Министерство образования Российской Федерации

Тверской государственный технический университет

Кафедра Программного обеспечения

Программа для работы с личными делами студентов

Функциональное тестирование методом черного ящика

Выполнил: Бакеев А.В.

Проверил: Мальков А.А

Тверь, 2024

Оглавление

[Введение 3](#_Toc181099840)

[Постановка задачи 3](#_Toc181099841)

[Тестовые сценарии 4](#_Toc181099842)

[Классы эквивалентности 5](#_Toc181099843)

[Тестирование 6](#_Toc181099844)

# Введение

*Функциональное тестирование* объекта-тестирования планируется и проводится на основе требований к тестированию, заданных на этапе определения требований. Цель функциональных тестов состоит в том, чтобы проверить соответствие приложения установленным требованиям.

Основным источником исходных данных для тестирования являются функциональные и нефункциональные требования к разрабатываемой системе. Эти требования могут создаваться в различной форме. Например, это может быть Техническое Задание (по любому из ГОСТов) или артефакты RUP (документ VISION – Концепция, спецификации прецедентов и т.д.).

В основе функционального тестирования лежит методика "черного ящика". Идея тестирования сводится к тому, что группа тестировщиков проводит тестирование, не имея доступа к исходным текстам тестируемого приложения. При этом во внимание принимается только входящие требования и соответствие им тестируемым приложением.

При тестировании "*черного ящика"* (black box) программа рассматривается как объект, внутренняя структура которого неизвестна. Тестировщик вводит данные и анализирует результат, но, как именно работает программа, он не знает. Подбирая тесты, специалист ищет интересные с его точки зрения входные данные и условия, которые могут привести к нестандартным результатам. Интересны для него прежде всего те представители каждого класса входных данных, на которых с наибольшей вероятностью могут проявиться ошибки тестируемой программы.

Тестирование «черного ящика» обеспечивает поиск следующих категорий ошибок:

* некоректных или отсутствующих функций;
* ошибок интерфейса;
* ошибок во внешних структурах данных или в доступе к внешней базе данных;
* ошибок инициализации и завершения.

# Постановка задачи

**Вариант 2.** Разработать программный модуль «Личные дела студентов». Программный модуль предназначен для получения сведений о студентах сотрудниками деканата, профкома и отдела кадров. Сведения должны храниться в течение всего срока обучения студентов и использоваться при составлении справок и отчетов.

1. Написать тестовые сценарии для тестирования вашего приложения.

2. Определить классы эквивалентности и соответсвующий набор входных тестовых данных.

3. Провести тестирование в соответствии со сценарием и указать результаты.

# Тестовые сценарии

Тестовый сценарий: Проверка работы формы проверяющий id

#### **Цель**

Проверить, что при вводе ID в форму и нажатии кнопки "Войти", открывается соответствующая форма (студента или сотрудника) или не происходит никаких действий в случае, если ID невалидный.

#### **Предварительные условия**

* Открыта страница с формой "Форма Входа".

#### **Шаги тестирования**

1. **Тестирование четного ID**
   * **Данные:** Введите валидный четный ID (например, **2**).
   * **Ожидаемое действие:** Нажатие на кнопку "Войти" должно открыть форму студента.
2. **Тестирование нечетного ID**
   * **Данные:** Введите валидный нечетный ID (например, **3**).
   * **Ожидаемое действие:** Нажатие на кнопку "Войти" должно открыть форму сотрудника
3. **Тестирование не валидного ID (пустая строка)**
   * **Данные:** Оставьте поле ID пустым и нажмите "Войти".
   * **Ожидаемое действие:** Сообщение с просьбой ввести ID (форма не открывается).
4. **Тестирование не валидного ID (буквы)**
   * **Данные:** Введите нечисловую строку (например, abc).
   * **Ожидаемое действие:** Сообщение с просьбой ввести ID в корректной форме (форма не открывается).
5. **Тестирование ID с пробелами**
   * **Данные:** Введите значение с пробелами (например, 4).
   * **Ожидаемое действие:** Нажатие на кнопку "Войти" должно открыть форму студента (пробелы игнорируются).
6. **Тестирование отрицательного четного ID**
   * **Данные:** Введите отрицательный нечетный ID (например, -3).
   * **Ожидаемое действие:** Нажатие на кнопку "Войти" должно открыть форму сотрудника (знак игнорируется)
7. **Тестирование дробного ID**
   * **Данные:** Введите дробный ID (например, **-3.3**).
   * **Ожидаемое действие:** Нажатие на кнопку "Войти" должно попросить ввести корректный ID.

