**5 Системное тестирование**

Системное тестирование - это тестирование программного обеспечения, выполняемое на полной, интегрированной системе, с целью проверки соответствия системы исходным требованиям.[8] Системное тестирование относится к методам тестирования чёрного ящика, и, тем самым, не требует знаний о внутреннем устройстве системы. Тестирование бывает:

* функциональное тестирование;
* тестирование пользовательского интерфейса;
* тестирование совместимости;
* тестирование безопасности;
* тестирование производительности;
* автоматическое тестирование;
* интеграционное тестирование.

**5.1** **Функциональное тестирование**

Функциональное тестирование - это тестирование программного обеспечения в целях проверки реализуемости функциональных требований, то есть способности программного обеспечения в определённых условиях решать задачи, нужные пользователям. Функциональные требования определяют, что именно делает программное обеспечение, какие задачи оно решает.

Функциональные требования включают в себя:

* функциональная пригодность;
* точность;
* способность к взаимодействию;
* соответствие стандартам и правилам;
* защищённость.

Тестирование и отладка программы являются наиважнейшими этапами разработки любых программных продуктов. Цель этого этапа – проверка правильности и точности реализации функций, выполнение которых возлагается на данный программный продукт. В случае выявления некоторых неточностей и ошибок необходимо проведение работ по их исправлению и доработке программного продукта до требуемого уровня.

На основе функций, которые должны быть протестированы в разработанной информационной обучающей системе, составлен чек-лист (checklist), приведенный в таблице 5.1.

Таблица 5.1 – Чек-лист

|  |  |  |  |
| --- | --- | --- | --- |
| Тестируемый модуль | | Тестируемая функция | Результат |
| 1 | | 2 | 3 |
| Запуск приложения | | Функция запуска | Выполнено успешно |
| Выход из приложения | | Функция закрытия программы |
| Ввод функции | | Определение правильности распознания функции, вводимой в виде строки | Выполнено успешно |
| Ввод букв | Вывод сообщении об ошибке | | Выполнено успешно |
| Сохранение и загрузка | Сохранение решения в файл | | Выполнена успешно |
| Загрузка данных задачи из файла | |

В качестве тестирования программного продукта был выбран тест-кейс (**Test Case)**.

**Test Case** - это артефакт, описывающий совокупность шагов, конкретных условий и параметров, необходимых для проверки реализации тестируемой функции или её части.

Тестирование производилось на ОС «Windws 10.

Таблица 5.2 – Тест-кейсы

|  |  |  |  |
| --- | --- | --- | --- |
| Краткое описание | Предварительные условия | Шаги для воспроизведения | Ожидаемый результат |
| Тестирование загрузки данных | 1.Запустить программу | 1. Нажать кнопку «загрузить»  2. Выбрать текстовый файл с данными функции  3. Нажать кнопку открыть | Все поля заполнятся данными из файла |
| Тестирование кнопки «Пример» | 1.Запустить программу | 1.Нажать кнопку пример | Все поля заполнятся данными примера |
| Тестирование сохранения | 1.Запустить программу | 1. Нажать кнопку «пример»  2. Нажать кнопку «решить»  3. Нажать кнопку «Сохранить»  4. Выбрать место и названия файла для сохранения  5. Подтвердить сохранение | При открытии сохраненного файла будут указаны функция, пошаговое решение и ответ. |
| Тестирование неправильного ввода | 1.Запустить программу | 1. Ввести в поля буквы  2. Нажать кнопку Решить | Пользователь увидит сообщение о некорректном вводе |
| Тестирование решения | 1.Запустить программу | 1. Нажать кнопку «пример»  2. Нажать кнопку «решить» | В самом правом поле пошагово будет написано решение функции. В поле ответа будет указаны координаты точки. |

**5.2** **Оценка безопасности**

До выпуска .NET Framework весь код, выполняемый на компьютере пользователя, имел те же права или разрешения на доступ к ресурсам, которые имел пользователь компьютера. [9]Например, если пользователю был разрешен доступ к файловой системе, то код также получал доступ к файловой системе Если пользователю был разрешен доступ к базе данных, то и коду предоставлялся доступ к ней. Наличие таких прав или разрешений может быть допустимо для кода в исполняемых файлах, которые пользователь собственноручно установил на локальном компьютере, но оно неприемлемо в тех случаях, когда речь идет о потенциально вредоносном коде, поступившем из Интернета или локальной интрасети. Такой код не должен получать доступ к ресурсам компьютера пользователя без его разрешения.

