**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Сибирский государственный университет науки и технологий**

**имени академика М.Ф. Решетнева»**

ИИТК

(институт/факультет)

**ДНЕВНИК-ОТЧЕТ**

**ПРОХОЖДЕНИЯ ПРАКТИКИ ОБУЧАЮЩИМСЯ**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Рылов\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Артем\_Евгеньевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(имя, отчество)

Направление \_\_09.03.04 «Программная инженерия»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Группа \_\_\_\_\_\_\_БПИ18-01\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Вид практики\_\_УЧЕБНАЯ ПРАКТИКА\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Тип практики\_\_Практика по получению первичных профессиональных умений и навыков, в том числе первичных умений и навыков научно-исследовательской деятельноси\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Сроки прохождения практики с \_\_\_\_\_13.07.2020\_\_\_\_ по \_\_\_\_25.07.2020\_\_\_\_\_

Оценка кафедры по результатам прохождения практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_Моргунова Ольга Николаевна\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_25.07.2020\_\_

ФИО руководителя практики от Университета подпись дата

Красноярск 2020 г.

**ПАМЯТКА ОБУЧАЮЩЕМУСЯ**

**При оформлении на практику обучающийся обязан иметь следующие документы:**

* паспорт;
* трудовую книжку, за исключением случаев, когда трудовой договор заключается впервые;
* страховое свидетельство обязательного пенсионного страхования (СНИЛС);
* документы воинского учета – для военнообязанных и лиц, подлежащих призыву на военную службу;
* идентификационный номер налогоплательщика (ИНН);
* предписание и справку-форму для обучающихся, проходящих практику на режимных предприятиях;
* методические указания по организации практики;
* направление от Университета на данное предприятие;
* настоящий дневник-отчет с заполненными разделами.

**В период прохождения практики обучающийся обязан:**

* прибыть на место прохождения практики в сроки, установленные календарным учебным графиком;
* выполнить индивидуальное задание, выданное руководителем практики;
* соблюдать требования охраны труда и пожарной безопасности, правил внутреннего трудового распорядка;
* вести дневник-отчет практики (для обучающихся по программам бакалавриата и специалитета), где отражать ход выполнения индивидуального задания, описывать выполненную работу, и оформить полученные результаты в соответствии с требованиями, установленными программой практики и методическими указаниями.

**По окончании практики обучающийся обязан:**

* сдать на предприятие всю документацию, которой он пользовался в период практики;
* получить справку-форму о допуске на режимное предприятие для возврата её в 1-й отдел университета (для режимных предприятий);
* сдать пропуск на предприятие;
* своевременно оформить и сдать дневник-отчет по итогам практики руководителю от Университета.

**Обучающемуся необходимо знать:**

* при подведении итогов работы обучающегося на практике принимается во внимание оценка результатов прохождения практики, данная руководителем практики от предприятия, качество дневника-отчета;
* неудовлетворительные результаты промежуточной аттестации по практике или непрохождение промежуточной аттестации по практике при отсутствии уважительных причин признаются академической задолженностью;
* обучающиеся, не ликвидировавшие в установленные сроки академическую задолженность, отчисляются из Университета приказом проректора по образовательной деятельности по представлению директора института (декана факультета, директора филиала) как не выполнившие обязанности по добросовестному освоению образовательной программы и выполнению учебного плана.

Ознакомлен\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата)

**Наименование организации, в которой обучающийся проходит практику:**\_ "ФГБОУ ВО «Сибирский государственный университет науки и технологий имени академика М.Ф. Решетнева», НИЛ ИИТК"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Руководителем практики от Университета назначен:**

\_\_\_Моргунова Ольга Николаевна\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, имя, отчество)

\_\_\_Доцент каф. ИВТ.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(должность на кафедре)

Контактный телефон \_\_\_89135657334\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Руководителем практики от Организации назначен:**

\_\_\_ Моргунова Ольга Николаевна\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, имя, отчество)

\_\_\_ Доцент каф. ИВТ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(должность в цехе, отделе)

Контактный телефон \_\_\_89135657334\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата фактического прибытия обучающегося в организацию \_\_13.07.2020\_\_\_\_

**М. П.**

Дата фактического убытия обучающегося из организации \_\_\_\_25.07.2020\_\_\_\_

Вводный инструктаж провел: «\_13\_»\_\_\_июля\_\_\_2020г.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(должность)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(ФИО, подпись)

Инструктаж на рабочем месте провел: «\_13\_»\_\_ июля\_\_\_2020г.

\_\_\_\_ Зав. каф ИВТ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(должность)

\_\_\_\_ Фоворская Маргарита Николаевна \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(ФИО, подпись)

Инструктаж на новом рабочем месте провел: «\_\_»\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(должность)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(ФИО, подпись)

**Индивидуальное задание на практику**

|  |  |
| --- | --- |
| **№ п/п** | **Наименование работ** |
| 1 | Ознакомление с индивидуальным заданием согласно варианту |
| 2 | Изучение стратегий тестирования программ |
| 3 | Выбор функций для тестирования и выбор конкретных методов тестирования на основе стратегии «белого ящика» |
| 4 | Выбор модулей для тестирования на основе стратегии «черного ящика» |
| 5 | Разработка тестовых программ |
| 6 | Оформление отчета |
| 7 | Сдача отчета и дневника на проверку преподавателю |
| 8 | Защита отчета |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Задание выдал: Руководитель практики от Университета

\_\_\_\_\_Моргунова Ольга Николаевна\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Ф.И.О., подпись, дата)

Задание согласовал: Руководитель практики от Организации

\_\_\_\_\_Моргунова Ольга Николаевна\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Ф.И.О., подпись, дата)

