ПРИЛОЖЕНИЕ

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Рязанский государственный радиотехнический университет имени В.Ф. Уткина»

КАФЕДРА «ЭЛЕКТРОННЫЕ ВЫЧИСЛИТЕЛЬНЫЕ МАШИНЫ»

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ
«Инструментальные средства разработки программного обеспечения»

Направление подготовки

38.03.05 Бизнес-информатика

ОПОП академического бакалавриата

«Бизнес-информатика»

Квалификация (степень) выпускника — бакалавр

Форма обучения — очная, очно-заочная

1 ОБЩИЕ ПОЛОЖЕНИЯ

Оценочные материалы – это совокупность учебно-методических материалов (практических заданий, описаний форм и процедур проверки), предназначенных для оценки качества освоения обучающимися данной дисциплины как части ОПОП.

Цель – оценить соответствие знаний, умений и владений, приобретенных обучающимся в процессе изучения дисциплины, целям и требованиям ОПОП в ходе проведения промежуточной аттестации.

Основная задача – обеспечить оценку уровня сформированности компетенций, закрепленных за дисциплиной.

Контроль знаний обучающихся проводится в форме промежуточной аттестации. Промежуточная аттестация проводится в форме зачета.

Форма проведения зачета – тестирование, письменный опрос по теоретическим вопросам.

2 ОПИСАНИЕ ПОКАЗАТЕЛЕЙ И КРИТЕРИЕВ ОЦЕНИВАНИЯ КОМПЕТЕНЦИЙ

Сформированность каждой компетенции (или ее части) в рамках освоения данной дисциплины оценивается по трехуровневой шкале:

1) пороговый уровень является обязательным для всех обучающихся по завершении освоения дисциплины;

2) продвинутый уровень характеризуется превышением минимальных характеристик сформированности компетенций по завершении освоения дисциплины;

3) эталонный уровень характеризуется максимально возможной выраженностью компетенций и является важным качественным ориентиром для самосовершенствования.

Уровень освоения компетенций, формируемых

дисциплиной: Описание критериев и шкалы оценивания

тестирования:

Шкала оценивания	Критерий
3 балла (эталонный уровень)	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 85 до 100%
2 балла (продвинутый уровень)	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 70 до 84%
1 балл (пороговый уровень)	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 50 до 69%
0 баллов	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 0 до 49%

Описание критериев и шкалы оценивания теоретического вопроса:

Шкала оценивания	Критерий
3 балла (эталонный уровень)	выставляется студенту, который дал полный ответ на вопрос, показал глубокие систематизированные знания, смог привести примеры, ответил на дополнительные вопросы преподавателя
2 балла (продвинутый уровень)	выставляется студенту, который дал полный ответ на вопрос, но на некоторые дополнительные вопросы преподавателя ответил только с помощью наводящих вопросов

1 балл (пороговый уровень)	выставляется студенту, который дал неполный ответ на вопрос в билете и смог ответить на дополнительные вопросы только с помощью преподавателя
0 баллов	выставляется студенту, который не смог ответить на вопрос

На промежуточную аттестацию (зачет) выносится тест, два теоретических вопроса. Максимально студент может набрать 6 баллов. Итоговый суммарный балл студента, полученный при прохождении промежуточной аттестации, переводится в традиционную форму по системе «зачтено», «не зачтено».

Оценка «зачтено» выставляется студенту, который набрал в сумме не менее 4 баллов (выполнил одно задание на эталонном уровне, другое – не ниже порогового, либо оба задания выполнит на продвинутом уровне). Обязательным условием является выполнение всех предусмотренных в течение семестра практических и лабораторных работ заданий.

Оценка «не зачтено» выставляется студенту, который набрал в сумме менее 4 баллов, либо имеет к моменту проведения промежуточной аттестации несданные практические, либо лабораторные работы.

