**Государственное бюджетное профессиональное образовательное учреждение**

**«Нижегородский Губернский колледж»**

Методическая комиссия «Информатика и вычислительная техника»

Допущен к защите:

преподаватель

\_\_\_\_\_\_\_\_\_\_\_\_ Л.В. Мухина,

\_\_\_\_\_\_\_\_\_\_\_\_ Е.П. Голубева,

«21»\_\_марта\_\_2025 г.

**ОТЧЕТ** **ПО**

**УЧЕБНОЙ ПРАКТИКЕ**

ПМ.01 РАЗРАБОТКА МОДУЛЕЙ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ КОМПЬЮТЕРНЫХ СИСТЕМ

Руководитель: \_\_\_\_\_\_\_\_\_\_ Мухина Л.В., Голубева Е.П. 21.03.2025 г.

Студент: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Токарева Элина Олеговна 21.03.2025 г.

Специальность, группа: 09.02.07, 43П

Нижний Новгород

2025 г.

**Содержание**

[Введение 4](#_Toc192089419)

[Основная часть 5](#_Toc192089420)

[Модуль №1. Разработка, администрирование и защита баз данных 5](#_Toc192089421)

[Модуль №2: Разработка модулей программного обеспечения для компьютерных систем 6](#_Toc192089422)

[Модуль №3: Сопровождение и обслуживание программного обеспечения компьютерных систем 10](#_Toc192089423)

[Модуль №4: Осуществление интеграции программных модулей 17](#_Toc192089424)

[Заключение 18](#_Toc192089425)

[Приложения 19](#_Toc192089426)

[Приложение А. 19](#_Toc192089427)

[Приложение Б. 20](#_Toc192089428)

[Приложение В. 21](#_Toc192089429)

# Введение

Целью учебной практики является выполнение четырех сессий демонстрационного экзамена 2025 года, описанных в документе КОД 09.02.07-2-2025 Том 1, соответствующих модулям:

1. Разработка, администрирование и защита баз данных;
2. Разработка модулей программного обеспечения для компьютерных систем;
3. Сопровождение и обслуживание программного обеспечения компьютерных систем;
4. Осуществление интеграции программных модулей.

Каждый модуль предполагал выполнение конкретных задач, включая анализ требований, разработку программных решений, создание модульных тестов, описание тестовых сценариев и проектирование диаграмм.

В рамках третьей сессии описаны тестовые сценарии для процедуры редактирования партнера, демонстрирующие различные исходы работы алгоритма. Особое внимание уделено четвертой сессии, где требовалось реализовать метод для расчета количества материалов в виде библиотеки и разработать проект модульных тестов.

# Основная часть

## Модуль №1. Разработка, администрирование и защита баз данных

Получить ER-диаграмму средствами СУБД: ER-диаграмма должна быть представлена в формате PDF и содержать таблицы, связи между ними, атрибуты и ключи (типами данных на данном этапе можно пренебречь).

На основе описания предметной области создана база данных в СУБД Postgres для разрабатываемой системы. База данных находится в 3 нормальной форме с обеспечением ссылочной целостности.

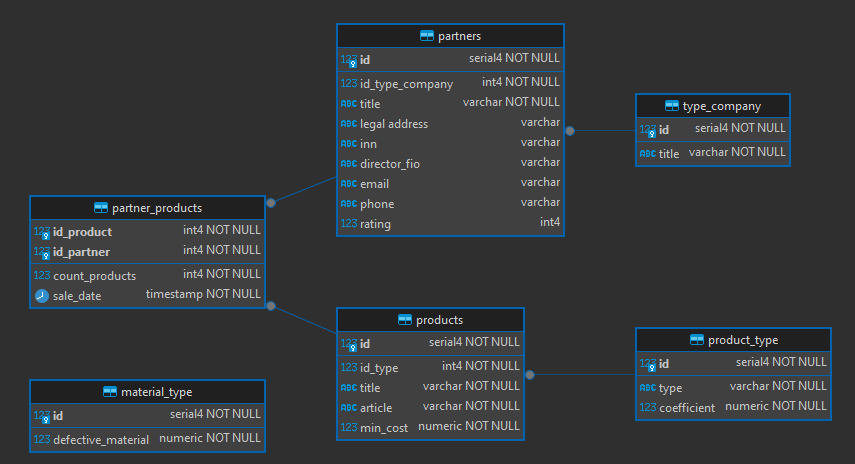


Рисунок №1 – Схема базы данных

Скрип базы данных представлен в приложении 1.

