АВТОНОМНАЯ НЕКОММЕРЧЕСКАЯ ОРГАНИЗАЦИЯ
ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ СОЦИАЛЬНЫЙ ИНСТИТУТ»

Утверждаю
Декан факультета
_____Ж.В. Игнатенко
«18» мая 2026 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Разработка мобильных приложений

Направление подготовки: 09.03.02 Информационные системы и технологии

Направленность (профиль) программы: Проектирование информационных систем и их компонентов

Квалификация выпускника: Бакалавр

Форма обучения очная, заочная

Разработана
Канд. эконом. наук, доцент
_____ Р.А. Воронкин

Согласована
зав. кафедрой ИС
_____Ж.В. Игнатенко

Рекомендована
на заседании кафедры ИС
от «18» мая 2026г.
протокол № 9
Зав. кафедрой _____ Ж.В. Игнатенко

Одобрена
на заседании учебно-методической
комиссии факультета
от «18» мая 2026 г.
протокол № 9
Председатель УМК
_____ Ж.В. Игнатенко

Ставрополь, 2026 г.

Содержание

1. Цели освоения дисциплины ... 3
2. Место дисциплины в структуре ОПОП ... 3
3. Планируемые результаты обучения по дисциплине ... 3
4. Объем дисциплины и виды учебной работы ... 4
5. Содержание и структура дисциплины ... 5
5.1. Содержание дисциплины ... 5
5.2. Структура дисциплины ... 5
5.3. Занятия семинарского типа ... 6
5.4. Курсовой проект (курсовая работа, расчетно-графическая работа, реферат, контрольная работа) ... 6
5.5. Самостоятельная работа ... 6
6. Образовательные технологии ... 7
7. Оценочные материалы для текущего контроля успеваемости и промежуточной аттестации ... 8
8. Учебно-методическое и информационное обеспечение дисциплины ... 40
8.1. Основная литература ... 40
8.2. Дополнительная литература ... 40
8.3.Программноеобеспечение ... **Ошибка! Закладка не определена.**
8.4. Профессиональные базы данных ... **Ошибка! Закладка не определена.**
8.5. Информационные справочные системы ... **Ошибка! Закладка не определена.**
8.6. Интернет-ресурсы ... **Ошибка! Закладка не определена.**
8.7. Методические указания по освоению дисциплины ... 40
9. Материально-техническое обеспечение дисциплины ... 45
10. Особенности освоения дисциплины лицами с ограниченными возможностями здоровья ... 46

1. ЦЕЛИ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Целями освоения дисциплины «Программирование мобильных устройств» являются: формирование у обучающихся представлений о принципах разработки программного обеспечения для мобильных платформ, освоение современных средств и технологий создания мобильных приложений, а также формирование профессиональных компетенций, необходимых для проектирования, реализации, тестирования и сопровождения программ для мобильных устройств.

Задачи при изучении дисциплины:

1. Изучение особенностей мобильных устройств и мобильных платформ, их архитектурных, программных и пользовательских характеристик, а также роли мобильных приложений в современной цифровой среде.
2. Освоение базовых и прикладных средств программирования мобильных устройств, включая язык Kotlin, среду разработки Android Studio, средства сборки, отладки и тестирования мобильных приложений.
3. Изучение принципов проектирования и разработки пользовательского интерфейса мобильных приложений с учетом требований эргономики, адаптивности и удобства взаимодействия с пользователем.
4. Овладение практическими навыками создания мобильных приложений, включая разработку экранных форм, обработку пользовательских событий, организацию навигации, работу с данными и использование основных компонентов Android-приложений.
5. Формирование навыков анализа, отладки, тестирования и сопровождения мобильных программных продуктов, а также подготовки приложений к практическому использованию и дальнейшему развитию.

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОПОП

Дисциплина «Разработка мобильных приложений» входит в Блок 1(Б.1.ДВ.2) «Дисциплины (модули)», часть, формируемую участниками образовательных отношений – обязательные дисциплины.

Предшествующие дисциплины (курсы, модули, практики)	Последующие дисциплины (курсы, модули, практики)
Безопасность информационных систем Технологии программирования Интеллектуальные информационные системы и технологии	

3. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ

Код и наименование компетенции	**Код и наименование индикатора (индикаторов) достижения компетенции**	**Результаты обучения**

ПК-8 Владение навыками использования различных технологий разработки программного обеспечения	ПК-8.1. Определяет формальные методы конструирования программного обеспечения	**Знает** методы конструирования программного обеспечения **Умеет** определять формальные методы конструирования программного обеспечения **Владеет навыками** конструирования программного обеспечения
	ПК-8.2. Выполняет работы и управляет работами по формализации и моделированию программного обеспечения	**Знает** методы управления работами по формализации и моделированию программного обеспечения **Умеет** выполнять работы и управляет работами по формализации и моделированию программного обеспечения **Владеет навыками** работы и управляет работами по формализации и моделированию программного обеспечения

4. ОБЪЕМ ДИСЦИПЛИНЫ И ВИДЫ УЧЕБНОЙ РАБОТЫ

Общий объем дисциплины составляет 4 зачетных единиц,144 академических часа.

