**Лабораторная работа №9**

**Проведение модульного тестирования программного обеспечения.**

**Цель работы:** сформировать умение выполнять модульное тестирование программного обеспечения

*Выводы:*

На основе проведённых тестов стала понятна корректность работы модулей исходного кода. В первом случае исходный код был полностью корректен и был хорошо протестирован. Во втором случае были использованы методы класса Assert. Результаты теста были удовлетворительными.

Модульное тестирование имеет ряд преимуществ, например: помогает в разработке и рефакторинге лучшего кода; гарантирует обеспечение гарантированного качества сборки, когда она интегрируется со сборкой; дефекты в приложении выявляются и устраняются на ранней стадии, что снижает стоимость тестирования.

Так же юнит-тесты очень просты в создании и использовании, что делает их доступными большому количеству пользователей.

**Контрольные вопросы:**

*1. Зачем нужно модульное тестирование?*

Модульное тестирование выявляет проблемы на ранних стадиях цикла разработки. Это включает в себя как ошибки в реализации программиста, так и недостатки или отсутствующие части спецификации устройства. Процесс написания тщательного набора тестов заставляет автора продумывать входы, выходы и условия ошибок и, таким образом, более четко определять желаемое поведение устройства.

*2. Что проверяется в модульном тестировании?*

Модульное тестирование проверяет функциональность конкретного куска кода, обычно по одной функции за раз. Интеграционное тестирование проверяет интерфейсы между компонентами, чтобы собранные воедино модули формировали систему, работающую, как задумано. Это важный момент, потому что большое количество тестов, которые называют модульными, на самом деле являются интеграционными тестами, а разработчики считают их модулями.

*3. Что нужно подвергать модульному тестированию?*

В нормальных системах модульные тесты нужно писать для:

* Модулей — неделимых изолированных частей системы, которые выполняют какую-то одну задачу (функция, метод, класс).
* Публичных методов.
* Защищённых методов, но только в редких случаях и когда никто не видит.
* Багов и их исправлений.