Заказчик системы предоставил файлы с данными (с пометкой import в ресурсах) для переноса в новую систему. Необходимо подготовить данные файлов для импорта и загрузить в разработанную базу данных.

## Модуль №2: Разработка модулей программного обеспечения для компьютерных систем

В рамках данного модуля была разработана подсистема для работы с партнерами компании, обеспечивающая следующие функции:

1. Просмотр списка партнеров

На главной форме приложения отображается список партнеров согласно макету, информация о которых хранится в базе данных. Цвета элементов определены в соответствии с руководством по стилю.

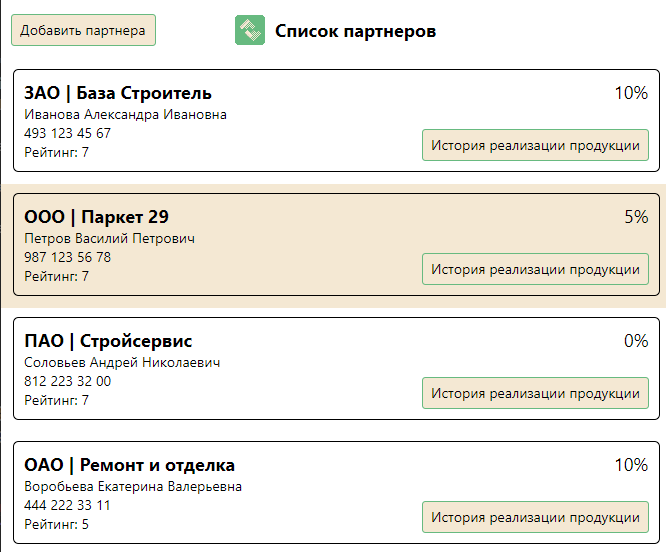


Рисунок №2 – Список партнеров

В конструкторе PartnersViewModel рассчитывается скидка для партнера на основе объема продаж и заполняется поле Discount для каждого элемента листа. Код конструктора находится в приложении А, Табл. 7.

1. Добавление/редактирование данных о партнере

Реализованы формы для добавления нового партнера и редактирования существующих данных. При нажатии на кнопку «Добавить партнера» открывается форма редактирования, для этого в классе создается новый объект модели. Код находится в приложении А, Табл. 8. Форма представлена на рисунке №3.

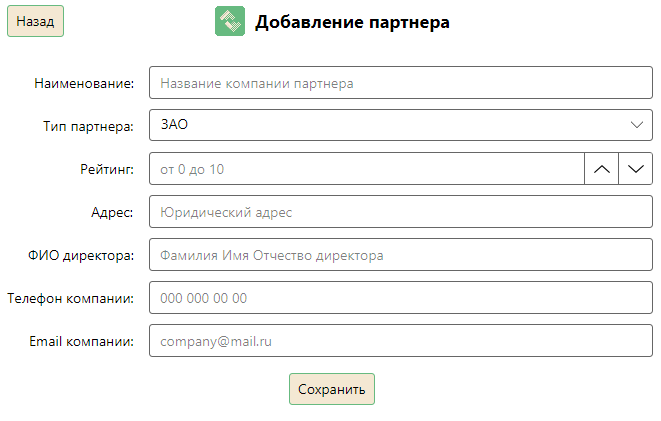


Рисунок №3 – Добавление партнера

При нажатии на элемент – открывается форма редактирования партнера, для этого в конструктор класса передается выбранный объект. Код находится в приложении А, Табл. 9. Форма представлена на рисунке №4.