Вид учебной работы	Всего часов		Триместр	
			9	A
	ОФО	ЗФО	ОФО	ЗФО
Контактная работа (всего)	**40,2**	**16,2**	**40,2**	**16,2**
в том числе:				
1) занятия лекционного типа (ЛК)	**20**	**6**	**20**	**6**
из них				
-лекций	20	6	20	6
2) занятия семинарского типа (ПЗ)	**20**	**10**	**20**	**10**
-семинары (С)				
-практические занятия (ПР)	20	10	20	10
-лабораторные работы (ЛР)				
3) групповые консультации				
4) индивидуальная работа				
5) промежуточная аттестация	**0,2**	**0,2**	**0,2**	**0,2**
Самостоятельная работа (всего) (СР)	**103,8**	**123,8**	**103,8**	**127,8**
в том числе:				
Курсовой проект (работа)				
Расчетно-графические работы				
Контрольная работа				
Реферат				
Самоподготовка (самостоятельное изучение разделов, лекционного материала и материала учебников и учебных пособий, подготовка к лабораторным и практическим занятиям, контролю и т.д.)	100	120	100	120
Вид промежуточной аттестации (зачет)	3,8	3,8	3,8	3,8

Общий объем, час	**144**	**144**	**144**	**144**

5. СОДЕРЖАНИЕ И СТРУКТУРА ДИСЦИПЛИНЫ

5.1. Содержаниедисциплины

№ раздела (темы)	Наименование раздела (темы)	Содержание раздела (темы)
1.	Введение в язык Kotlin и базовые конструкции	Назначение языка Kotlin, его особенности и преимущества в разработке мобильных приложений. Структура программы на Kotlin. Переменные и константы. Базовые типы данных. Арифметические, логические и строковые операции. Ввод и вывод данных. Комментарии и правила оформления исходного кода..
2.	Управляющие конструкции и функции в Kotlin	Условные операторы if и when. Циклы for, while, do while. Операторы управления выполнением программы. Объявление и вызов функций. Параметры функций. Возвращаемые значения. Область видимости переменных. Использование функций для структурирования программного кода..
3.	Коллекции, строки и обработка данных в Kotlin	Работа со строками. Массивы. Списки, множества и словари. Основные операции обработки коллекций. Перебор элементов. Поиск, фильтрация и преобразование данных. Использование стандартных средств Kotlin для обработки текстовой и числовой информации.
4.	Основы объектно-ориентированного программирования в Kotlin	Классы и объекты. Свойства и методы. Конструкторы. Модификаторы доступа. Наследование и переопределение. dataclass. Основы безопасной работы с данными, null-безопасность. Обработка исключений. Применение объектно-ориентированного подхода при разработке программ на Kotlin..
5.	Введение в разработку мобильных приложений под Android	Понятие мобильной платформы Android. Архитектура Android-приложения. Установка и настройка Android Studio. Структура проекта Android. Основные файлы и каталоги проекта. Файл AndroidManifest.xml. Жизненный цикл Activity. Сборка и запуск приложения на эмуляторе и мобильном устройстве..
6.	Разработка пользовательского интерфейса Android-приложений	Основы построения пользовательского интерфейса. Разметка экранов. Контейнеры и элементы управления. Работа с TextView, EditText, Button, ImageView, списками и другими компонентами интерфейса. Обработка событий. Принципы адаптивного и удобного интерфейса мобильного приложения..
7.	Навигация и работа с данными в Android-приложениях	Переходы между экранами приложения. Использование Intent. Передача данных между Activity. Основы работы с Fragment. Получение и обработка пользовательского ввода. Хранение данных с помощью SharedPreferences, файлов и базовых средств локального хранения. Организация логики взаимодействия между экранами и данными приложения.
8.	Тестирование, отладка и итоговая разработка мобильного приложения	Использование средств отладки в Android Studio. Поиск и устранение ошибок. Проверка корректности работы интерфейса и логики приложения. Основы тестирования мобильных приложений. Разработка учебного Android-приложения, объединяющего пользовательский интерфейс, навигацию и работу с данными. Подготовка приложения к демонстрации и сопровождению.