3 ПАСПОРТ ОЦЕНОЧНЫХ МАТЕРИАЛОВ ПО ДИСЦИПЛИНЕ

Контролируемые разделы (темы) дисциплины	**Код контролируемой компетенции (или её части)**	**Вид, метод, форма оценочного мероприятия**
Раздел 1. Основные понятия и цели дисциплины	**ПК-3.1, ПК-3.2**	**Зачет**
Раздел 2. Системы контроля версий	**ПК-3.1, ПК-3.2**	**Зачет**
Раздел 3. Система контроля версий Subversion	**ПК-3.1, ПК-3.2**	**Зачет**
Раздел 4. Система контроля версий GIT	**ПК-3.1, ПК-3.2**	**Зачет**
Раздел 5. Системы отслеживания ошибок, средства автоматизации тестирования	**ПК-4.1, ПК-4.2, ПК-4.3**	**Зачет**
Раздел 6. Виртуализация	**ПК-4.1, ПК-4.2, ПК-4.3**	**Зачет**
Раздел 7. Контейнеризация	**ПК-4.1, ПК-4.2, ПК-4.3**	**Зачет**
Раздел 8. Системы управления проектами	**ПК-4.1, ПК-4.2, ПК-4.3**	**Зачет**

4 ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ ИЛИ ИНЫЕ МАТЕРИАЛЫ

4.1. Промежуточная аттестация в форме зачета

Код компетенции	Результаты освоения ОПОП Содержание компетенций
ПК-3	Способен применять знания и умения в области информационных технологий в рамках предконтрактного, аналитического и проектного этапов автоматизации задач организационного управления и бизнес-процессов

ПК-3.1. Применяет знания и умения в области информационных технологий при автоматизации задач организационного управления и бизнес-процессов
Знать Основы информационных технологий, применяемых при автоматизации задач организационного управления **Уметь** Применять знания и умения в области ИТ при решении задач автоматизации **Владеть** Навыками использования современных информационных средств автоматизации
ПК-3.2. Использует инструментальные средства автоматизации задач организационного управления и бизнес-процессов
Знать Современные инструментальные средства автоматизации **Уметь** Отбирать и применять инструментальные средства автоматизациизадач организационного управления **Владеть** Навыками использования современных инструментальных средств

1. Что такое система контроля версий? (выберите один вариант ответа)

а) Программное обеспечение, призванное избавить программиста от необходимости загружать проект к себе на машину

б) Программное обеспечение, призванное автоматизировать работу с историей файла, и организовать защищенное хранилище проекта

в) Программное обеспечение, позволяющее версионировать проект

2. Что такое репозиторий? (выберите один вариант ответа)

а) Место, где система контроля версий хранит все документы вместе с их историей

б) Директория для вашего проекта

в) Рабочая копия документов

3. Соедините описание системы контроля версий с её названием (соединить)

Локальные	СКВ, в которых вся работа производится с центральным хранилищем (все действия, так или иначе, зависят от него)
Централизованные	СКВ, главной парадигмой которых является локализация данных на машине каждого разработчика проекта

Децентрализованные	СКВ, хранящиеся только на локальном компьютере

4. Объедините СКВ с её типом (соединить)

GIT	Локальная
SVN	Распределённая и централизованная
Mercurial	Централизованная
CVS	Децентрализованная
Bazaar	Распределённая и локальная

5. К какому виду систем контроля версий относится Subversion? (выберите один вариант ответа)

а) Централизованному

б) Распределенному

в) Локальному

г) Всем вышеперечисленным

6. Соотнесите состояние хранилища и то, какие действия предпримет Subversion во время коммита (соединить)

Хранилище изменялось локально и не устарело	Система не сделает ничего, вам необходимо внести изменения и сделать коммит
Хранилище не изменялось и устарело	Никаких
Хранилище изменялось локально и устарело	Заберет изменения с сервера
Хранилище не изменялось и не устарело	Система забирает к себе изменения с сервера и пользователю нужно разрешить конфликты, если необходимо

7. Соотнесите модель с её описанием (соединить)

Блокирование - Изменение - Разблокирование	Оба пользователя копируют файл и редактируют его на своём компьютере. Затем один из пользователей отправляет изменения на сервер. Второй пользователь заканчивает свои изменения и тоже хочет отправить файлы на сервер, но не может, т.к. там уже находится обновленная версия. Он должен сначала взять изменения с сервера, объединить изменения, разрешить конфликты, если они есть, и затем отправить свои изменения на сервер.

Копирование — Изменение - Слияние	Пользователь, решивший редактировать файл, блокирует его, в то время другой пользователь не может его редактировать. При разблокировке файла следующим участником другой пользователь может забрать изменения предыдущего пользователя и редактировать файл дальше.