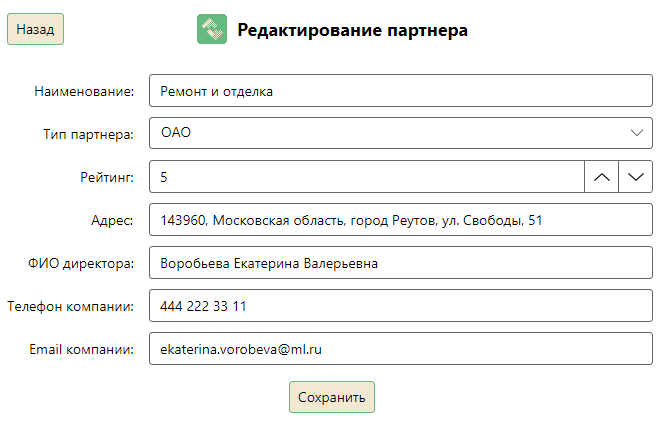


Рисунок №4 – Редактирование партнера

При добавлении или редактировании проверяются корректность введенных данных (обязательные поля, формат контактной информации).

1. Просмотр истории реализации продукции партнером.

Для каждого партнера доступна история реализации продукции, которая отображается в отдельной форме, при нажатии на кнопку «История реализации продукции». Код, получающий из базы данных историю конкретного партнера, находится в приложении А, Табл. 10. Форма представлена на рисунке №5.

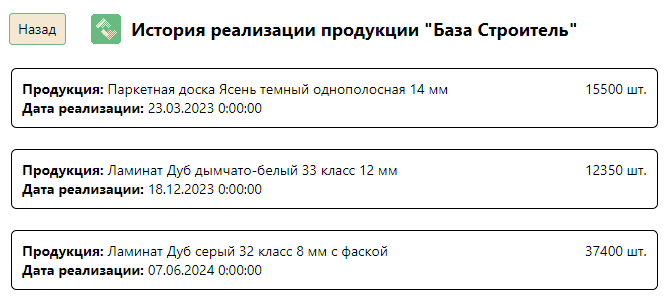


Рисунок №5 – История реализации продукции партнером

В ходе работы разработки выполнено тестирование модуля для проверки функциональности. Приложение корректно работает, аварийного завершения работы не происходит.

## Модуль №3: Сопровождение и обслуживание программного обеспечения компьютерных систем

Реализован последовательный пользовательский интерфейс, позволяющий перемещаться между окнами (страницами) приложения: Переход на форму добавления/редактирования партнера осуществляется с главной формы списка партнеров. Для возврата на предыдущую страницу используется кнопка «Назад».

Каждое окно имеет соответствующий заголовок, отражающий его назначение (например, «Список партнеров», «Добавление партнера», «Редактирование партнера»).

Реализована обработка исключительных ситуаций с помощью try-catch, код находится в приложении Б, Табл. 11. Пользователь уведомляется о совершенных ошибках или запрещенных действиях.

В приложении присутствуют всплывающие окна с соответствующими заголовками и пиктограммами (ошибка, предупреждение, информация). Текст сообщений содержит информацию об ошибках и рекомендации по их исправлению.

Окна уведомлений представлены на рисунках №6-8.

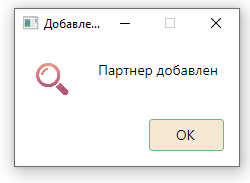


Рисунок №6 – Окно уведомления о добавлении партнера

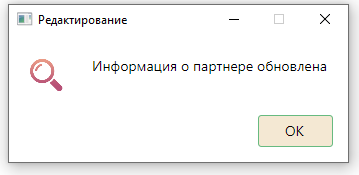


Рисунок №7 – Окно уведомления о редактировании партнера

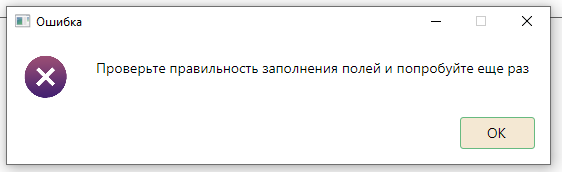


Рисунок №8 – Окно уведомления об ошибке

Также в рамках данной сессии описаны тестовые сценарии для процедуры редактирования партнера, демонстрирующие различные исходы работы алгоритма.

