеет материалом.

Оценка «хорошо» выставляется обучающемуся, который твердо усвоил программный материал, грамотно и по существу излагает его без существенных ошибок, правильно применяет теоретические положения при решении конкретных задач, с небольшими погрешностями приводит формулировки определений, по ходу изложения допускает небольшие пробелы, не искажающие содержания ответа.

Оценка «удовлетворительно» выставляется обучающемуся, который не совсем твердо владеет программным материалом, знает основные теоретические положения изучаемой темы, при ответах допускает малосущественные погрешности, искажения логической последовательности при изложении материала, неточную аргументацию теоретических положений, испытывает затруднения при ответе на дополнительные вопросы.

Оценка «неудовлетворительно» выставляется обучающемуся, имеющему серьезные пробелы в знании учебного материала, допускающему принципиальные ошибки при ответе на вопросы.

Форма промежуточной аттестации по дисциплине *экзамен*.

Окончательные результаты обучения (формирования компетенций) определяются посредством перевода баллов, набранных студентом в процессе освоения дисциплины, в оценки:

– базовый уровень сформированности компетенции считается достигнутым если результат обучения соответствует оценке «удовлетворительно» (50 до 64 рейтинговых баллов);

– повышенный уровень сформированности компетенции считается достигнутым, если результат обучения соответствует оценкам «хорошо» (65-85 рейтинговых баллов) и «отлично» (86-100 рейтинговых баллов).

Дополнительные контрольные испытания

для студентов, набравших менее 50 баллов (в соответствии с Положением «О модульно-рейтинговой системе»), формируются из числа оценочных средств по темам, которые не освоены студентом.