Приложение №1 к Плану тестирования.

Задание на тестирование подключаемой библиотеки eLib

Настоящее задание является приложением к Плану тестирования системы клиент-серверной системы. Необходимо проводить разработку тестирующего кода в соответствии с Планом тестирования. Все работы по данному заданию разделить на следующие части:

- Рефакторинг существующего кода с целью улучшения тестируемости кода.
- Разработка шаблона для реализации блока тестов.
- Разработка модульных (unit) тестов eLib.
- Разработка оценочных тестов eLib.

1. РЕФАКТОРИНГ eLIB

Для обеспечения качественного автоматического unit-тестирования еще до начала его проведения необходим тестоориентированный код. При одновременной создании разработчиком класса вместе с тестами это условие выполняется сразу. В ситуации, когда Unit-тесты разрабатываются после кода необходим рефакторинг. С целью увеличения *тестируемости* компонента провести переработку кода по следующим направлениям:

1.1. Убрать лишние устаревшие методы.
1.2. Выделить работу со словарем в отдельный класс, который можно будет тестировать отдельно.
1.3. Использовать объектно-ориентированную C++ привязку, предоставляемую, вместо функционально-ориентированной C-привязки, которая используется сейчас.
1.4. Избавиться от глобальных функций, нарушающих принцип объектно-ориентированного программирования. Всю логику перенести непосредственно в методы класса.
1.5. Не производить открытие документа в каждом методе.
1.6. Функции по работе с индексными файлами преобразовать либо в отдельный класс (классы), либо в приватные методы существующего класса. Это опять же позволит их тестировать.
1.7. Необходимо сделать хотя бы минимальные комментарии к каждому методу и принципу ее функционирования и использования, поскольку с этим кодом начинается групповая работа.

2. РАЗРАБОТКА ШАБЛОНА ДЛЯ РЕАЛИЗАЦИИ БЛОКА ТЕСТОВ

Минимальным заданием автоматизированного тестирования является блок тестов. Блок тестов реализуется как отдельный исполняемый файл, который может содержать модульные, интеграционные, нагрузочные и т.п. тесты.

Необходимо разработать проект-шаблон блока модульных тестов для класса или группы классов одного компонента. Необходимо сделать это на примере eLib.

Использовать для этих целей надлежит специализированные классы модульного (unit) тестирования, что включает в себя классы unit-тестов, списки и шаблоны данных. Для модульного тестирования eLib используется AutoTestLib.

Данный шаблон представляет собой консольное приложение, которое выводит результаты в stdout, чтобы иметь возможность перенаправить их в другое место.

Важно выводить результаты тестирования по каждому тесту сразу по мере поступления данных, чтобы иметь возможность отлавливать фатальное завершение блока тестов. Исключения в работе также должны записываться в stdout.

3. РАЗРАБОТКА МОДУЛЬНЫХ (UNIT) ТЕСТОВ eLIB

Создается проект консольное приложение, в которое включаются исходные файлы. Тестироваться должен каждый класс. Для каждого метода тестируемого класса (либо для некоторой группы методов, если они работают в связке) создается отдельный *тестирующий класс***,** в котором для каждого варианта развития событий (включая ошибочные варианты) создается отдельный *тестирующий метод*. При написании тестирующих методов необходимо придерживаться идеологии «тестов, ориентированных на данные». Необходимо провести тестирование с 10-й различными наборами данных. Наборы не изменяются.

Все тестовые классы запускаются в консольном приложении через единый каркас и выводят результат в соответствии с шаблоном, разработанным ранее.

Контрольный пример должен иметь понятное лаконичное описание, которое является его документацией и присутствует в отчете о тестировании.

Общие правила.

- Необходимо тестировать правильное объявление и квалификаторы каждого класса. Для этого тестирующий код должен создать объект класса.
- Необходимо проводить тестирование успешного (наличия функции) завершения метода.
- Необходимо проверять возможность изменения каждого свойства в допустимом диапазоне.
- Необходимо тестировать каждый метод и свойство, таким образом обеспечивается покрытие структуры.
- Необходимо писать тест на каждый возможный ошибочный вариант поведение метода и использования свойств.
- Необходимо тестировать необходимые взаимосвязи классов в рамках необходимого наследования, полиморфизма, процедур вызова и т.п.

4. РАЗРАБОТКА ОЦЕНОЧНЫХ ТЕСТОВ РАБОТЫ eLIB

В связи с тем, что eLib использует много стороннего кода, необходимо провести дополнительно тестирование методом «черного ящика».

Необходимо экспертное тестирование качественных показателей работы eLib, которое также провести с 10-ю исходными наборами данных. Для определения средних значений используется 100 однотипных вызовов метода. Экспертные границы производительности устанавливаются с большим запасом, чтобы не вызывать ложных ошибок прогонки.

Правила.

- Требуется проверка факта, генерации эталонного элемента и сравнения хеша сгенерированного элемента с эталонным хешем.
- Требуется проверка результатов поиска, сравнение с эталонным значением.
- …
- …
- …
- Требуется определение времени выполнения каждого метода (полезной функции) и сравнение его с экспертным показателем. Тест неудачен, если время обработки намного больше.
- Требуется определение пикового расхода памяти за цикл и сравнение с эталонным расходом. Тест неудачен, если расход намного больше.
- Требуется подтверждение, факта отсутствия утечек памяти в каждом методе.