Таблица №1 – Аннотация теста

|  |  |
| --- | --- |
| **Название проекта** | Мастер пол |
| **Рабочая версия** | 1.0 |
| **Имя тестирующего** | Токарева Элина |
| **Дата(ы) теста** | 04.03.2025 |

Таблица №2 – Test-case #1

|  |  |
| --- | --- |
| **Test Case #** | TC\_EP\_1: edit partner test case #1 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Редактирование партнера с полной информацией |
| **Краткое изложение теста** | Тест редактирование партнера с полностью заполненной информацией |
| **Этапы теста** | 1. Нажать на нужного партнера  2. В окне редактирования партнера заполнить все поля измененными данными  3. Нажать кнопку «Сохранить» |
| **Тестовые данные** | Данные партнера |
| **Ожидаемый результат** | Отображение окна с уведомлением об обновлении данных и изменение данных в базе |
| **Фактический результат** | Отображение окна с уведомлением об обновлении данных и изменение данных в базе |
| **Статус** | Зачтен |
| **Предварительное условие** | Приложение запущено |
| **Постусловие** | Возврат пользователя обратно к списку партнеров |
| **Примечания/комментарии** | - |

Таблица №3 – Test-case #2

|  |  |
| --- | --- |
| **Test Case #** | TC\_EP\_2: edit partner test case #2 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Редактирование партнера с указанием отрицательного рейтинга |
| **Краткое изложение теста** | Тест редактирование партнера с указанием отрицательного рейтинга |
| **Этапы теста** | 1. Нажать на нужного партнера  2. В окне редактирования партнера заполнить все поля измененными данными  3. В поле «Рейтинг» ввести «-1»  4. Нажать кнопку «Сохранить» |
| **Тестовые данные** | Данные партнера |
| **Ожидаемый результат** | Поле рейтинг очищается и не дает сделать вставку числа менее 0 или более 10 |
| **Фактический результат** | Поле рейтинг очищается и не дает сделать вставку числа менее 0 или более 10 |
| **Статус** | Зачтен |
| **Предварительное условие** | Приложение запущено |
| **Постусловие** | - |
| **Примечания/комментарии** | - |

Таблица №4 – Test-case #3

|  |  |
| --- | --- |
| **Test Case #** | TC\_EP\_3: edit partner test case #3 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Редактирование партнера с указанием рейтинга нечисловыми значениями |
| **Краткое изложение теста** | Тест редактирование партнера с указанием рейтинга словом «десять» |
| **Этапы теста** | 1. Нажать на нужного партнера  2. В окне редактирования партнера заполнить все поля измененными данными  3. В поле «Рейтинг» ввести «десять»  4. Нажать кнопку «Сохранить» |
| **Тестовые данные** | Данные партнера |
| **Ожидаемый результат** | Поле рейтинг очищается и не дает сделать вставку нечислового значения |
| **Фактический результат** | Поле рейтинг очищается и не дает сделать вставку нечислового значения |
| **Статус** | Зачтен |
| **Предварительное условие** | Приложение запущено |
| **Постусловие** | - |
| **Примечания/комментарии** | - |

Таблица №5 – Test-case #4

|  |  |
| --- | --- |
| **Test Case #** | TC\_EP\_4: edit partner test case #4 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Редактирование партнера без информации |
| **Краткое изложение теста** | Тест редактирование партнера с незаполненной информацией |
| **Этапы теста** | 1. Нажать на нужного партнера  2. В окне редактирования партнера очистить все поля (которые возможны)  3. Нажать кнопку «Сохранить» |
| **Тестовые данные** | Данные партнера |
| **Ожидаемый результат** | Отображение окна ошибки с уведомлением о том, что не все данные заполнены |
| **Фактический результат** | Отображение окна ошибки с уведомлением о том, что не все данные заполнены |
| **Статус** | Зачтен |
| **Предварительное условие** | Приложение запущено |
| **Постусловие** | - |
| **Примечания/комментарии** | - |