5.2. Структурадисциплины

№ раздела (темы)	Наименование раздела (темы)	Всего*	Количество часов							
			Л		ЛР		К		СР	
			ОФО	ЗФО	ОФО	ЗФО	ОФО	ЗФО	ОФО	ЗФО
1	Введение в язык Kotlin и базовые конструкции	14/13	2	-	-	1	-	-	12	12
2	Управляющие конструкции и функции в Kotlin	16/14	2	1	2	1	-	-	12	12
3	Коллекции, строки и обработка данных в Kotlin	18/16	4	1	2	1	-	-	12	14
4	Основы объектно-ориентированного программирования в Kotlin	22/16	4	1	6	1	-	-	12	14
5	Введение в разработку мобильных приложений под Android	16/21	2	1	4	2	-	-	12	18
6	Разработка пользовательского интерфейса Android-приложений	14/19	2	-	-	1	-	-	12	18
7	Навигация и работа с данными в Android-приложениях	18/21	2	1	4	2	-	-	12	18
8	Тестирование, отладка и итоговая разработка мобильного приложения	20/20	2	1	2	1	-	-	16	18
	Реферат	-	-	-	-	-	-	-	-	-
	Зачет	4/4	-	-			-	-	-	-
	Общий объем	**144/144**	**20**	**6**	**20**	**10**	**-**	**-**	**100**	**124**

5.3. Занятиясеминарскоготипа

№ п/п	№ раздела (темы)	Вид занятия	Наименование	Количество часов	
				ОФО	ЗФО
1	1	ПР	Введение в Kotlin. Базовый синтаксис языка	2	1
2	2	ПР	Условные конструкции и циклы в Kotlin	2	1
3	2	ПР	Функции и основы модульного программирования в Kotlin	2	1
4	3	ПР	Коллекции и работа со строками в Kotlin	2	1
5	4	ПР	Объектно-ориентированное программирование в Kotlin	2	-
6	4	ПР	Обработка исключений и основы функционального стиля в Kotlin	2	1
7	5	ПР	Введение в разработку Android-приложений. Среда Android Studio	2	-
8	6	ПР	Пользовательский интерфейс Android-приложения	2	1
9	7	ПР	Навигация и взаимодействие между экранами Android-приложения	2	-
10	8	ПР	Работа с данными и итоговая мини-разработка Android-приложения	2	-

5.4. Курсовой проект (курсовая работа, расчетно-графическая работа, реферат, контрольная работа)

5.5. Самостоятельнаяработа

№ темы	Виды самостоятельной работы	Количество часов	
		ОФО	ЗФО
1	Изучение источников информации по теме. Подготовка к лабораторной работе	12	12
2	Изучение источников информации по теме. Подготовка к лабораторной работе	12	12
3	Изучение источников информации по теме. Подготовка к лабораторной работе	12	14
4	Изучение источников информации по теме. Подготовка к лабораторной работе	12	14
5	Изучение источников информации по теме. Подготовка к лабораторной работе	12	18
6	Изучение источников информации по теме. Подготовка к лабораторной работе	12	18
7	Изучение источников информации по теме. Подготовка к лабораторной работе	12	18
8	Изучение источников информации по теме. Подготовка к лабораторной работе	16	18
	Реферат	-	-
	Подготовка к аттестации	4	4

6. ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ

Основные технологии обучения:

– работа с правовой информацией, в том числе с использованием современных компьютерных технологий, ресурсов сети Интернет;

– работа с текстами учебника, дополнительной литературой;

– работа с таблицами, схемами;

– выполнение тестовых заданий по темам;

– участие в дискуссиях;

– работа с документами.

Информационные технологии, используемые при осуществлении образовательного процесса по дисциплине:

– сбор, хранение, систематизация, обработка и представление учебной и научной информации;

– обработка различного рода информации с применением современных информационных технологий;

– самостоятельный поиск дополнительного учебного и научного материала, с использованием поисковых систем и сайтов сети Интернет, электронных энциклопедий и баз данных;

– использование электронной почты для рассылки и асинхронного общения, чата преподавателей и обучающихся, переписки и обсуждения возникших учебных проблем для синхронного взаимодействия;

– использование дистанционных образовательных технологий (при необходимости).

При чтении лекций используется компьютерная техника для демонстрации слайдов с помощью программного приложения MicrosoftPowerPoint. На практических занятиях студенты представляют результаты выполнения самостоятельной работы, подготовленные с помощью программного продукта MicrosoftWord. При выполнении практических заданий на практических занятиях, студентами используется программное обеспечение: Windows 7, MicrosoftOffice.

