**Занятие №25**

**Тема работы:** «Формирование тестов»

**Цель работы:** «Научиться, выполняя кодирование, создавать собственные компоненты. Производить тестирование модулей ПС.»

**Ход работы**

**Задание 1**

Провел тестирование своего ПС.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Выполненные действия | Шаги | Ожидаемый результат |
| 1 | Выбор обработки результатов | 1. Запустить программу  2. Нажать на кнопку «Обработать результаты» | 1. Программа запуститься  2. Открывается новое окно |
| 2 | Просмотр справки | 1. Запустить программу  2. Нажать на кнопку «Справка» | 1. Программа запуститься  2. Произойдёт открытие справки в новом окне |
| 3 | Просмотр «О программе» | 1. Запустить программу  2. Нажать на кнопку «О программе» | 1. Программа запуститься  2. Произойдёт открытие окно «О программе» в новом окне |
| 4 | Просмотр списка предметов | 1. Запустить программу  2. Нажать на кнопку «Обработать результаты»  3. Выбрать нужный предмет | 1. Программа запуститься  2. Откроется окно «Список предметов»  3. Выбрать нужный предмет |
| 5 | Обработка результатов для русского языка | 1. Запустить программу  2. Нажать на кнопку «Обработать результаты»  3. Выбрать предмет «Русский язык» | 1. Программа запуститься  2. Откроется окно «Список предметов»  3. Окно «Русский язык» открыто |
| 6 | Просмотр журнала учащихся | 1. Запустить программу  2. Нажать на кнопку «Обработать результаты»  3. Выбрать предмет «Русский язык» | 1. Программа запуститься  2. Откроется окно «Список предметов»  3. Окно «Русский язык» открыто |
| 7 | Добавление абитуриента | 1. Запустить программу  2. Нажать на кнопку «Обработать результаты»  3. Выбрать предмет «Русский язык»  4. Заполнить поля  5. Нажать на кнопку «Добавить» | 1. Программа запуститься  2. Откроется окно «Список предметов»  3. Окно «Русский язык» открыто  4. Поля заполнены  5. Абитуриент добавлен |
| 8 | Удаление абитуриента | 1. Запустить программу  2. Нажать на кнопку «Обработать результаты»  3. Выбрать предмет «Русский язык»  4. Установить курсор на нужного абитуриента  5. Нажать на кнопку «Удалить» | 1. Программа запуститься  2. Откроется окно «Список предметов»  3. Окно «Русский язык» открыто  4. Абитуриент удален |
| 9 | Изменение записи | 1. Запустить программу  2. Нажать на кнопку «Обработать результаты»  3. Выбрать предмет «Русский язык»  4. Установить курсор на нужного абитуриента  5. Заполнить поля  6. Нажать на кнопку «Добавить» | 1. Программа запуститься  2. Откроется окно «Список предметов»  3. Окно «Русский язык» открыто  4. Поля заполнены  5. Абитуриент добавлен |
| 10 | Выход из программы | 1. Запустить программу  2. Нажать на кнопку «Выход» | 1. Программа запуститься  2. Произойдёт выход из программы |

**Задание 2**

Ответил на контрольные вопросы.

**1. Модульное тестирование** – это тип тестирования программного обеспечения, при котором тестируются отдельные модули или компоненты программного обеспечения. Его цель заключается в том, чтобы проверить, что каждая единица программного кода работает должным образом.

2. Данный вид тестирование выполняется разработчиками на этапе кодирования приложения. Модульные тесты изолируют часть кода и проверяют его работоспособность.

3. 1) Анализ продукта

2) Разработка тестовой стратегии

3) Определение цели теста

4) Определение критерий тестирования

5) Планирование ресурсов

6) Планирование тестовой среды

7) Расписание и оценка.

4. **Интеграционное тестирование**– это тип тестирования, при котором программные модули объединяются логически и тестируются как группа.

5. **Интеграционные тесты** показывают, работает ли функция или сломана, поскольку они подчеркивают код в сценарии, близком к реальности. Они вызывают один или несколько программных методов или функций и проверяют, действуют ли они должным образом.

Напротив, **модульный тест** , тестирующий один метод, опирается на (часто ошибочное) предположение о том, что rest программного обеспечения работает правильно, потому что он явно издевается над каждой зависимостью.

6. **Монолитный метод сборки** предполагает выполнение авто­номного тестирования каждого модуля, а затем их одновремен­ную сборку и тестирование в комплексе.

**Пошаговое тестирование** предполагает последовательное подключение к набору уже оттестированных модулей очеред­ного тестируемого модуля. В качестве примера рассмотрим программу, состоящую из 9 модулей.

**Драйвер***-* это модуль, обеспечивающий вызов и передачу тес­тируемому модулю необходимых входных данных и обработку результатов.

**Заглушка***-* это модуль, имитирующий функции модулей, вызываемых тестируемым.

7. При восходящем тестировании программа собирается и тестируется «снизу вверх». Только модули самого нижнего уровня («терминальные» модули; модули, не вызывающие других модулей) тестируются изолированно, автономно. Затем тестируются модули, непосредственно вызывающие уже проверенные. Эти модули более высокого уровня тестируются не автономно, а вместе с уже проверенными модулями более низкого уровня. Процесс повторяется до тех пор, пока не будет достигнута вершина. Здесь завершается и тестирование модулей, и тестирование сопряжений программы.

8. При нисходящем подходе программа собирается и тестируется «сверху вниз». Изолированно тестируется только головной модуль. Затем с ним соединяются (например, редактором связей) один за другим модули, непосредственно вызываемые им, и тестируется полученная комбинация. Процесс повторяется до тех пор, пока не будут собраны и проверены все модули.

9. Сравнение нисходящего и восходящего тестирования

Нисходящее тестирование

|  |  |
| --- | --- |
| Преимущества | Недостатки |
| 1.  Имеет преимущества, если ошибки главным образом в верхней части программы.  2.  Представление теста облегчается после подключения функции ввода-вывода.  3.  Раннее формирование структуры программы позволяет провести ее демонстрацию пользователю и служит моральным стимулом. | 1.  Необходимо разрабатывать заглушки.  2.  Заглушки часто оказываются сложнее, чем кажется вначале.  3.  До применения функций ввода-вывода может быть сложно представлять тестовые данные в заглушки.  4.  Может оказаться трудным или невозможным создать тестовые условия.  5.  Сложнее оценка результатов тестирования.  6.  Допускается возможность формирования представления о совмещении тестирования и проектирования.  7.  Стимулируется не завершение тестирования некоторых классов/модулей. |

Восходящее тестирование

|  |  |
| --- | --- |
| Преимущества | Недостатки |
| 1.  Имеет преимущества, если ошибки главным образом в классе/модуле нижнего уровня.  2.  Легче создавать тестовые условия.  3.  Проще оценка результатов. | 1.  Необходимо разрабатывать драйверы.  2.  Программа как единое целое не существует до тех пор, пока не добавлен последний класс/модуль. |

10. **Системное тестирование** – это тестирование программного обеспечения выполняемое на полной, интегрированной системе, с целью проверки соответствия системы исходным требованиям, как функциональным, так и не функциональным.

11. Виды тестов при системном тестировании

* Функциональное тестирование
* Тестирование пользовательского интерфейса
* Юзабилити-тестирование
* Тестирование совместимости
* Тестирование на основе модели
* Тестирование безопасности
* Тестирование производительности
* Регрессионное тестирование