8. Отметьте, какие команды вы можете использовать для получения подсказок в Subversion (выберите несколько вариантов ответа)

а) svn help

б) svn resolve

в) svn commit

г) svn update

д) svn log

е) svn info

ж) svn revert

з) svn delete

9. Какие команды вы можете использовать для создания/получения репозитория? (выберите несколько вариантов ответа)

а) svnadmin create /path/to/rep

б) svn import mytree file:///usr/local/svn/newrepos/some/project \
-m "Initial import"

в) svn checkout http://svn.example.com/repos/calc

10. Отметьте команды для внесения изменений в рабочую копию (выберите несколько вариантов ответа)

а) svn add

б) svn delete

в) svn copy

г) svn move

д) svn mkdir

е) svn update

11. Объедините способы разрешения конфликтов с их описанием (соединить)

1) Объединить конфликтный файл: Конфликтный файл будет выглядеть так: <<<<<<< имя файла ваши изменения =======	Отмена ваших изменений

результат автоматического слияния с репозиторием >>>>>>> ревизия 2) Выполнить "svn resolved"	
1) Выполнить "svn revert file"	Объединение вручную
Скопировать filename.rOLDREV или filename.rNEWREV	Копирование одного из файлов (своего или чужого)

12. Вопрос: верно ли, что в Git и Subversion используются принципиально разные подходы к хранению файлов репозитория? (выберите один вариант ответа)

а) да

б) нет

13. Вопрос: каковы были основные цели, преследуемые при создании Git? (выберите один вариант ответа)

а)

1. Простая архитектура.
2. Полная децентрализация.
3. Хорошая поддержка нелинейной разработки.

б)

1. Высокая скорость работы.
2. Применение подхода CVS.
3. Поддержка нелинейной разработки.

в)

1. Простая архитектура.
2. Централизованный подход.
3. Поддержка нелинейной разработки.

14. Под какой лицензией выпущена системы контроля версий Git? (краткий ответ)

---

15. Какой алгоритм использует Git для вычисления хэш-сумм? (выберите один вариант ответа)

а) CRC32

б) MD6

в) SHA-1

г) SHA-2

16. Что такое ветка в Subversion? (выберите один вариант ответа)

а) Направление разработки, которое существует независимо от другого направления

б) Направление разработки, которое существует независимо от другого направления, однако имеющие с ним общую историю

в) Копия репозитория у любого из разработчиков

17. В каких случаях есть необходимость создать ветку? (выберите несколько вариантов ответа)

а) Изменения, которые вы хотите внести, могут повредить работающему коду

б) Вы хотите написать улучшение/оптимизировать существующий код

в) Вы хотите зафиксировать изменения

18. Выберите верные сведения о создании ветвей в Subversion (выберите несколько вариантов ответа)

а) При использовании команды svn copy <удаленный url> <удаленный url> -m “Сообщение коммита”, вы создаёте ветку на удалённом сервере, а не на локальной машине

б) При использовании команды svn copy <удаленный url> <удаленный url> -m “Сообщение коммита”, вы создаёте ветку на локальной машине

в) При использовании команды svn copy <удаленный url> <удаленный url> -m “Сообщение коммита”, вам не обязательно иметь рабочую копию

г) При использовании команды svn copy <локальная директория> <локальная директория> вы создаёте ветку на локальной машине

д) При использовании команды svn copy <локальная директория> <локальная директория> вы создаёте ветку на удаленном сервере, а не на локальной машине

19. При создании ветки в Git командой git branch <branchname>, происходит ли автоматический переход на эту ветку? (выберите один вариант ответа)

а) да

б) нет

20. Что происходит при команде git checkout <branchname> (выберите один вариант ответа)

а) Мы переходим в директорию branchname

б) Указатель HEAD перемещается на ветку branchname

в) Мы создаём ветку branchname и перемещаемся на неё

21. По каким протоколам можно настроить работу Git на сервере? (выберите несколько вариантов ответа)

а) HTTP

б) FTP

в) SSH

г) Telnet

д) Git

е) SMTP

22. Что вам нужно сделать с проектом в случае «вы хотите внести изменения в репозиторий, в который у вас нет доступа»? (выберите один вариант ответа)

а) fork

б) pull request

23. Выберите случаи, когда вам может потребоваться использовать GitHub (выберите несколько вариантов ответа)

а) создать свой проект с открытым исходным кодом / использовать как хранилище кода, который не хотелось бы потерять