**Рабочий график проведения практики**

|  |  |  |
| --- | --- | --- |
| **Период** | **Краткое описание выполняемых работ** | **Отметка руководителя от Организации о качестве выполнения работ** (отлично/ хорошо/ удовлетворительно) |
| 13.07.2020 | Ознакомление с рекомендованной литературой по теме «Методы тестирования программного обеспечения» |  |
| 14.07.2020 | Выбор конкретных методов на основе тестирования стратегии «белого ящика» |  |
| 15.07.2020 | Выбор методов тестирования на основе стратегии «черного ящика» |  |
| 16.07.2020 | Выбор функций для тестирования на основе стратегии «белого ящика» |  |
| 17.07.2020 | Выбор модулей для тестирования на основе стратегии «черного ящика» |  |
| 18.07.2020-20.07.2020 | Программная реализация тестовых программ |  |
| 21.07.2020-23.07.2020 | Оценка полученных результатов тестирования |  |
| 24.07.2020 | Оформление отчета по практике |  |
| 25.07.2020 | Защита разработанной программы и отчета по практике |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Руководитель практики от Организации

\_\_\_\_\_\_\_Моргунова Ольга Николаевна\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Ф.И.О., подпись, дата)

**Отчет о прохождении \_\_\_\_\_**\_УЧЕБНОЙ**\_\_\_\_\_\_ практики**

(*вид практики*)

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 7](#_Toc46149511)

[ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ 8](#_Toc46149512)

[ГЛАВА 1 10](#_Toc46149513)

[ГЛАВА 2 18](#_Toc46149514)

[ЗАКЛЮЧЕНИЕ 23](#_Toc46149515)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 24](#_Toc46149516)

ВВЕДЕНИЕ

Учебная практика охватывает перечень вопросы, которые связанны с разработкой программного обеспечения и его поддержания: с анализом требований, проектированием архитектуры программы, проектированием модулей, реализацией проекта в исходных кодах, тестировании программ. Важное внимание уделяется самостоятельному выбору студентом предметной области, умению самостоятельно проводить сбор, анализ научно-технической информации, изучать отечественный и зарубежный опыт в данной предметной области.

Цель прохождения практики – закрепление теоретических знаний в области информационных технологий, полученных за время обучения, и развитие практических навыков в создании компьютерных программ.

Задачами практики являются:

– изучение состава, структуры, принципов реализации и функционирования информационных технологий, используемых при создании информационных систем;

–  изучение базовых и прикладных информационных технологий;

–  изучение инструментальных средств информационных технологий;

–  изучение основных этапов создания программного продукта;

–  формирование у студентов навыков ведения самостоятельной работы с

технической документацией.

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

Для выполнения индивидуального задания по учебной практике в качестве объекта тестирования выбрано приложения под операционную систему iOS, разработанное в ходе выполнения курсовой работы. Данное приложение создано для узнавания актуального расписания занятий СибГУ.

Приложение было разработано на языке программирования Swift. В нем реализован следующий функционал: загрузка информации о преподавателях/группах/кабинетах и их расписание занятий, хранение этой информации в базе данных на телефоне, отображение этой информации в удобном виде, поиск по этой информации.

Приложение позволяет открыть расписание любого преподавателя, группы, кабинета и просматривать его без доступа к интернету (предварительно загрузив его), даты занятий всегда будут актуальны. Так же есть возможность узнать информацию, например, электронную почту, интересующей группы.

*Цель практики* – получить навыки тестирования программ на основе различных стратегий и методов, в процессе тестирования использовать знания и навыки, полученные в 1-4 семестрах.

*Задачи:*

1) Изучить литературу по тестированию программ.

2) Выбрать в качестве объекта тестирования программу.

3) Ознакомиться со стратегией тестирования «белый ящик». Выбрать функции для тестирования, выбрать методику тестирования, описать методику тестирования, разработать тестовую программу для тестирования функций, выполнить тестирование функций.

4) Ознакомиться со стратегией тестирования «черный ящик». Выбрать модули для тестирования, описать методику тестирования, выполнить тестирование модулей.

5) Подвести итог тестирования. Представить процесс тестирования в виде таблицы, в которой необходимо отразить: вводимые данные или выполняемые действия пользователя, реакцию программы, оценку правильности реакции программы.

ГЛАВА 1

Тестирование стратегией «белый ящик» выполняется, когда необходимо обнаружить проблемы во внутренней структуре программы. Предполагается, что человек, тестирующий программу, хорошо знает логику программы и все ее внутренние механизмы, которые будут поддаваться тестированию. Следовательно, такое тестирование не может быть выполнено обычным пользователем. Общая задача тестирования стратегией «белый ящик» - обеспечить проверку каждого шага алгоритма программы.

Главное преимущество всех типов стратегий тестирования стратегией «белый ящик» состоит в том, что при тестировании принимается во внимание структура всей программы (модуля), что облегчает локализацию ошибок.

Тестирование «белым ящиком» будет произведено методом покрытия решений. Суть этого метода в том, что тестовые наборы должны содержать граничные значения, обычные значения, отсутствующие значения и некорректные.

В языке Swift, на котором будет проводиться тестирование, есть встроенная библиотека для проведения модульного тестирования, или юнит-тестирования. Она позволяет писать стандартизированные тесты для программ, понятные и легко читаемые для многих программистов.

Суть юнит-тестирования состоит в написании тестов для каждого нетривиального метода или функции. Запуская эти тесты после каждого изменения с коде программы можно проверить, не привело ли это изменение к некорректной работе программы, и быстро отреагировать на появившуюся ошибку.

Для тестирования стратегией «белый ящик» были выбраны функции, предназначенные для работы с базой данных, которые записывают и считывают расписание занятий, а также конвертируют его в вид, пригодный для использования в приложении.