Тестовый сценарий: Проверка работы формы студента

#### **Цель**

Проверить, что при открытии формы студента на ней отображаются все нужные поля (ФИО, город рождения, средний балл, статус о работе). Так же эти поля не доступны для редактирования с основной формы, а для их изменения нужно нажать кнопку «редактировать», после чего открывается форма для редактирования этих данных. На ней есть кнопка «Сохранить», она должна закрыть форму редактирования.

#### **Предварительные условия**

* Открыта страница с формой "Форма Студента".

#### **Шаги тестирования**

1. **Тестирование заполнения полей**
   * **Данные:** Поля должны быть заполнены
   * **Ожидаемое действие:** При открытии формы в полях должны уже присутствовать данные
2. **Тестирование ручного ввода в поля**
   * **Данные:** Попробовать ввести данные в форме «Студент»
   * **Ожидаемое действие:** При попытки начать редактировать поля формы у вас не должно это получиться
3. **Тестирование работы кнопки «Редактировать»**
   * **Данные:** Нажмите кнопку «редактировать» внизу экрана
   * **Ожидаемое действие:** Открывается под-форма, позволяющая редактировать поля, отображаемые на основной форме
4. **Тестирование ввода данных на под-форме «Редактирование»**
   * **Данные:** Ввести данные в поля (ФИО, город рождения, средний балл, статус о работе)
   * **Ожидаемое действие:** Редактирование полей будет доступно, данные можно изменять
5. **Тестирование кнопки «Сохранить» на под-форме «Редактирование»**
   * **Данные:** Нажать кнопку «сохранить»
   * **Ожидаемое действие:** Нажатие на кнопку «сохранить» приведет к тому, что под-форма для редактирования закроется

# Классы эквивалентности

Класс 1: данные вводимые в форму проверки ID, что для студента, что для сотрудника, должны быть одного вида – целое число

Класс 2: данные, вводимые в форме студента, в разделе «редактировать», должны быть того же типа данных, что и данные выводимые на начальную форму студента

Класс 3: для получения справки или отчета, о каждом студенте, получаемые сотрудниками университета всегда одинаковые. Точнее, общий набор данных по нажатию на кнопку «справка» и «отчет» - одинаковы

# Тестирование

Проверка работы формы проверяющий id. Результаты:

* 1. Тестирование четного ID

Пройдено успешно – открывает форму студента

* 1. Тестирование нечетного ID

Пройдено успешно – открывает форму сотрудника

* 1. Тестирование пустой строки

Пройдено успешно – выводит сообщение с просьбой ввести ID

* 1. Тестирование строки, заполненной буквами

Пройдено успешно – выводит сообщение с просьбой ввести ваш ID в корректной форме

* 1. Тестирование строки с правильным ID и пробелами

Пройдено успешно – проблемы (что до, что после) игнорируются и число считывается правильно

* 1. Тестирование отрицательного, целого, четного ID

Пройдено успешно – знак числа не влияет на его обработку (четность и нечестность обрабатывается правильно)

* 1. Тестирование дробного ID

Пройдено успешно – выводит сообщение с просьбой ввести ваш ID в корректной форме

Проверка работы формы студента. Результаты:

1. Пройден успешно – при открытии формы (из формы «Форма входа») все поля уже открываются с данными о студенте
2. Пройден успешно – при попытке редактировать какое-либо поле (ФИО, город рождения, средний балл, статус о работе)
3. Пройден успешно – кнопка «Редактировать» открывает под-форму для редактирования данных
4. Пройден успешно – на под-форме можно редактировать поля (ФИО, город рождения, средний балл, статус о работе)
5. Пройден успешно – при нажатии на кнопку «сохранить» под-форма закрывается