б) воспользоваться сторонней библиотекой (не входящей в список стандартных)

в) внести вклад в уже существующий проект на GitHub

Типовые вопросы открытого типа

1. Запишите команду SVN, позволяющую создать новый репозиторий (svnadmin create <путь>)
2. Запишите команду SVN, позволяющую создать рабочую копию (svn checkout <путь до репозитория>)
3. Запишите команду SVN, позволяющую просмотреть информацию о репозитории (svn info)
4. Запишите команду SVN, позволяющую просмотреть историю изменений (svn log)
5. Запишите команду SVN, позволяющую добавить файл под версионный контроль (svn add)
6. Запишите команду SVN, позволяющую обновить рабочую копию (svn update)
7. Запишите команду SVN, позволяющую зафиксировать изменения в репозиторий (svn commit)
8. Запишите команду SVN, позволяющую убрать файл из-под версионного контроля (svn delete)
9. Запишите команду SVN, позволяющую привести рабочую копию в согласованное состояние и очистить лог незавершенных операций (svn cleanup)
10. Запишите команду SVN, позволяющую объединить изменение из двух ветвей (svn merge)
11. Запишите команду GIT, позволяющую создать пустой репозиторий (git init)
12. Запишите команду GIT, позволяющую клонировать существующий репозиторий (git clone <путь>)
13. Запишите команду GIT, позволяющую выполнить первоначальную настройку репозитория (git config)
14. Запишите команду GIT, позволяющую проиндексировать изменения в рабочей директории (git add)
15. Запишите команду GIT, позволяющую зафиксировать изменения в локальный репозиторий (git commit)
16. Запишите команду GIT, позволяющую зафиксировать изменения в удаленный репозиторий (git push)
17. Запишите команду GIT, позволяющую получить данные из удаленного репозитория (git pull)
18. Запишите команду GIT, позволяющую создать ветку (git branch <имя ветки>)
19. Запишите команду GIT, позволяющую переключиться на созданную ветку (git checkout <имя ветки>)
20. Запишите команду GIT, позволяющую присоединить метку к текущему коммиту (git tag)
21. Запишите команду GIT, позволяющую выполнить слияние двух веток (git merge)
22. Запишите команду GIT, позволяющую просмотреть историю коммитов(git log)
23. Запишите команду GIT, позволяющую просмотреть проиндексированные изменения (git diff -staged)
24. Запишите команду GIT, позволяющую просмотреть выполненные локальные изменения (git diff)
25. Запишите команду GIT, позволяющую просмотреть справку по интересующей команде (git help <команда>)

Код компетенции	Результаты освоения ОПОП Содержание компетенций
ПК-4	Способен применять знания и умения в области программирования информационных систем в рамках предконтрактного, аналитического и проектного этапов автоматизации задач организационного управления и бизнес-процессов

ПК-4.1. Проектирует и формирует дизайн ИС
Знать Место и роль инструментальных средств при проектировании и формировании дизайна ИС **Уметь** Использовать инструментальные средства при проектировании и формировании дизайна ИС **Владеть** Навыками практического использования инструментальных средств при проектировании ИС
ПК-4.2. Моделирует ИС
Знать Место и роль инструментальных средств при моделировании ИС **Уметь** Использовать инструментальные средства при моделировании ИС **Владеть** Навыками практического использования инструментальных средств при моделировании ИС
ПК-4.3. Программирует ИС
Знать Место и роль инструментальных средств в процессе программирования ИС **Уметь** Использовать инструментальные средства в процессе программирования ИС **Владеть** Навыками практического использования инструментальных средств в процессе программирования ИС

24. Что такое система отслеживания ошибок? (выберите один вариант ответа)

а) Прикладная программа, которая ищет ошибки в вашем коде, и следит за этими участками

б) Прикладная программа, разработанная с целью помочь разработчикам программного обеспечения учитывать и контролировать ошибки, найденные в программах, пожелания пользователей, а также следить за процессом устранения этих ошибок и выполнением или невыполнения пожеланий

25. Что такое «баг» (в программировании)? (выберите один вариант ответа)

а) Жук, который забрался в системный блок компьютера

б) Ошибка в программе, которая выдаёт неожиданный или неправильный результат

в) Программа, которая пишется, чтобы найти ошибки в программе

26. На каком этапе программы (или кем) могут быть обнаружены «баги»? (выберите несколько вариантов ответа)

