**Лабораторная работа №14 «Тестирование интеграции»**

**Цель работы.** Получить практические навыки отладки программ с помощью отладчика среды программирования.

1. Что такое тестирование программы?

**Тестирование** программного обеспечения (Software Testing) — **проверка** соответствия реальных и ожидаемых результатов поведения **программы**, проводимая на конечном наборе **тестов**, выбранном определённым образом/

1. Что такое отладка программы?

**Отла́дка** — этап разработки [компьютерной программы](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0), на котором обнаруживают, локализуют и устраняют ошибки

3.Какие стадии тестирования выделяют при разработке программного обеспечения?

Шестой **этап** – **тестирование** программы. Различается два вида **тестирования**: автономное и комплексное. При автономном **тестированию** подвергаются отдельные **программные** модули, из которых состоит **программный** комплекс. Комплексное **тестирование** заключается в проверке всего **программного** комплекса.

4.Какие различают подходы в формировании тестовых наборов?

Существуют два подхода к формированию тестов: структурный и функциональный.

Структурный подход базируется на том, что известна структура и алгоритмы тестируемого ПО, («стеклянный ящик»). Тесты строят так, чтобы проверить правильность реализации заданной логики в коде программы. Функциональный подход основывается на том, что структура ПО не известна («черный ящик»), тесты строят, опираясь на функциональные спецификации.

5.В чем суть тестирования методом ―покрытия операторов‖?

**Метод** **покрытия** **операторов**. Целью этого **метода** **тестирования** является выполнение каждого **оператора** программы хотя бы один раз. Если для **тестирования** задать значения переменных А = 2, B = 0, Х=3, **будет** реализован путь асе, т. е. каждый **оператор** программы выполнится один раз. Но если внести в алгоритм ошибки — заменить в первом условии and на or, а во втором Х> 1 на Х< 1.

6.В чем суть тестирования методом ―покрытия решений‖?

Более сильный **метод** **тестирования** известен как **покрытие** **решений** (**покрытие** переходов). Согласно данному **методу** должно **быть** написано достаточное число **тестов**, такое, что каждое направление перехода должно **быть** реализовано по крайней мере один раз. **Покрытие** **решений** обычно удовлетворяет критерию **покрытия** операторов.

7.В чем суть тестирования методом ―покрытия условий‖?

**Покрытие** кода (Code Coverage) - оценка **покрытия** исполняемого кода **тестами**, путем отслеживания непроверенных в процессе **тестирования** частей программного обеспечения. Тестовое **покрытие** на базе анализа потока управления - оценка **покрытия** основанная на определении путей выполнения кода программного модуля и создания выполняемых **тест** кейсов для **покрытия** этих путей.

8.В чем суть тестирования методом ―комбинаторного покрытия условий‖?

**Комбинаторные** **методы** построения **тестов** основаны на разделении каждого тестового воздействия на ряд элементов и построении **тестов** как всевозможных комбинаций полученных элементов, объединяемых по определенным правилам. **Комбинаторные** **методы** дают более высокую полноту **покрытия**, чем вероятностные, и при этом требуют ненамного больше ресурсов.

9.В чѐм суть метода эквивалентных разбиений?

**Эквивалентное** **разбиение** – это **метод** тестирования «черного ящика». Идея тестирования по **методу** **разбиения** классов эквивалентности состоит в том, чтобы исключить набор входных данных, которые заставляют систему вести себя одинаково и давать одинаковый результат при тестировании программы.

10.В чѐм суть метода анализа граничных значений?

**Анализ** **граничных** **значений** - это **метод** тестирования программного обеспечения, при котором тесты предназначены для включения представителей **граничных** **значений** в диапазон.

11.В чѐм суть метода анализа причинно-следственных связей?

**Анализ** **причинно**-**следственных** **связей** – это структурированный **метод**, применяемый для определения возможных причин нежелательного события или проблемы.