Таблица №6 – Test-case #5

|  |  |
| --- | --- |
| **Test Case #** | TC\_UI\_5: user Interface test case #5 |
| **Приоритет тестирования** | Низкий |
| **Заголовок/название теста** | Кнопка редактирования изменяет цвет при нажатии |
| **Краткое изложение теста** | Тест изменения цвета кнопки «Сохранить» при нажатии |
| **Этапы теста** | 1. Нажать на нужного партнера  2. В окне редактирования партнера заполнить все поля измененными данными  3. Зажать кнопку «Сохранить» |
| **Тестовые данные** | Данные партнера |
| **Ожидаемый результат** | Кнопка «Сохранить» изменила цвет, пока удерживается пользователем |
| **Фактический результат** | Кнопка «Сохранить» изменила цвет, пока удерживается пользователем |
| **Статус** | Зачтен |
| **Предварительное условие** | Приложение запущено |
| **Постусловие** | - |
| **Примечания/комментарии** | - |

## Модуль №4: Осуществление интеграции программных модулей

Разработка метода GetQuantityForProduct для расчета количества материалов. Код метода представлен в приложении В. Табл 12. Метод принимает параметры:

* Идентификатор типа продукции (целое число).
* Идентификатор типа материала (целое число).
* Количество продукции (целое число).
* Параметры продукции (2 вещественных положительных числа).

Возвращает целое число – количество необходимого материала с учетом брака. Если в качестве параметров метода будут указаны несуществующие типы продукции, материалов или другие неподходящие данные, то метод возвращает -1.

Создан отдельный проект модульных тестов для тестирования метода:

Разработано 15 тестов: 10 тестов низкой сложности, код которых представлен в приложении В. Табл 13 и 5 тестов высокой сложности, код которых представлен в приложении В. Табл 14.

Результат тестов представлен на рисунке №9.

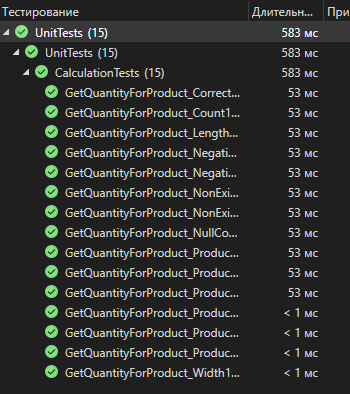


Рисунок №9 – Окно уведомления о добавлении партнера

# Заключение

В ходе выполнения задания успешно выполнены все поставленные задачи, соответствующие четырем модулям демонстрационного экзамена.

В рамках первой сессии проведен анализ требований к разработке, администрированию и защите баз данных, что позволило сформировать четкое понимание структуры и функциональности системы.

Во второй сессии разработаны модули программного обеспечения, соответствующие требованиям предметной области.

В третьей сессии описаны 5 тестовых сценариев для процедуры редактирования партнера, охватывающих различные варианты исходов работы алгоритма.

В четвертой сессии реализован метод для расчета количества материалов в виде библиотеки, а также разработан проект модульных тестов, включающий 15 методов (10 низкой сложности и 5 высокой сложности).

На основании анализа файла «Задание на проектирование» составлены диаграммы использования и деятельности, что позволило визуализировать процессы и взаимодействия в системе.

Все результаты работы систематизированы и представлены в репозитории с подробным описанием в файле README.md.

Выполнение данного задания позволило закрепить навыки работы с базами данных, разработки программных модулей, модульного тестирования и проектирования UML-диаграмм.