Приведем пример тестирования только для методов, работающих с расписанием групп, упустив реализацию тестирования методов для работы с преподавателями и кабинетами, так как они схожи.

Так как для составления, записи, чтения, и, соответственно, тестирования расписания занятий потребуется некоторое количество объектов групп, преподавателей и кабинетов, запишем в базу данных по 100 объектов каждого класса, и сформируем эти объекты по такому принципу: все поля будут содержать его порядковый номер (идентификатор), чтобы легко можно было идентифицировать объект и понять насколько правильно проходит процесс. Справиться с этой задачей помогут функции write100Groups(), write100Professors(), write100Places() (листинг 1).

Листинг 1- Функции для заполнения базы данных объектами

**@testable** **import** Timetable

**class** TestHepler {

**static** **func** write100Groups() {

**for** i **in** 1...100 {

**let** group = RGroup()

group.id = i

group.name = "\(i) groupName"

group.email = "\(i) groupEmail"

DataManager.shared.write(group: group)

}

}

**static** **func** write100Professors() {

**for** i **in** 1...100 {

**let** professor = RProfessor()

professor.id = i

professor.name = "\(i) professorName"

professor.email = "\(i) professorEmail"

professor.phone = "\(i) professorPhone"

professor.department = "\(i) professorDepartmert"

DataManager.shared.write(professor: professor)

}

}

**static** **func** write100Places() {

**for** i **in** 1...100 {

**let** place = RPlace()

place.id = i

place.name = "\(i) placeName"

DataManager.shared.write(place: place)

}

}

}

Запустим их единожды, так как состояние базы данных сохраняется между запусками приложения и тестов.

Посмотрев на рисунок 1, можно убедиться, что запись прошла успешно.

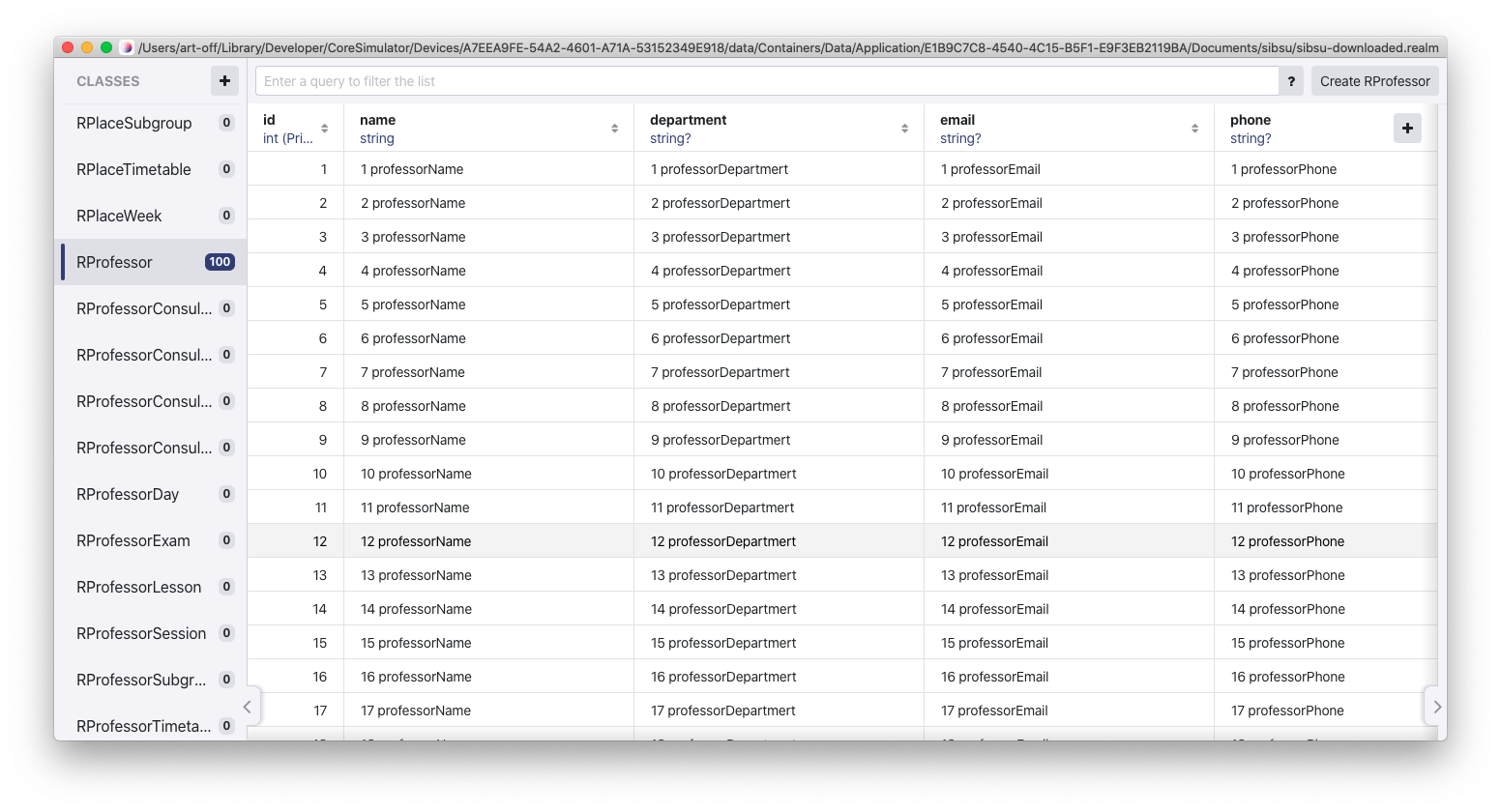


Рисунок 1 – Принцип заполнения базы данных

В первую очередь тестирование пройдет функция конвертации расписания занятий из вида, в котором она хранится в базе данных, в вид в котором используется в приложении. Они отличаются тем, что в виде для хранения в базе данных используются индексы преподавателей, групп, кабинетов и типов занятий (лекция, практика, лабораторная работа), в виде для использования в приложении же никаких индексов нет, а лишь полные названия групп и кабинетов, полные имена преподавателей и понятное для обычного человека определения типа занятия. Исходных код функции приведен ниже (листинг 2).

