1. Читали ли вы чужой код? Если да, то что это был за код? Опишите свои впечатления от него.

2 типа чтения было. Когда во время учебы помогал однокурсникам с дипломами, и когда приходилось править плагины для WP. Впечатления от второго более хорошие, код более чистый и с комментариями, но в принципе разбирался в обоих случаях в коде.

1. Опишите известные вам языки программирования и их предназначение.

Опишу с которыми так или иначе работал и мое мнение о них, чтобы не было как копипаста из интернета.

* Pascal – мертвый язык, годится только для обучения наверное. Сейчас наверное и этого не делают.
* Delphi – pascal с формами. Для обучения в свое время использовался, можно делать какие-то проекты, но наверно используется сейчас очень мало.
* Basic/Visual Basic – похожи на связку Pascal/Dephi, разве что на нем макросы для Microsoft Office пишут.
* Assembler – язык программирования низкого уровня, используется для программирования микроконтроллеров
* Java – в мое понимании язык мобильных платформ. Раньше писались игры на телефоне, сейчас почти каждое 2 приложение на Android пишется на Java.
* PHP – язык для серверной части ПО. Очень распространен.
* Python – альтернатива для PHP, судя по вакансия менее распространен, но так как понимаю его больше, вижу больше способов применения – от серверной части ПО до ботов, парсеров и прочих скриптов, облегчающих жизнь.
* С# - ООП. Работал с ним только в виде работы с формами, но может реализовывать и тех же ботов, и ту же серверную часть ПО

1. Что такое "компилятор", зачем он нужен и почему некоторые языки обходятся без него?

Преимущества использования компилятора заключаются в том, что он обычно работает быстрее, чем интерпретируемый код, так как ему не нужно обрабатывать код «на лету» во время работы приложения.

Кроме этого, скомпилированная программа будет проверена на наличие ошибок во время компиляции. Если есть команды, которые не понравились компилятору, то о них будет сообщено. Это позволяет исправлять все ошибки перед запуском программы.

1. Что такое "фреймворк" и для чего он нужен? Приведите примеры известных вам фреймворков.

Набор библиотек для упрощения написания ПО (и увеличение скорости написания). Как минус проблемы с безопасностью. Знаю только про Bootstrap, но не работал с ним.

1. Что за приставка "http://" перед адресами сайтов и почему она всё чаще теперь становится "https://"?

Протокол прикладного уровня передачи данных. Меняется на https так как сайты делают редирект на расширение протокола HTTP для поддержки шифрования в целях повышения безопасности (c поддержкой SSL-сертификатов)

1. Самая популярная библиотека для разработки фронтенд-приложений, ReactJS, моделирует логику в виде компонентов.  
   Если бы вам нужно было на ReactJS разработать страницу профиля ВКонтакте (например, <https://vk.com/id1>), на какие компоненты вы бы её разбили? Почему именно так?

Не работал с этой библиотекой. Напишу как думаю, не уверен что верно

* Меню
  + Картинка+текст
* Блок «Аватар»
  + Фото
  + Кнопка «Написать»
  + Кнопка «Подписаться»
  + Кнопка «…»
* Блок «Фото/Видео»
  + Медиа
    - Название раздела
    - Превью
    - Названия медиа
* Блок «Описание»
  + Блок «Имя»
    - ФИО
    - Статус
  + Разделы
    - Название раздела
    - Пункт раздела
      * Название пункта
      * Значение пункта
  + Цифры
    - Значение
    - Раздел значения
* Блок «Фотографии»
  + Название раздела
  + Фотографии
* Блок «Стена»
  + Запись
    - Автор
    - Текст
    - Дата
    - Лайки
    - Комментарии
      * Автор комментария
      * Текст комментария
      * Дата комментария
      * Лайки

1. SqlServer, PostgreSQL, SQLLite, MySQL, Oracle, Microsoft Access - разные базы данных с разным функционалом, которые разрабатываются, в основном, разными компаниями с разным видением своего продукта.

Однако все эти базы используют один и тот же язык запросов - SQL, и не планируют от него отказываться. Как так получается? Что такого в SQL, что он подходит всем этим базам?

А если он такой чудесный, то почему многие другие базы данных, вроде MongoDB или Cassandra, его не используют?

Основные достоинства языка SQL заключаются в следующем:

− стандартность –использование языка SQL в программах стандартизировано международными организациями;

− независимость от конкретных СУБД – все распространенные СУБД используют SQL, т.к. реляционную базу данных можно перенести с одной СУБД на другую с минимальными доработками;

− возможность переноса с одной вычислительной системы на другую – СУБД может быть ориентирована на различные вычислительные системы, однако приложения, созданные с помощью SQL, допускают использование как для локальных БД, так и для крупных многопользовательских систем;

− реляционная основа языка – SQL является языком реляционных БД, поэтому он стал популярным тогда, когда получила широкое распространение реляционная модель представления данных. Табличная структура реляционной БД хорошо понятна, а потому язык SQL прост для изучения;

− возможность создания интерактивных запросов – SQL обеспечивает пользователям немедленный доступ к данным, при этом в интерактивном режиме можно получить результат запроса за очень короткое время без написания сложной программы;

− возможность программного доступа к БД – язык SQL легко использовать в приложениях, которым необходимо обращаться к базам данных. Одни и те же операторы SQL употребляются как для интерактивного, так и программного доступа, поэтому части программ, содержащие обращение к БД, можно вначале проверить в интерактивном режиме, а затем встраивать в программу;

− обеспечение различного представления данных – с помощью SQL можно представить такую структуру данных, что тот или иной пользователь будет видеть различные их представления. Кроме того, данные из разных частей БД могут быть скомбинированы и представлены в виде одной простой таблицы, а значит, представления пригодны для усиления защиты БД и ее настройки под конкретные требования отдельных пользователей;

− возможность динамического изменения и расширения структуры БД – язык SQL позволяет манипулировать структурой БД, тем самым обеспечивая гибкость с точки зрения приспособленности БД к изменяющимся требованиям предметной области;

− поддержка архитектуры клиент-сервер – SQL – одно из лучших средств для реализации приложений на платформе клиент-сервер. SQL служит связующим звеном между взаимодействующей с пользователем клиентской системой и серверной системой, управляющей БД, позволяя каждой из них сосредоточиться на выполнении своих функций.

В MongoDB и Cassandra другой подход из-за специфики направления. В них делается попытка решить проблемы масштабируемости и доступности за счёт и согласованности данных.

1. <https://github.com/Sokolmi/Square>
2. Не совсем понял задание. Можете прислать пример базы данных, я напишу SQL запрос