# Приложения

## Приложение А.

Таблица №7 – Код конструктора PartnersViewModel

|  |
| --- |
| public PartnersViewModel()  {  Partners = MainWindowViewModel.dbContext.Partners.Include(p => p.IdTypeCompanyNavigation)  .Include(p => p.PartnerProducts).ToList();  foreach (var item in Partners)  {  int sum = item.PartnerProducts.Sum(it => it.CountProducts);  if (sum < 10000)  item.Discount = 0;  else if (sum >= 10000 && sum < 50000)  item.Discount = 5;  else if (sum >= 50000 && sum < 300000)  item.Discount = 10;  else if (sum >= 300000)  item.Discount = 15;  }  } |

Таблица №8 – Код конструктора EditAddPartnerViewModel

|  |
| --- |
| public EditAddPartnerViewModel()  {  Heading = "Добавление партнера";  TypesCompany = MainWindowViewModel.dbContext.TypeCompanies.ToList();  Partner = new Partner();  Partner.IdTypeCompanyNavigation = TypesCompany.First();  } |

Таблица №9 – Код конструктора EditAddPartnerViewModel с аргументом

|  |
| --- |
| public EditAddPartnerViewModel(int IdPartner)  {  Heading = "Редактирование партнера";  TypesCompany = MainWindowViewModel.dbContext.TypeCompanies.ToList();  Partner = MainWindowViewModel.dbContext.Partners.Include(p => p.IdTypeCompanyNavigation).First(p => p.Id == IdPartner);  } |

Таблица №10 – Код конструктора HistoryViewModel

|  |
| --- |
| public HistoryViewModel(Partner partner)  {  Heading = $"История реализации продукции \"{partner.Title}\"";  Products = MainWindowViewModel.dbContext.PartnerProducts.Where(pp => pp.IdPartner == partner.Id).Include(pp => pp.IdProductNavigation).ToList();  } |

## Приложение Б.

Таблица №11 – Код асинхронной функции Save с обработкой ошибок

|  |
| --- |
| [RelayCommand]  public async Task Save()  {  try  {  if (Partner.Id == 0)  {  MainWindowViewModel.dbContext.Partners.Add(Partner);  MainWindowViewModel.dbContext.SaveChanges();  await MessageBoxManager.GetMessageBoxStandard("Добавление", "Партнер добавлен", ButtonEnum.Ok, Icon.Info).ShowAsync();  }  else  {  MainWindowViewModel.dbContext.Partners.Update(Partner);  MainWindowViewModel.dbContext.SaveChanges();  await MessageBoxManager.GetMessageBoxStandard("Редактирование", "Информация о партнере обновлена", ButtonEnum.Ok, Icon.Info).ShowAsync();  }  MainWindowViewModel.Instance.UserControl = new PartnersView();  }  catch  {  await MessageBoxManager.GetMessageBoxStandard("Ошибка", "Проверьте правильность заполнения полей и попробуйте еще раз", ButtonEnum.Ok, Icon.Error).ShowAsync();  }  } |

## Приложение В.

Таблица №12 – Код функции GetQuantityForProduct в библиотеке

|  |
| --- |
| public int GetQuantityForProduct(int productType, int materialType, int count, float width, float length)  {  double coefficient = 0;  double percentDefective = 0;  switch (productType)  {  case 1: coefficient = 1.1; break;  case 2: coefficient = 2.5; break;  case 3: coefficient = 8.43; break;  default: return -1;  }  switch (materialType)  {  case 1: percentDefective = 0.3; break;  case 2: percentDefective = 0.12; break;  default: return -1;  }  if (count <= 0 || width <= 0 || length <= 0) return -1;  double S = width \* length;  double countProducts = (S \* coefficient \* count);  double defective = (countProducts \* percentDefective) / 100; return (int)Math.Ceiling(countProducts + defective);  } |