Листинг 2 – Функция конвертации расписания занятий

**func** convertGroupTimetable(from timetable: RGroupTimetable, groupName: String) -> GroupTimetable {

**var** groupWeeks = [GroupWeek]()

// пробегаемся по всем неделям (по дву)

**for** week **in** timetable.weeks {

// заполняем массив дней nil, потом если будут учебные дни в этой недели - заменю значение

**var** groupDays: [GroupDay?] = [**nil**, **nil**, **nil**, **nil**, **nil**, **nil**]

// пробегаемся во всем дням недели

**for** day **in** week.days {

**var** groupLessons = [GroupLesson]()

// пробегаемся по всем занятиям дня

**for** lesson **in** day.lessons {

**var** groupSubgroups = [GroupSubgroup]()

// пробегаемся по всех подргуппам занятия

**for** subgroup **in** lesson.subgroups {

**var** professorsName = [String]()

// берем имя преподавателя из БД с помощью id

**for** id **in** subgroup.professors {

**let** optionalProfessor = DataManager.shared.getProfessor(withId: id)

**if** **let** professor = optionalProfessor {

professorsName.append(professor.name)

} **else** {

professorsName.append("Неопознанный")

}

}

// берем имя кабинета с помощью id

**let** optionalPlace = DataManager.shared.getPlace(withId: subgroup.place)

**let** placeName: String

**if** **let** place = optionalPlace {

placeName = place.name

} **else** {

placeName = "Неопознанное"

}

// копируем подргуппу

**let** groupSubgroup = GroupSubgroup(

number: subgroup.number,

subject: subgroup.subject,

type: subgroupType[subgroup.type] ?? "(Неопознанный)",

professors: professorsName,

professorsId: Array(subgroup.professors),

place: placeName,

placeId: subgroup.place)

groupSubgroups.append(groupSubgroup)

}

// добавляем занятие в массив занятий

**let** groupLesson = GroupLesson(time: lesson.time, subgroups: groupSubgroups)

groupLessons.append(groupLesson)

}

**let** groupDay = GroupDay(lessons: groupLessons)

// проверяем, подходит ли number для вставки в массив groupDays (0-понедельник, 5-суббота)

**if** day.number >= 0 && day.number <= 5 {

// заменяем nil

groupDays[day.number] = groupDay

}

}

**let** groupWeek = GroupWeek(days: groupDays)

groupWeeks.append(groupWeek)

}

**let** groupTimetable = GroupTimetable(groupId: timetable.groupId, groupName: groupName, weeks: groupWeeks)

**return** groupTimetable

}

Для упрощения тестирования создадим функцию, которая будет создавать расписания занятий с нужными нам параметрами (листинг 3).

Листинг 3 – Функция, создающая объекты расписания занятий

**static** **func** createGroupTimetable(withPlaceId placeId: Int, type: Int, professorsId: [Int]) -> RGroupTimetable {

**let** groupTimetable = RGroupTimetable()

groupTimetable.groupId = 1

**let** groupWeek1 = RGroupWeek()

groupWeek1.number = 0

**let** groupDay1 = RGroupDay()

groupDay1.number = 0

**let** groupLesson1 = RGroupLesson()

groupLesson1.time = "time0"

**let** groupSubgroup = RGroupSubgroup()

groupSubgroup.number = 0

groupSubgroup.place = placeId

groupSubgroup.subject = "sub1"

groupSubgroup.type = type

**for** professorId **in** professorsId {

groupSubgroup.professors.append(professorId)

}

groupLesson1.subgroups.append(groupSubgroup)

groupDay1.lessons.append(groupLesson1)

groupWeek1.days.append(groupDay1)

groupTimetable.weeks.append(groupWeek1)

**return** groupTimetable

}

При отсутствии или некорректном значении идентификатора преподавателя, группы или кабинета должен возвращать «Неопознанный», такая же тактика с неизвестным номером типа занятий. С приведенном коде теста все случаи возможных «поломок» приведены (листинг 4).

Листинг 4 – Тестирование конвертации расписания занятий

**func** testConvertationTimetable() {

// MARK: **Нормальные входные данные**

**var** groupTimetableDB = TestHepler.createGroupTimetable(withPlaceId: 1, type: 2, professorsId: [5, 6])

// Конвертируем

**var** groupTimetableApp = Converter.shared.convertGroupTimetable(from: groupTimetableDB, groupName: "groupName1")

// Проверяем имя и id

XCTAssertEqual(groupTimetableApp.groupName, "groupName1")

XCTAssertEqual(groupTimetableApp.groupId, groupTimetableDB.groupId)

**let** lessonApp = groupTimetableApp.weeks[0].days[0]!.lessons[0]

**let** lessonDB = groupTimetableDB.weeks[0].days[0].lessons[0]

// Проверяем время

XCTAssertEqual(lessonApp.time, lessonDB.time)

**let** subgroupApp = lessonApp.subgroups[0]

**let** subgroupDB = lessonDB.subgroups[0]

// Проверяем номер недели

XCTAssertEqual(subgroupApp.number, subgroupDB.number)

// Проверяем кабинет, взятый из базы данных

XCTAssertEqual(subgroupApp.place, "1 placeName")

