МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «Полоцкий государственный университет»

Факультет информационных технологий

Кафедра технологий программирования

**ЛАБОРАТОРНАЯ РАБОТА №2**

по дисциплине: **«Надёжность программного обеспечения»**

на тему: ««Дефекты. Описание дефектов. Определение критичности и приоритетности дефекта»

Вариант 2

ВЫПОЛНИЛ студент группы 15-КБ

Осипенко В.В.

ПРОВЕРИЛ преподаватель

Попкова Д.В.

Новополоцк 2017 г.

**Цель работы**: Изучить понятие «дефект», изучить и научиться различать понятия «критичность» и «приоритетность» дефектов, научиться описывать дефекты, найденные в приложении.

**Краткие теоретические сведения**

Дефект (жарг. – баг) – ни что иное, как изъян в разработке программного продукта, который вызывает несоответствие ожидаемых результатов выполнения программного продукта и фактически полученных результатов. В различных IT-компаниях, где есть отдел тестировщиков, описание дефектов происходит фактически одинаково, иногда формы описания дефектов отличаются парой мелочей.

Описание дефекта состоит из нескольких пунктов:

1 Суть проблемы (Headline).

2 Степень критичности дефекта (Severity).

3 Алгоритм воспроизведения дефекта (Description).

4 Ожидаемое поведение в описанной ситуации (Expected result).

5 Вспомогательное средство передачи информации о проблеме (Attachment).

6 Степень критичности дефекта (Priority).

В поле Severity тестировщики выставляют критичность дефекта с точки зрения конечного пользователя. На основании уровня критичности, указанного в поле Severity, а также информации из других источников, менеджеры проектов определяют срочность исправления дефекта, т.е. назначают приоритет Критичность не влияет напрямую на порядок исправления дефектов. Очередность исправления в первую очередь зависит от приоритета. Критичность – это не единственный фактор, который влияет на приоритет.

**Таблица 1 –** Уровни критичности дефектов

|  |  |
| --- | --- |
| **Уровень критичности** | **Описание дефекта** |
| Blocker (Bugzilla) | Дефект полностью блокирует работу приложения. Продолжать тестирование при наличии такого дефекта невозможно. |

Продолжение таблицы 1

|  |  |
| --- | --- |
| Уровень критичности | Описание дефекта |
| Critical (CQ, Bugzilla, Jira) | Дефект полностью (CQ, Jira) или частично (Bugzilla) блокирует работу приложения. Продолжать тестирование при наличии такого дефекта невозможно. |
| Major (CQ, Bugzilla, Jira) | Дефект нарушает нормальную работу одной или нескольких функций приложения, но не препятствует дальнейшему проведению тестов. |
| Average (CQ, реализовано на корпоративной JIRA) Normal (Bugzilla, Jira) | Дефект частично влияет на основные функции приложения, но выполнение сценария в ходе тестирования возможно при минимальных изменениях. Графический дефект, значительно влияющий на восприятие проекта пользователями. |
| Minor (CQ, Bugzilla, Jira) | Несущественная функциональная ошибка (CQ, Bugzilla, Jira) или дефект графического интерфейса (CQ, Jira). Исправление незначительно улучшит поведение или выполнение сценария. Тестирование проводится согласно сценарию без каких-либо изменений. |
| Trivial (Bugzilla) | Дефект графического интерфейса, не вызывающий значительного ухудшения восприятия проекта пользователями. |

Продолжение таблицы 1

|  |  |
| --- | --- |
| Enhancement (CQ, Bugzilla, Jira) | Мелкий дефект, не требующий обязательного исправления, или рекомендация, не предполагающая обязательного внесения изменений. Тестирование проводится согласно сценарию без каких-либо изменений. |

В данном разделе описываются критерии, на которые ориентируется тестировщик при определении критичности дефекта. Критерии являются частью стандарта, которого тестировщики должны придерживаться. Наличие критериев гарантирует, что процесс определения критичности дефектов соответствует требованиям компании. Помимо указанных критериев, при определении уровня критичности дефектов тестировщику следует ориентироваться на целевую аудиторию проекта и сценарии его использования. Необходимо отметить, что определение уровня критичности дефекта является субъективным процессом, при этом оно зависит не только от восприятия критичности ошибок самим тестировщиком, но и от проекта, на котором найдена ошибка. По согласованию с командой разработки либо с Заказчиком, процесс определения уровней критичности может быть изменен. Помимо всего, на определение уровней критичности дефектов может влиять фаза развития проекта - на начальных этапах любые дефекты внешнего вида проекта могут вноситься исключительно с незначительным уровнем важности, зато на этапе тестирования финальной версии продукта любая ошибка в слове на странице может оцениваться как серьезная.

**Анализ задания.**

В рамках данного задания требуется найти всевозможные дефекты функциональности и интерфейса кнопок «4», «5», «6», «+», «-» программы «Калькулятор». Описать найденные дефекты, как это показано в теоретической части.

**Найденные дефекты**

**Дефект №1**

**Headline**

При сложении 3-ёх и более чисел, калькулятор складывает только 2 последних числа.

**Severity:**

Major (CQ, Bugzilla, Jira) - Дефект нарушает нормальную работу одной или нескольких функций приложения, но не препятствует дальнейшему проведению тестов.

**Description**

Шаг 1: Запускаем калькулятор.

Шаг 2: Cлаживаем 3 числа.