Таблица №13 – Тесты низкой сложности

|  |
| --- |
| Calc calc = new Calc();  //10 методов низкой сложности  [TestMethod]  public void GetQuantityForProduct\_ProductType1MaterialType1\_CorrectResult()  {  int expected = 14895;  int result = calc.GetQuantityForProduct(1, 1, 15, 20, 45);  Assert.AreEqual(expected, result);  }  [TestMethod]  public void GetQuantityForProduct\_ProductType2\_MaterialType1\_CorrectResult()  {  int expected = 33852;  int result = calc.GetQuantityForProduct(2, 1, 15, 20, 45);  Assert.AreEqual(expected, result);  }  [TestMethod]  public void GetQuantityForProduct\_ProductType3MaterialType1\_CorrectResult()  {  int expected = 114147;  int result = calc.GetQuantityForProduct(3, 1, 15, 20, 45);  Assert.AreEqual(expected, result);  }  [TestMethod]  public void GetQuantityForProduct\_ProductType1MaterialType2\_CorrectResult()  {  int expected = 14868;  int result = calc.GetQuantityForProduct(1, 2, 15, 20, 45);  Assert.AreEqual(expected, result);  }  [TestMethod]  public void GetQuantityForProduct\_ProductType2MaterialType2\_CorrectResult()  {  int expected = 33791;  int result = calc.GetQuantityForProduct(2, 2, 15, 20, 45);  Assert.AreEqual(expected, result);  }  [TestMethod]  public void GetQuantityForProduct\_ProductType3MaterialType2\_CorrectResult()  {  int expected = 113942;  int result = calc.GetQuantityForProduct(3, 2, 15, 20, 45);  Assert.AreEqual(expected, result);  }  [TestMethod]  public void GetQuantityForProduct\_CorrectSmallValues\_CorrectTypeData()  {  int result = calc.GetQuantityForProduct(1, 2, 1, 1, 1);  Assert.IsInstanceOfType(result, typeof(int));  }  [TestMethod]  public void GetQuantityForProduct\_Count1LessCount100\_ResultTrue()  {  int count1 = calc.GetQuantityForProduct(1, 1, 1, 20, 45);  int count2 = calc.GetQuantityForProduct(1, 1, 100, 20, 45);  bool result = count1 < count2;  Assert.IsTrue(result);  }  [TestMethod]  public void GetQuantityForProduct\_Width10LessWidth40\_ResultTrue()  {  int count1 = calc.GetQuantityForProduct(1, 1, 15, 10, 45);  int count2 = calc.GetQuantityForProduct(1, 1, 15, 40, 45);  bool result = count1 < count2;  Assert.IsTrue(result);  }  [TestMethod]  public void GetQuantityForProduct\_Length10LessLength40\_ResultTrue()  {  int count1 = calc.GetQuantityForProduct(1, 1, 15, 20, 10);  int count2 = calc.GetQuantityForProduct(1, 1, 15, 20, 40);  bool result = count1 < count2;  Assert.IsTrue(result);  } |

Таблица №14 – Тесты высокой сложности

|  |
| --- |
| //5 методов высокой сложности  [TestMethod]  public void GetQuantityForProduct\_NonExistentMaterialType\_ErrorCode()  {  int expected = -1;  int result = calc.GetQuantityForProduct(0, 2, 15, 20, 45);  Assert.AreEqual(expected, result);  }  [TestMethod]  public void GetQuantityForProduct\_NonExistentProductType\_ErrorCode()  {  int expected = -1;  int result = calc.GetQuantityForProduct(1, 0, 15, 20, 45);  Assert.AreEqual(expected, result);  }  [TestMethod]  public void GetQuantityForProduct\_NegativeCount\_ErrorCode()  {  int expected = -1;  int result = calc.GetQuantityForProduct(1, 2, -10, 20, 45);  Assert.AreEqual(expected, result);  }  [TestMethod]  public void GetQuantityForProduct\_NegativeWidthAndLength\_ErrorCode()  {  int expected = -1;  int result = calc.GetQuantityForProduct(1, 2, 15, -2, -15);  Assert.AreEqual(expected, result);  }  [TestMethod]  public void GetQuantityForProduct\_NullCount\_ReturnIsNotNull()  {  int result = calc.GetQuantityForProduct(1, 2, Convert.ToInt32(null), 20, 45);  Assert.IsNotNull(result);  } |