XCTAssertEqual(subgroupApp.placeId, subgroupDB.place)

// Проверяем преподавателей, взятых из базы данных

XCTAssertEqual(subgroupApp.professors[0], "5 professorName")

XCTAssertEqual(subgroupApp.professorsId[0], subgroupDB.professors[0])

XCTAssertEqual(subgroupApp.professors[1], "6 professorName")

XCTAssertEqual(subgroupApp.professorsId[1], subgroupDB.professors[1])

// Проверяем название предмета

XCTAssertEqual(subgroupApp.subject, subgroupDB.subject)

**let** subgroupType = [

0: "(Лекция)",

1: "(Практика)",

2: "(Лабораторная работа)",

]

// Проверяем тип занятия

XCTAssertEqual(subgroupApp.type, subgroupType[subgroupDB.type])

// MARK: **Несуществующий id кабинета**

groupTimetableDB = TestHepler.createGroupTimetable(withPlaceId: 245, type: 2, professorsId: [5, 6])

groupTimetableApp = Converter.shared.convertGroupTimetable(from: groupTimetableDB, groupName: "groupName1")

XCTAssertEqual(groupTimetableApp.weeks[0].days[0]!.lessons[0].subgroups[0].place, "Неопознанное")

// MARK: **Несуществующий тип занятия**

groupTimetableDB = TestHepler.createGroupTimetable(withPlaceId: 1, type: 4, professorsId: [5, 6])

groupTimetableApp = Converter.shared.convertGroupTimetable(from: groupTimetableDB, groupName: "groupName1")

XCTAssertEqual(groupTimetableApp.weeks[0].days[0]!.lessons[0].subgroups[0].type, "(Неопознанный)")

// MARK: **Несуществующий преподаватель**

groupTimetableDB = TestHepler.createGroupTimetable(withPlaceId: 1, type: 2, professorsId: [102, 6])

groupTimetableApp = Converter.shared.convertGroupTimetable(from: groupTimetableDB, groupName: "groupName1")

XCTAssertEqual(groupTimetableApp.weeks[0].days[0]!.lessons[0].subgroups[0].professors[0], "Неопознанный")

XCTAssertEqual(groupTimetableApp.weeks[0].days[0]!.lessons[0].subgroups[0].professors[1], "6 professorName")

// MARK: **Некорректный id кабинета**

groupTimetableDB = TestHepler.createGroupTimetable(withPlaceId: -11, type: 2, professorsId: [5, 6])

groupTimetableApp = Converter.shared.convertGroupTimetable(from: groupTimetableDB, groupName: "groupName1")

XCTAssertEqual(groupTimetableApp.weeks[0].days[0]!.lessons[0].subgroups[0].place, "Неопознанное")

// MARK: **Некорректный id преподавателя**

groupTimetableDB = TestHepler.createGroupTimetable(withPlaceId: 1, type: 2, professorsId: [-234, 6])

groupTimetableApp = Converter.shared.convertGroupTimetable(from: groupTimetableDB, groupName: "groupName1")

XCTAssertEqual(groupTimetableApp.weeks[0].days[0]!.lessons[0].subgroups[0].professors[0], "Неопознанный")

XCTAssertEqual(groupTimetableApp.weeks[0].days[0]!.lessons[0].subgroups[0].professors[1], "6 professorName")

}

Мы можем проверить правильность взятия преподавателя, группы и кабинета из базы данных, так как знаем те объекты, которые хранятся под определенными индексами.

Как можно убедиться из рисунка 2, тесты пройдены без ошибок, то есть функции правильно обработали все возможные ситуации.

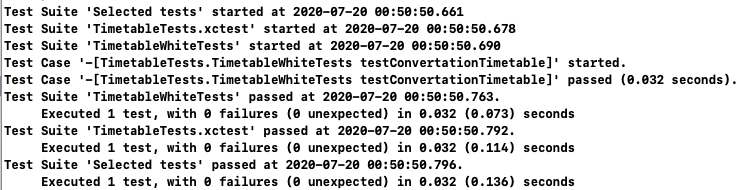


Рисунок 2 – Результаты тестирования конвертации

Теперь протестируем функции записи и чтения расписания занятий из базы данных. За это отвечает функции write(groupTimetable: RGroupTimetable) и getTimetable(forGroupId: Int). Ниже приведен исходный код этих функций (листинг 5 и 6).

Листинг 5 – Функция записи в базу данных

**func** write(groupTimetable: RGroupTimetable) {

**try**? userRealm.write {

userRealm.add(groupTimetable, update: .modified)

}

}

Листинг 6 – Функция чтения из базы данных

**func** getTimetable(forGroupId groupId: Int) -> GroupTimetable? {

**let** optionalTimetable = userRealm.object(ofType: RGroupTimetable.**self**, forPrimaryKey: groupId)

**let** optionalGroup = downloadedRealm.object(ofType: RGroup.**self**, forPrimaryKey: groupId)

**guard** **let** timetable = optionalTimetable **else** { **return** **nil** }

**guard** **let** group = optionalGroup **else** { **return** **nil** }

**let** groupTimetable = Converter.shared.convertGroupTimetable(from: timetable, groupName: group.name)

**return** groupTimetable

}

Будут проверены 4 случая: объект существует, объект не существует, объект был перезаписан, некорректный индекс. Снизу приведен код тестов (листинг 7).