В .NET Framework введена инфраструктура, называемая системой управления доступом для кода, которая позволяет отличать разрешения (или права) от прав, которые имеет пользователь. По умолчанию код, поступающий из Интернета или интрасети, может выполняться только в режиме частичного доверия. В режиме частичного доверия приложение подвергается ряду ограничений: помимо всего прочего, приложению отказано в доступе к локальным жестким дискам, и оно не может запускать неуправляемый код. .NET Framework управляет ресурсами, к которым коду разрешен доступ в зависимости от идентификатора этого кода: от того, откуда он поступил, имеет ли он сборки со строгими именами, подписаны ли они сертификатом и т. д.

Технология ClickOnce, используемая для развертывания Windows Forms приложений, помогает упростить разработку приложений, работающих в режиме частичного доверия, с полным доверием или с частичным доверием с повышенными разрешениями. Технология ClickOnce предоставляет такие функции, как повышение уровня разрешений и развертывание доверенных приложений, поэтому приложение может запрашивать полное доверие или повысить уровень разрешений от локального пользователя в ответственном способе.

Общее представление о безопасности в .NET Framework

Управление доступом для кода позволяет считать код надежным в той или иной степени в зависимости от происхождения кода и других аспектов его удостоверения. Подробнее о данных, на основании которых среда CLR определяет политику безопасности, см. в разделе Свидетельство. Это помогает защитить компьютеры от вредоносного кода, а доверенный код — от умышленного или случайного нарушения безопасности. Управление доступом для кода также позволяет лучше контролировать действия, которые может выполнять приложение, так как вы можете указать только те разрешения, которые необходимы для работы приложения. Управление доступом для кода влияет на весь управляемый код, который предназначен для среды CLR, даже если этот код не производит ни одной проверки разрешений, предоставленных посредством управления доступом для кода. Дополнительные сведения о безопасности в .NET Framework см. в разделе Основные понятия безопасности и основы управления доступом для кода.

Если пользователь запускает исполняемый файл Windows Forms непосредственно с веб-сервера или из общей папки, то степень доверия, предоставляемая приложению, зависит от того, где размещается код и как он запущен. При запуске приложения происходит его автоматическая оценка, после которой приложение получает именованный набор разрешений от среды CLR. По умолчанию код на локальном компьютере получает набор разрешений полного доверия, код из локальной сети — набор разрешений локальной интрасети, а код из Интернета — набор разрешений Интернета.

**5.3 Тестирование производительности**

Тестирование производительности - это тестирование, которое проводится для определения скорости работы системы или её части при заданной нагрузке. Тестирование производительности стремится учесть производительность уже на стадии проектирования и моделирования и системы, до начала основной стадии разработки.

Тестирование производительности служит таким типичным целям:

* для демонстрации того, что система удовлетворяет критериям производительности;
* для определения производительность какой из двух или нескольких систем лучше;
* для определения, какой элемент нагрузки или часть системы приводит к снижению производительности.

Для оценки времени загрузки страниц системы был использован встроенный в VisualStudio прибор для отслеживания.

Приложение открывается за пару секунд и использует 20 мб памяти.

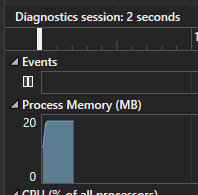


Рисунок 5.1 – открытие приложения

Открытие справочника занимает также несколько секунд. С открытым окном приложение занимает 66 метра. При дальнейшей работе с формами скачков не обнаружено

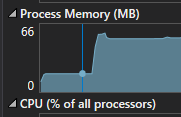


Рисунок 5.2 – открытие формы