а) В процессе тестирования программы

б) В процессе отладки программы

в) На этапе проектирования программы

г) Сторонними пользователями приложения

27. Расставьте в нужном порядке последовательность действий при обнаружении ошибки технического характера (порядок)

а) Сообщение об обнаруженной ошибке технической поддержке приложения

б) Взятие ошибки в обработку членом команды разработки

в) Обнаружение ошибки пользователем

г) Исправление ошибки (закончено)

д) Выпуск патча (исправления) к существующему приложению

е) Занесение ошибки в систему отслеживания ошибок

ж) Работа над исправлением ошибки (члена команды разработки)

з) Смена статуса ошибки в системе отслеживания ошибок на «в работе»

и) Смена статуса ошибки в системе отслеживания ошибок на «закрыта»

28. Какие сведения об дефекте могут хранится в системе отслеживания ошибок? (выберите несколько вариантов ответа)

а) Номер (идентификатор) дефекта

б) Кто сообщил о дефекте

в) Версия продукта, в которой обнаружен дефект

г) Обсуждение того, кто возьмет задачу по устранению

д) Серьёзность (критичность) дефекта и приоритет решения

е) Описание шагов для выявления дефекта (воспроизведения неправильного поведения программы)

ж) Кто ответственен за устранение дефекта

з) Обсуждение возможных решений и их последствий

и) Текущее состояние (статус) дефекта

к) Версия продукта, в которой дефект исправлен

29. В каком состоянии может быть дефект? (выберите несколько вариантов ответа)

а) Открыт

б) Назначен

в) Некому назначить

г) Проверен

д) Отклонен

е) Закрыт

30. Что такое тестирование? (выберите несколько вариантов ответа)

а) Одна из техник контроля качества, включающая в себя активности по планированию работ, проектированию тестов, выполнению тестирования и анализу полученных результатов

б) Работа тестировщика

в) Проверка соответствия между реальным и ожидаемым поведением программы, осуществляемая на конечном наборе тестов, выбранном определенным образом

31. Выберите виды тестов, входящие в пирамиду тестирования (Майк Кон) (выберите несколько вариантов ответа)

а) юнит-тесты

б) интеграционные тесты

в) тесты корректности работы операционной системы, на которой будет установлено приложение

г) тесты пользовательского интерфейса

д) тесты правильности сборки аппаратной платформы компьютера

32. Выберите положения, правильные для юнит тестирования (выберите несколько вариантов ответа)

а) Должны не зависеть от окружения, на котором они выполняются

б) Запускаться регулярно в автоматическом режиме

в) Должны выполняться под специально настроенным окружением

г) Должны запускаться вручную для контроля за ними

33. Соедините методологию тестирования и ее описание (соединить)

Тестирование черного ящика	метод тестирования программного обеспечения, который предполагает, что внутренняя устройство системы известны тестировщику
Тестирование методом серого ящика	метод тестирования, базируется только лишь на тестировании по функциональной спецификации и требованиям, при этом не имея доступа во внутреннюю структуру кода и базу данных.
Тестирование методом белого ящика	Метод тестирования ПО, который предполагает, что внутреннее устройство программы нам известно лишь частично

34. Расставьте в нужном порядке последовательность действий при разработке через тестирование (порядок)

а) Пишеться тест, покрывающие желаемое изменение

б) Проводится рефакторинг нового кода к соответствующим стандартам

в) Пишется код, который позволит пройти тест

35. Соедините вид теста и то, что он проверяет (соединить)

Интеграционные тесты	Тестирование каждой нетривиальной функции или метода
Юнит-тесты	Тестирование корректного взаимодействия с другими приложениями
Тесты пользовательского интерфейса	Тестирование правильности работы пользовательского интерфейса приложения

36. Какие функции должна выполнять служба CI? (выберите несколько вариантов ответа)

а) получение исходного кода из репозитория
б) сборка проекта (компиляция)
в) выполнение тестов
г) развёртывание готового проекта
д) отправка отчетов (на электронную почту)
е) последовательное выведение релиза в промышленную среду

37. Как вы помните, CD включает в себя и CI. Напишите кратко, какую главную функцию выполняет CD, но не выполняет CI (краткий ответ)

---

38. Какие задачи входят в обязанности Dev-Ops инженера? (выберите несколько вариантов ответа)