Листинг 7 – Тестирование записи и чтения из базы данных

**func** testWriteAndGetToRealm() {

// MARK: **Нормальные входные данные**

// Создаем расписание с индексом 1 и временем первого занятия 11:11

**var** groupTimetableDB1 = TestHepler.createGroupTimetable(withPlaceId: 1, type: 2, professorsId: [-234, 6])

groupTimetableDB1.groupId = 1

groupTimetableDB1.weeks[0].days[0].lessons[0].time = "11:11"

// Записываем расписание с индексом 1

DataManager.shared.write(groupTimetable: groupTimetableDB1)

// Считываем расписание с индексом 1

**var** groupTimetableDB2 = DataManager.shared.getTimetable(forGroupId: 1)

// Проверяем равность этих расписаний

XCTAssertEqual(groupTimetableDB2?.groupId, 1)

XCTAssertEqual(groupTimetableDB2!.weeks[0].days[0]!.lessons[0].time, "11:11")

// MARK: **Несуществующие данные**

groupTimetableDB2 = DataManager.shared.getTimetable(forGroupId: 4)

XCTAssertNil(groupTimetableDB2)

// MARK: **Перезаписанные данные**

groupTimetableDB1 = TestHepler.createGroupTimetable(withPlaceId: 1, type: 2, professorsId: [-234, 6])

groupTimetableDB1.groupId = 1

groupTimetableDB1.weeks[0].days[0].lessons[0].time = "22:22"

DataManager.shared.write(groupTimetable: groupTimetableDB1)

groupTimetableDB2 = DataManager.shared.getTimetable(forGroupId: 1)

XCTAssertEqual(groupTimetableDB2?.groupId, 1)

XCTAssertEqual(groupTimetableDB2!.weeks[0].days[0]!.lessons[0].time, "22:22")

// MARK: **Некорректные данные**

groupTimetableDB2 = DataManager.shared.getTimetable(forGroupId: -22)

XCTAssertNil(groupTimetableDB2)

}

Для более точной идентификации (кроме индекса) используется время первого занятия, чтобы была большая уверенность, что это то самое расписание занятий.

После проведения тестов, и получения результатов, которые видно на рисунке 3, можно сказать, что функции записи и чтения из базы данных работают корректно при рассмотренных случаях.

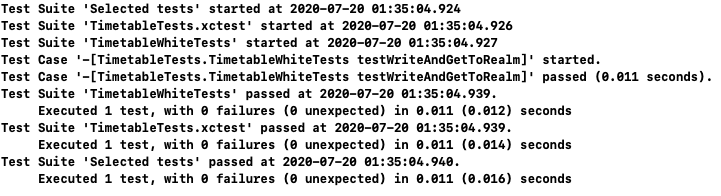


Рисунок 3 – Результаты тестирования работы с базой данных

После проведения тестов, и получения результатов, которые видно на рисунке 3, можно сказать, что функции записи и чтения из базы данных работают корректно при рассмотренных случаях.

ГЛАВА 2

Тестирование стратегией «черный ящик» представляет из себя метод тестирования, при котором функциональность исследуется без знания внутреннего устройства программы и деталей реализации. Тесты стратегией «черный ящик» пишут с опором на требования и спецификации, которым должна удовлетворять программа на конечном этапе разработки.

Плюсы данного метода тестирования:

- возможность быстрого выявления ошибок в функциональных спецификациях программы;

- тестировать может любой человек, даже обычный пользователь;

- выявления недостаточной производительности системы;

- можно составить тест-кейсы сразу после составления спецификаций и требований.

«Полное покрытие тестами» программного продукта в большинстве случаев неразумно, а иногда и невозможно, ведь для этого нужно проверить все данные, с которыми будет работать программа. Поэтому, часто применяется «частичное покрытие тестами», при таком подходе не нужно проверять все данные, которые могут пойти на вход программе, а лишь какой-то небольшой набор, с помощью которого можно определить возможное недочеты в программном продукте.

В тестируемом приложении немалую часть функциональности занимает работа с датами, так как верное отображения даты занятия в расписании – очень важно. В связи с этим, тестирование методом «черный ящик» будет проведено с модулями, которые отвечают за работу с датой, а если подробней, то получение даты следующего и предыдущего выбранного дня недели, узнавание четности недели (сравнивая с официальным расписанием СибГУ), а так же номер дня недели от даты.

Для удобности сравнения дат была создана функция, которая сравнивает только год, месяц и день от текущей даты. Это необходимо, так как в Swift объект класса Date имеет так же поля для хранения часов, минут и секунд, но, исходя из задачи, нам не важны эти поля. Код приведен ниже (листинг 8).

Листинг 8 – Функция для сравнения дат

**static** **func** equalDate(date1: Date, yearDate2: Int, monthDate2: Int, dayDate2: Int) {

// Год

XCTAssertEqual(Calendar.current.component(.year, from: date1), yearDate2)

// Месяц

XCTAssertEqual(Calendar.current.component(.month, from: date1), monthDate2)

// День

XCTAssertEqual(Calendar.current.component(.day, from: date1), dayDate2)

}

Первым тестирование пройдет модуль с помощью которого есть возможность получать предыдущую или следующую дату от выбранного дня недели. Способом тестированием данного модуля был выбран «анализ граничных значений», суть которого состоит в выборе граничных дат, которые начинают или заканчивают год, неделю или месяц. Особое внимание будет уделено получению даты следующего и предыдущего понедельника, так как только поиск этого дня недели используется внутри приложения. Код тестов приведен ниже (листинг 9).

Листинг 9 – Тестирование получение даты следующего и предыдущего выбранного дня недели

**func** testNextAndPreviousWeekday() {

// MARK: **Стартовая дата (в начале недели)**

**let** date\_20\_07\_2020 = Calendar.current.date(byAdding: DateComponents(year: 50, month: 6, day: 19), to: Date(timeIntervalSince1970: 0))!