Шаг 3: Нажимаем на кнопку “=” и получаем конечный результат.

**Expected result**

Необходимо исправить алгоритм сложения цифр, калькулятор должен слаживать все введённые цифры, а не только последние.

**Attachment**

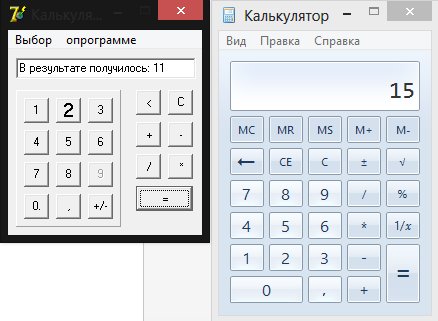


Рисунок 1 – Сравнение работ калькуляторов (Дефект №1)

**Priority**

Major priority.

**Дефект №2**

**Headline**

При вычитании с 3-мя и более чиселами, калькулятор производит вычитание только с 2 последними числами.

**Severity:**

Major (CQ, Bugzilla, Jira) - Дефект нарушает нормальную работу одной или нескольких функций приложения, но не препятствует дальнейшему .проведению тестов.

**Description**

Шаг 1: Запускаем калькулятор.

Шаг 2: Производим вычитания c 3-мя числами.

Шаг 3: Нажимаем на кнопку “=” и получаем конечный результат.

**Expected result**

Необходимо исправить алгоритм вычитания цифр, калькулятор должен вычитать все введённые цифры, а не только последние.

**Attachment**

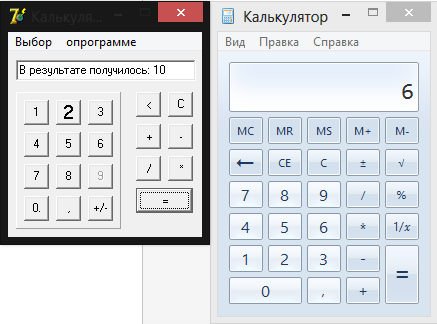


Рисунок 2 - Сравнение работ калькуляторов (Дефект №2)

**Priority**

Major priority.

**Дефект №3**

**Headline**

При выполнении двух различных операций(сложения и вычитания) одновременно, калькулятор производит только последнюю операцию.

**Severity:**

Major (CQ, Bugzilla, Jira) - Дефект нарушает нормальную работу одной или нескольких функций приложения, но не препятствует дальнейшему .проведению тестов.

**Description**

Шаг 1: Запускаем калькулятор.

Шаг 2: Производим сложение цифр

Шаг 3: Производим вычитание цифр.

Шаг 3: Нажимаем на кнопку “=” и получаем конечный результат.

**Expected result**

Необходимо исправить алгоритм выполнения нескольких операций одновременно, калькулятор должен производить все требуемые операции.

**Attachment**

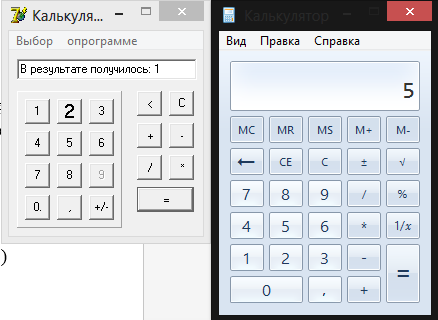


Рисунок 3 – Сравнение работ калькуляторов (Дефект №3)

**Priority**

Major priority.

**Дефект №4**

**Headline**

При вводе знаков “+”, “-” без последующего ввода числа, становится невозможным отменить последнее действие и вывести конечный результат.

**Severity:**

Minor (CQ, Bugzilla, Jira) - Несущественная функциональная ошибка (CQ, Bugzilla, Jira). Исправление незначительно улучшит поведение или выполнение сценария. Тестирование проводится согласно сценарию без каких-либо изменений

**Description**

Шаг 1: Запускаем калькулятор.

Шаг 2: Производим вычислительные операции.

Шаг 3: Ставим знак “+” либо знак “-” после последнего числа в наших расчётах.

**Expected result**

Необходимо добавить возможность отмены введённого знака после вычислений и получение конечного результата.

**Attachment**

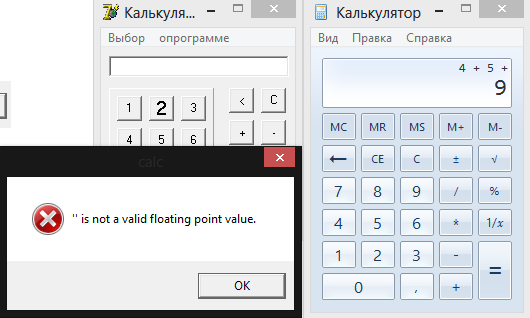


Рисунок 4 – Сравнение работ калькуляторов (Дефект №4)

**Priority**

Minor priority.

**Выводы о проделанной работе.**

В данной лабораторной работе мы занимались поисками багов, а также научились основам создания документации. Мы узнали о уровнях критичности и от чего они зависят. В данном варианте задания уровни критичности были на уровнях: Major (серьезный) и Minor (незначительный) т.к. в полученном варианте было обнаружено несколько серьезных а также один незначительный все данные ошибки не приводили к критическим сбоям программы однако они ухудшали качество работы с программой и могли ввести пользователя в заблуждение.