а) Системное администрирование

б) Тестирование кода разработчиков

в) Программирование

г) Использование облачных технологий

д) Автоматизация крупной инфраструктуры

39. Соедините подкатегорию виртуализации с ее описанием (соединить)

виртуализация платформ	Данный вид виртуализации преследует своей целью комбинирование или упрощение представления аппаратных ресурсов для пользователя и получение неких пользовательских абстракций оборудования, пространств имен, сетей и т.п

виртуализация ресурсов	Продуктом этого вида виртуализации являются виртуальные машины - некие программные абстракции, запускаемые на платформе реальных аппаратно-программных систем

40. Какие подвиды включает виртуализация платформ? (выберите несколько вариантов ответа)

а) Разделение ресурсов

б) Инкапсуляция

в) Частичная виртуализация

г) Паравиртуализация

д) Виртуализация уровня операционной системы

е) Полная эмуляция (симуляция).

ж) Частичная эмуляция (нативная виртуализация).

з) Виртуализация уровня приложений

41. Какие подвиды включает виртуализация ресурсов? (выберите несколько вариантов ответа)

а) Разделение ресурсов

б) Инкапсуляция

в) Частичная виртуализация

г) Паравиртуализация

д) Виртуализация уровня операционной системы

е) Кластеризация компьютеров и распределенные вычисления (grid computing)

ж) Частичная эмуляция (нативная виртуализация).

з) Виртуализация уровня приложений

и) Объединение, агрегация и концентрация компонентов

42. Что такое контейнеризация? (выберите один вариант ответа)

а) Подход к разработке программного обеспечения, при котором приложение или служба, их зависимости и конфигурация упаковываются вместе в образ контейнера

б) Абстракция вычислительных ресурсов и предоставление пользователю системы, которая «инкапсулирует» (скрывает в себе) собственную реализацию

в) Набор изолированных приложений, не взаимодействующих друг с другом

43. Преимущества контейнеризации перед виртуализацией (выберите несколько вариантов ответа)

а) Более простая настройка контейнеров

б) Контейнеры требуют гораздо меньше ресурсов

в) Контейнеры легко развертывать

г) Контейнеры быстрее запускаются

д) Приложение в контейнере выполняется в любой среде

44. Соедините термин с его определением (соединить)

Контейнер	пакет со всеми зависимостями и сведениями, необходимыми для создания контейнера
Образ контейнера	действие по созданию образа контейнера на основе сведений и контекста, предоставленных файлом Dockerfile, а также дополнительных файлов в папке, где создается образ
Сборка	текстовый файл, содержащий инструкции по сборке образа Docker
Dockerfile	экземпляр образа Docker

45. Какие компоненты входят в каждый контейнер Docker? (выберите несколько вариантов ответа)

а) Выбранная операционная система (например, дистрибутив Linux, Windows Nano Server или Windows Server Core)

б) Файлы, добавленные разработчиком (двоичные файлы приложения и т. п.)

в) Сведения о конфигурации (параметры среды и зависимости)

г) Тесты для приложения

Ответы:

№ вопроса	Ответ	№ вопроса	Ответ
1	б	34	а, в, б
2	а	35	1-2, 2-1, 3-3
3	1-3, 2-1, 3-2	36	а, б, в, г, д
4	1-4 , 2-3 , 3-4 , 4-3 , 5-2	37	Ведение релиза к промышленной среде
5	а	38	а, в, г, д
6	1-2 , 2-3 , 3-4 , 4-1	39	1-2, 2-1
7	1-2, 2-1	40	в, г, д, е, ж, з
8	а, е	41	а, е, и
9	а, б, в	42	а
10	а, б, в, г, д, е	43	б, в, г, д
11	1-2, 2-1, 3-3	44	1-4, 2-1, 3-2, 4-3
12	а	45	а, б, в
13	а		
14	GNU GPL 2		
15	в		
16	б		
17	а, б		
18	а, в, г		
19	б		
20	б		
21	а, в, д		
22	б		
23	а, б, в		
24	б		
25	б		
26	а, б, г		
27	в, а, е, б, з, ж, г, и, д		
28	Все кроме «г»		
29	а, б, г, д, е		
30	а, в		
31	а, б, г		
32	а, б		
33	1-2 , 2-3, 3-1		

Типовые теоретические вопросы:

