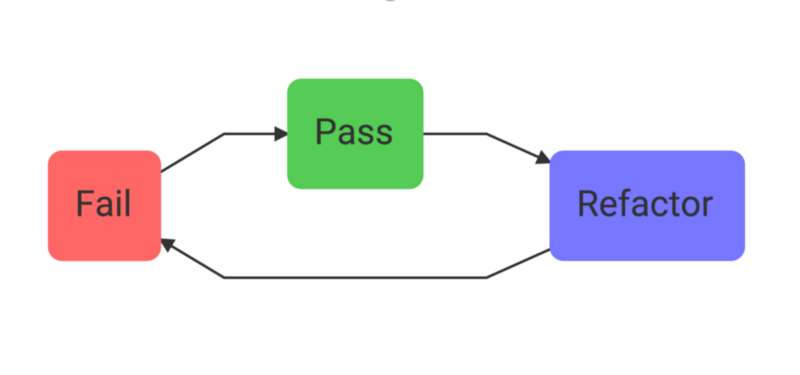
**Вопрос 1. Различные концепции разработки программного обеспечения (TDD, waterfall, cascade, iterative)**

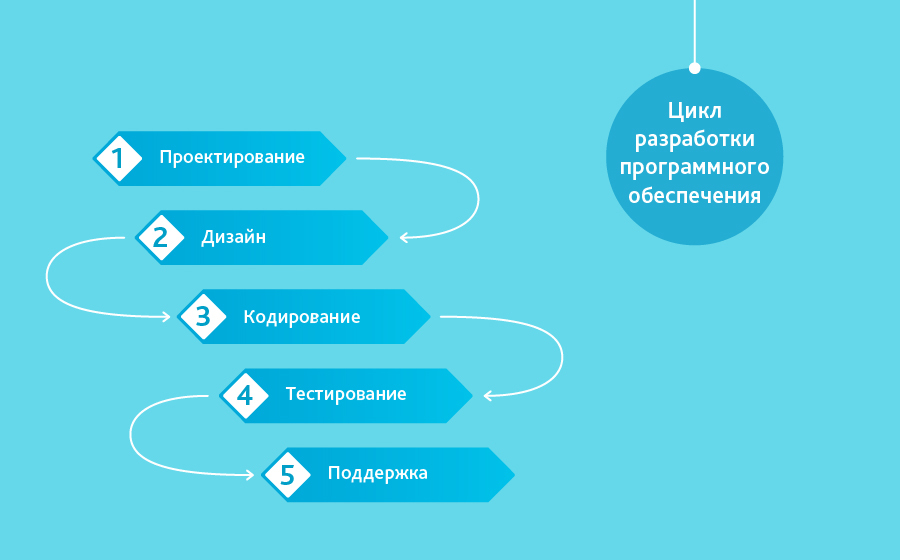
**TDD** расшифровывается как Test Driven Development (разработка через тестирование). Процесс, реализуемый в ходе применения этой методологии очень прост:



Вот основные принципы применения TDD:

1. Прежде чем писать код реализации некоей возможности, пишут тест, который позволяет проверить, работает ли этот будущий код реализации, или нет. Прежде чем переходить к следующему шагу, тест запускают и убеждаются в том, что он выдаёт ошибку. Благодаря этому можно быть уверенным в том, что тест не выдаёт ложноположительные результаты, это — своего рода тестирование самих тестов.
2. Создают реализацию возможности и добиваются того, чтобы она успешно прошла тестирование.
3. Выполняют, если это нужно, рефакторинг кода. Рефакторинг, при наличии теста, который способен указать разработчику на правильность или неправильность работы системы, вселяет в разработчика уверенность в его действиях.

**«Waterfall Model» (каскадная модель или «водопад»)**



Одна из самых старых, подразумевает последовательное прохождение стадий, каждая из которых должна завершиться полностью до начала следующей. В модели Waterfall легко управлять проектом. Благодаря её жесткости, разработка проходит быстро, стоимость и срок заранее определены. Но это палка о двух концах. Каскадная модель будет давать отличный результат только в проектах с четко и заранее определенными требованиями и способами их реализации. Нет возможности сделать шаг назад, тестирование начинается только после того, как разработка завершена или почти завершена. Продукты, разработанные по данной модели без обоснованного ее выбора, могут иметь недочеты (список требований нельзя скорректировать в любой момент), о которых становится известно лишь в конце из-за строгой последовательности действий. Стоимость внесения изменений высока, так как для ее инициализации приходится ждать завершения всего проекта. Тем не менее, фиксированная стоимость часто перевешивает минусы подхода. Исправление осознанных в процессе создания недостатков возможно, и, по нашему опыту, требует от одного до трех дополнительных соглашений к контракту с небольшим ТЗ.

Данная методология разработки ПО используется:

1. Только тогда, когда требования известны, понятны и зафиксированы. Противоречивых требований не имеется.
2. Нет проблем с доступностью программистов нужной квалификации.
3. В относительно небольших проектах.

«Incremental Model» (инкрементная модель)

В инкрементной модели полные требования к системе делятся на различные сборки. Терминология часто используется для описания поэтапной сборки ПО. Имеют место несколько циклов разработки, и вместе они составляют жизненный цикл «мульти-водопад». Цикл разделен на более мелкие легко создаваемые модули. Каждый модуль проходит через фазы определения требований, проектирования, кодирования, внедрения и тестирования. Процедура разработки по инкрементной модели предполагает выпуск на первом большом этапе продукта в базовой функциональности, а затем уже последовательное добавление новых функций, так называемых «инкрементов». Процесс продолжается до тех пор, пока не будет создана полная система.



Инкрементные модели используются там, где отдельные запросы на изменение ясны, могут быть легко формализованы и реализованы.

Данная методология разработки ПО используется:

1. Когда основные требования к системе четко определены и понятны. В то же время некоторые детали могут дорабатываться с течением времени.
2. Требуется ранний вывод продукта на рынок.
3. Есть несколько рисковых фич или целей.

Вопрос 2. Процедурно и объектно-ориентированные языки программирования и среды разработки.