// Следующий понедельник с начала недели

TestHepler.equalDate(date1: date\_20\_07\_2020.next(.monday), yearDate2: 2020, monthDate2: 7, dayDate2: 27)

// Предыдущая пятница с начала недели

TestHepler.equalDate(date1: date\_20\_07\_2020.previous(.friday), yearDate2: 2020, monthDate2: 7, dayDate2: 17)

// MARK: **Дата в конце недели**

**let** date\_26\_07\_2020 = Calendar.current.date(byAdding: DateComponents(year: 50, month: 6, day: 25), to: Date(timeIntervalSince1970: 0))!

// Следующий понедельник с конца недели

TestHepler.equalDate(date1: date\_26\_07\_2020.next(.monday), yearDate2: 2020, monthDate2: 7, dayDate2: 27)

// Предыдущий понедельник с конца недели

TestHepler.equalDate(date1: date\_26\_07\_2020.previous(.monday), yearDate2: 2020, monthDate2: 7, dayDate2: 20)

// MARK: **Дата в начале месяца**

**let** date\_01\_07\_2020 = Calendar.current.date(byAdding: DateComponents(year: 50, month: 6, day: 0), to: Date(timeIntervalSince1970: 0))!

// Следующий понедельник с начала месяца

TestHepler.equalDate(date1: date\_01\_07\_2020.next(.monday), yearDate2: 2020, monthDate2: 7, dayDate2: 6)

// Предыдущий понедельник с начала месяца

TestHepler.equalDate(date1: date\_01\_07\_2020.previous(.monday), yearDate2: 2020, monthDate2: 6, dayDate2: 29)

// MARK: **Дата в конце месяца**

**let** date\_31\_07\_2020 = Calendar.current.date(byAdding: DateComponents(year: 50, month: 6, day: 30), to: Date(timeIntervalSince1970: 0))!

// Следующий понедельник с конца месяца

TestHepler.equalDate(date1: date\_31\_07\_2020.next(.monday), yearDate2: 2020, monthDate2: 8, dayDate2: 3)

// Прерыдущий понедельник с конца месяца

TestHepler.equalDate(date1: date\_31\_07\_2020.previous(.monday), yearDate2: 2020, monthDate2: 7, dayDate2: 27)

// MARK: **Дата в начале года**

**let** date\_01\_01\_2020 = Calendar.current.date(byAdding: DateComponents(year: 50, month: 0, day: 0), to: Date(timeIntervalSince1970: 0))!

// Предыдущий понедельник с начала года

TestHepler.equalDate(date1: date\_01\_01\_2020.previous(.monday), yearDate2: 2019, monthDate2: 12, dayDate2: 30)

// Следующий понедельник с начала года

TestHepler.equalDate(date1: date\_01\_01\_2020.next(.monday), yearDate2: 2020, monthDate2: 1, dayDate2: 6)

// MARK: **Дата в конце года**

**let** date\_31\_12\_2020 = Calendar.current.date(byAdding: DateComponents(year: 50, month: 11, day: 30), to: Date(timeIntervalSince1970: 0))!

// Следующий понедельник с конца года

TestHepler.equalDate(date1: date\_31\_12\_2020.next(.monday), yearDate2: 2021, monthDate2: 1, dayDate2: 4)

// Предыдущий понедельник с конца года

TestHepler.equalDate(date1: date\_31\_12\_2020.previous(.monday), yearDate2: 2020, monthDate2: 12, dayDate2: 28)

}

Результаты тестирования приведены в таблице 1.1.

Таблица 1.1 – Тестирование получения даты следующего и предыдущего выбранного дня недели

|  |  |  |
| --- | --- | --- |
| Тесты | Результат | Комментарий |
| Дата последи недели | Положительный | Правильно находит дату следующего и предыдущего понедельника |
| Дата в конце недели | Положительный | Правильно находит дату следующего и предыдущего понедельника |
| Дата в начале недели | Положительный | Правильно находит дату следующего и предыдущего понедельника |
| Дата в конце месяца | Положительный | Правильно находит дату следующего и предыдущего понедельника |
| Дата в начале месяца | Положительный | Правильно находит дату следующего и предыдущего понедельника |
| Дата в конце года | Положительный | Правильно находит дату следующего и предыдущего понедельника |
| Дата в начале года | Положительный | Правильно находит дату следующего и предыдущего понедельника |

Как видно из таблицы 1.1, данный модуль работает корректно на всех граничных данных.

Следующий модуль, который следует протестировать – модуль определения номера дня недели от даты. Этот модуль будет протестирован с помощью способа «покрытия решений», так как можно с легкостью перебрать весь год с помощью цикла и проверить все выдаваемые номера дней недели, учитывая, что номера дней недели циклично меняются от 1 до 7. Код тестирования приведен ниже (листинг 10).

Листинг 10 – Тестирование получение дня недели из даты

**func** testNumberWeekday() {

// первый понедельник в году

**let** date\_06\_01\_2020 = Calendar.current.date(byAdding: DateComponents(year: 50, month: 0, day: 5), to: Date(timeIntervalSince1970: 0))!

**var** currNumberWeekday = 1

// Пробегаем по ближайшим 5 годам, для практически полного покрытия

**for** i **in** 0...1825 {

XCTAssertEqual(DateHelper.getNumberWeekday(byDate: Calendar.current.date(byAdding: .day, value: i, to: date\_06\_01\_2020)!), currNumberWeekday)

**if** currNumberWeekday < 7 {

currNumberWeekday += 1

} **else** {

currNumberWeekday = 1

}

}

}

Исходя из результатов тестов, приведенных на рисунке 4, можно сказать, что минимум ближайшие 5 лет этот модуль будет работать корректно и не принесет дополнительных проблем разработчику.