1. Системы управления версиями. Основные понятия.
2. Системы управления версиями. Сущность, назначение, преимущества.
3. Системы управления версиями. Виды и особенности.
4. Системы управления версиями. Основные представители на рынке.
5. Системы управления версиями. Модели версионирования. Достоинства и недостатки.
6. Subversion. Общие сведения, история создания.
7. Subversion. Основные понятия.
8. Subversion. Особенности хранения изменения и организации хранилища.
9. Subversion. Жизненный цикл проекта.
10. Subversion. Создание репозитория, импорт существующего. Создание рабочей копии.
11. Subversion. Создание рабочей копии. Обновление, внесение изменений в рабочую копию.
12. Subversion. Исправление конфликтов слияния.
13. Subversion. Анализ изменений, фиксация изменений. Просмотр истории изменений.
14. Subversion. Ветвление. Понятие ветки, суть работы в ветках.
15. Subversion. Способы создания веток. Работа с ветками.
16. Subversion. Копирование изменений между ветками.
17. Subversion. Ключевые понятия, стоящие за слиянием.
18. Subversion. Конфликты при объединении.
19. Git. Общие сведения, история создания.
20. Git. Основные понятия и определения.
21. Git. Особенности хранения изменения и организации хранилища. Целостность в Git.
22. Git. Жизненный цикл файла.
23. Git. Локальное выполнение операций.
24. Git. Первоначальная настройка репозитория.
25. Git. Жизненный цикл коммитов в Git.
26. Git. Создание репозитория, клонирование репозитория.
27. Git. Изменение файлов, запись изменений в репозиторий, просмотр изменений, отправка изменений на удаленный сервер.
28. Git. Механизм меток. Виды меток.
29. Git. Игнорирование файлов. Составление файла игнорирования.
30. Git. Ветвление. Понятие ветки, суть работы в ветках.
31. Git. Способы создания веток. Работа с ветками.
32. Git. Копирование изменений между ветками.
33. Git. Конфликты при слиянии. Способы разрешения конфликтов.
34. Git. Совместная серверная работа.
35. Git. Модель ветвления git flow.
36. Git. Поиск ломающего коммита методом половинного деления.
37. Системы отслеживания ошибок. Назначение, основные преимущества.
38. Системы отслеживания ошибок. Состав информации о дефекте.
39. Системы отслеживания ошибок. Жизненный цикл дефекта.
40. Системы отслеживания ошибок. Основные представители на рынке.

41. Тестирование программного обеспечения. Основные понятия, история тестирования.
42. Тестирование программного обеспечения. Пирамида тестирования.
43. Тестирование программного обеспечения. Модульное тестирование.
44. Тестирование программного обеспечения. Интеграционное тестирование.
45. Тестирование программного обеспечения. Тестирование UI.
46. Тестирование программного обеспечения. Виды тестирования.
47. Тестирование программного обеспечения. Тестирование методом чёрного ящика.
48. Тестирование программного обеспечения. Тестирование методом белого ящика.
49. Тестирование программного обеспечения. Тестирование методом серого ящика.
50. Тестирование программного обеспечения. Разработка через тестирование.
51. Тестовый фреймворк Test NG. Общие сведения и назначение.
52. Тестовый фреймворк Test NG. Автоматическая сборка. Особенности, преимущества.
53. Система автоматической сборки Gradle.
54. Тестовый фреймворк Test NG. Команды группы Assert.
55. Тестовый фреймворк Test NG. Аннотации.
56. Непрерывная интеграция и развертывание (CI/CD). Понятие и назначение.
57. CI/CD. Требования к проекту, организация CI. Преимущества и недостатки.
58. CI/CD. Непрерывная доставка. Общая схема CI/CD.
59. CI/CD. Основные принципы.
60. Devops. Понятие, области применения. Цели Devops.
61. Взаимодействие CI/CD и Devops.
62. Devops. Преимущества.
63. Виртуализация. Понятие, назначение.
64. Виртуализация. Виды виртуализации.
65. Виртуализация. Виды виртуализации платформ.
66. Виртуализация. Виды виртуализации ресурсов.
67. Виртуализация. Области применения, современное состояние.
68. Контейнеризация. Понятие, назначение, преимущества.
69. Контейнеризация. Docker. Назначение, терминология.
70. Контейнеризация. Рабочий процесс разработки приложений.