## **Лабораторная работа 7-8**

**Составить вопросы к собеседованию кандидата в соответствии с требованиями профессионального стандарта "Инженер-программист".**

* Часто говорят, что одна из самых важных задач объектно-ориентированного дизайна (и организации кода в целом) — это сильное сцепление и слабая связанность. Что это значит? Почему они так важны и как их достигнуть?
* Почему в большинстве языков индекс массива начинается с нуля?
* Как тесты и TDD влияют на организацию кода?
* Напишите фрагмент кода, нарушающий принцип Don't Repeat Yourself (DRY). Объясните, почему у него плохая организация, и исправьте.
* В чём разница между сцеплением и связанностью?
* Для чего полезен рефакторинг?
* Полезны ли комментарии в коде? Кто-то говорит, что их следует по возможности избегать, а в идеале они вообще не нужны. Вы согласны?
* Какая разница между организацией кода и архитектурой?
* Почему в TDD тесты пишутся прежде кода?
* C++ поддерживает множественное наследование, а Java позволяет классу реализовать несколько интерфейсов. Какое влияние оказывает использование этих средств на ортогональность? Существует ли разница во влиянии между использованием множественного наследования и множественных интерфейсов? Есть ли разница между использованием делегирования и наследования?
* Плюсы и минусы хранения логики предметной области в хранимых процедурах.
* По вашему мнению, почему объектно-ориентированная организация столько лет доминирует на рынке?
* Если у вашего кода плохая организация, как вы это поймёте?
* Расскажите о трёх главных недостатках вашего любимого языка программирования.
* Почему возрастает интерес к функциональному программированию?
* Что такое замыкание и для чего оно полезно? Что общего у замыканий и классов?
* Почему родные куки и сторонние куки обрабатываются настолько по-разному?
* Как бы вы управляли нумерацией версий Web Services API?
* С точки зрения бэкенда есть ли какие-нибудь недостатки или препятствия внедрению одностраничных приложений?
* Почему мы обычно так сильно стремимся, чтобы сервис был stateless? Что такого хорошего в stateless-коде? Почему и когда хранение состояния (statefullness) — это плохо?
* REST и SOAP: в каких ситуациях вы выберете один из этих подходов, а в каких другой?
* В веб-разработке подходы Model-View Controller и Model-View-View-Model во многом похожи как в бэкенде, так и во фронтенде. Что они из себя представляют и почему целесообразно их использовать?
* Как вы перенесёте приложение из одной базы данных в другую, например, из MySQL в PostgreSQL? Если вам придётся управлять таким проектом, какие проблемы вы ожидаете?
* Как реализовать ленивую загрузку? Когда она полезна? Какие есть подводные камни?
* Один из методов непрерывной интеграции называется сине-зелёный деплой. Он заключается в использовании двух сред производственной эксплуатации, максимально похожих — и деплой в одну из них во время работы другой, а затем безопасное переключение трафика из одной среды на вторую после тестирования в удобной обстановке. Этот метод усложняется, если деплой предполагает изменение структуры или содержимого БД. Хотелось бы обсудить с вами эту тему.
* Почему осуществление ветвлений в Mercurial или Git легче, чем в SVN?
* Каковы плюсы и минусы распределённых систем контроля версий, таких как Git, в сравнении с централизованными VCS вроде SVN?
* Можете описать рабочий процесс GitHub Flow и GitFlow?
* Что такое rebase (перемещение)?
* Почему слияние проще осуществлять в Mercurial или Git, чем в SVN и CVS?
* Зачем вообще нужен параллелизм? Объясните.
* Почему так трудно тестировать многопоточный/распараллеленный код?
* Что такое состояние гонки? Напишите пример на произвольном языке.
* Что такое взаимная блокировка? Вы можете написать какой-нибудь код, страдающий взаимными блокировками?
* Что такое ресурсный голод (process starvation)? Если нужно, давайте рассмотрим его определение.
* Что представляет собой алгоритм неблокирующей синхронизации без ожиданий (wait-free алгоритм)?
* Что такое гибкость (agility)?
* Как вы будете работать с легаси-кодом?
* Скажем, я ваш менеджер проекта и не эксперт в программировании. Попробуйте объяснить мне, что такое легаси-код и почему нужно заботиться о качестве кода.
* Я исполнительный директор вашей компании. Объясните мне Канбан и убедите вложиться в него.