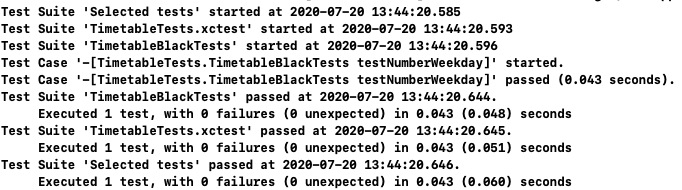


Рисунок 4 – Результаты тестирования получения номера дня недели

Последний модуль, который пройдет тестирование, будет модуль определения четности недели, беря информацию из сайта «Паллада» от СибГУ. Способ тестирования – «покрытие решений». Так же как и в тестировании прошлого модуля, проверим корректность выдаваемых значений со входными данными на следующие 5 лет. Код тестирования приведен ниже (листинг 11).

Листинг 11 – Тестирование определения четности недели

**func** testEvenWeek() {

// Начало четной недели (по сайту Паллада)

**let** date\_20\_07\_2020 = Calendar.current.date(byAdding: DateComponents(year: 50, month: 6, day: 19), to: Date(timeIntervalSince1970: 0))!

**var** currEven = **false**

**for** i **in** 0...150 {

print(Calendar.current.date(byAdding: .day, value: 7\*i, to: date\_20\_07\_2020)!)

XCTAssertEqual(DateHelper.weekIsEven(byDate: Calendar.current.date(byAdding: .day, value: 7\*i, to: date\_20\_07\_2020)!), currEven)

currEven = !currEven

}

}

Посмотрев на отчет о тестировании, часть которого приведена на рисунке 5, заметим, что после наступления 2023 года модуль выдает некорректные значения.



Рисунок 5 – Результат тестирования модуля четности недели

Исходя из результата, нужно будет переработать алгоритм, для получения корректных значений на протяжении всех 5 следующих лет.

ЗАКЛЮЧЕНИЕ

В результате выполнения задания по учебной практике закреплены теоретические знания в области информационных технологий, полученные за время обучения, и развиты практические навыки в тестировании программ.

Решены следующие задачи:

1) Изучена литература по тестированию программ различными стратегиями и методиками

2) В качестве объекта тестирования выбрана курсовая работа по дисциплине «Объектно-ориентированное программирование», реализованная на языке программирования Swift и служащая для донесения актуального расписания занятий СибГУ.

3) Изучена стратегия тестирования «белый ящик». Выбран «метод покрытия решений» для тестирования. Для тестирования были выбраны функции, которые работают с базой данных: производят запись, чтение и конвертацию данных для вида, который используется в приложении.

4) Изучена стратегия тестирования «черный ящик». Для тестирования этой стратегией были выбраны модули, которые работают с датами в приложении. Были выбраны разные методы тестирования: у одного модуля – «анализ граничных значений», у двух других – «полное покрытие решений».

5) Подведен итог тестирования

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Особенности тестирования «серого ящика» [Электронный ресурс]. – URL: <https://quality-lab.ru/blog/key-principles-of-gray-box-testing/> (дата обращения: 18.07.2020)

2. Тестирование белого и черного ящика [Электронный ресурс]. – URL: <http://unetway.com/tutorial/testirovanie-belogo-asika-i-cernogo-asika/> (дата обращения: 18.07.2020)

3. White/Black/Gray Box тестирование [Электронный ресурс]. – URL: <http://unetway.com/tutorial/testirovanie-belogo-asika-i-cernogo-asika/> (дата обращения: 18.07.2020)

4. Метод «белого ящика». Тестирование и отладка программного обеспечения – курсовая работа [Электронный ресурс]. – URL: <http://unetway.com/tutorial/testirovanie-belogo-asika-i-cernogo-asika/> (дата обращения: 19.07.2020)

**Отзыв руководителя от Организации о прохождении практики обучающимся**

1. Полученные компетенции в соответствии с программой практики
2. Умение применять методы и средства познания, обучения и самоконтроля для интеллектуального развития, повышения культурного уровня, профессиональной компетенции, сохранения своего здоровья, нравственного и физического самосовершенствования.
3. Умение критически оценивать свои достоинства и недостатки, наметить пути и выбрать средства развития достоинств и устранения недостатков
4. Способность использовать современные компьютерные технологии поиска информации для решения поставленной задачи, критического анализа этой информации и обоснования принятых идей и подходов к решению.
5. Способность проводить сбор, анализ научно-технической информации, отечественного и зарубежного опыта по тематике исследования.
6. Способность оформлять полученные рабочие результаты в виде презентаций, научно-технических отчетов, статей и докладов на научно-технических конференциях.
7. Характеристика работы обучающегося

В ходе прохождения практики по получению первичных профессиональных умений и навыков, в том числе первичных умений и навыков научно-исследовательской деятельности студент Рылов А. Е. показал умение применить и использовать знания, полученные в ВУЗе, для решения поставленных перед ни практических задач:

- изучены справочно-информационные издания по профилю задания;

- выполнено индивидуальное задание.

В ходе выполнения практического задания удалось закрепить знания по программированию, изучены методы тестирования программ, разработаны тестовые программы для выполнения тестирования.

Дополнительно получены следующие компетенции:

- способность к самоорганизации и самообразованию, а также способность осуществлять поиск, хранение, обработку и анализ информации из различных источников и баз данных, представлять ее в требуемом формате с использованием информационных, компьютерных и сетевых технологий.

Работа практиканта заслуживает оценки «отлично».

1. Замечания руководителя практики о прохождении практики обучающимся

Все оговоренные виды работ выполнены в срок, видимых замечаний не обнаружено. Обучающийся овладел всеми необходимыми теоретическими и практическими навыками. Замечаний к работе нет.

Руководитель практики от Организации

Доцент каф. ИВТ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Моргунова О. Н.

должность подпись расшифровка подписи

**М. П.**