Интерактивные и активные образовательные технологии

№ раздела (темы)	Вид занятия (Л, ПЗ, С, ЛР)	Используемые интерактивные образовательные технологии	Количество часов	
			ОФО	ЗФО

2	Л	Лекция-дискуссия	2	1
3	ЛР	Работа малыми группами	2	1
4	Л	Проблемная лекция.	2	1
5	ЛР	Работа малыми группами	2	1

7. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ И ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

Описание показателей оценивания компетенций, формируемых в процессе освоения дисциплины (модуля), и используемые оценочные средства приведены в таблице 1.

Таблица 1 – Показатели оценивания и оценочныесредствадля оценивания результатов обучения по дисциплине

Код и наименование формируемой компетенции	Код и наименование индикатора достижения формируемой компетенции	Показатели оценивания (результаты обучения)	Процедуры оценивания (оценочные средства)	
			текущий контроль успеваемости	промежуточная аттестация
ПК-8 Владение навыками использования различных технологий разработки программного обеспечения т	ПК-8.1. Определяет формальные методы конструирования программного обеспечения	**Знает** методы конструирования программного обеспечения	Контрольные вопросы Тестовоезадание	зачет (контрольные вопросы, тестовое задание)
		Умеет определять формальные методы конструирования программного обеспечения	Практическоезадание	зачет(ситуационнаязадача)
		Владеет навыками конструирования программного обеспечения	Практическоезадание	зачет(ситуационнаязадача)
	ПК-8.2. Выполняет работы и управляет работами по формализации и моделированию программного обеспечения	**Знает** методы управления работами по формализации и моделированию программного обеспечения	Контрольные вопросы Тестовоезадание	зачет (контрольные вопросы, тестовое задание)

Код и наименование формируемой компетенции	Код и наименование индикатора достижения формируемой компетенции	Показатели оценивания (результаты обучения)	Процедуры оценивания (оценочные средства)	
			текущий контроль успеваемости	промежуточная аттестация
		Умеет выполнять работы и управляет работами по формализации и моделированию программного обеспечения	Практическоезадание	зачет(ситуационнаязадача)
		Владеет навыками работы и управляет работами по формализации и моделированию программного обеспечения	Практическоезадание	зачет(ситуационнаязадача)
ПК-8				зачет

7.1. ОЦЕНОЧНЫЕ СРЕДСТВА, КРИТЕРИИ И ШКАЛА ОЦЕНКИ

Типовые задания для текущего контроля успеваемости

Перечень типовых контрольных вопросов для подготовки к устному опросу

Устные опросы проводятся во время лекций, практических занятий и возможны при проведении промежуточной аттестации в качестве дополнительного испытания при недостаточности результатов тестирования. Основные вопросы для устного опроса доводятся до сведения обучающихся на предыдущем занятии.

Развернутый ответ обучающегося должен представлять собой связное, логически последовательное сообщение на заданную тему, показывать его умение применять определения, правила в конкретных случаях.

1. Что представляет собой язык программирования Kotlin и в чем заключаются его основные особенности?
2. Почему Kotlin считается удобным языком для разработки мобильных приложений?
3. Каково назначение функции main() в программе на Kotlin?
4. Чем отличаются функции print() и println()?
5. В чем различие между var и val?
6. Что такое тип данных в Kotlin?
7. Какие базовые типы данных используются в Kotlin?
8. Чем отличаются типы Int, Long, Float и Double?
9. Для чего используются типы Boolean, Char и String?
10. Что такое вывод типа в Kotlin?
11. В каких случаях тип переменной указывается явно?
12. Какие арифметические операции поддерживаются в Kotlin?
13. Почему деление двух целых чисел может давать неожиданный результат?
14. Для чего используется функция readln()?

15. Почему при вводе чисел с клавиатуры требуется преобразование типов?
16. Как преобразовать строку в Int и Double?
17. Что такое строковые шаблоны и как они используются в Kotlin?
18. Для чего в программе применяются комментарии?
19. Почему важно соблюдать форматирование кода?
20. Какие условные конструкции используются в Kotlin?
21. В чем различие между if и when?
22. Какие циклы поддерживаются в Kotlin?
23. Для чего используются функции в Kotlin?
24. Что такое класс и объект в Kotlin?
25. Что понимается под null-безопасностью и обработкой исключений в Kotlin?
26. Что представляет собой платформа Android?
27. Почему Android является одной из основных платформ разработки мобильных приложений?
28. Какую роль выполняет Android Studio при создании мобильных приложений?
29. Из каких основных частей состоит проект Android-приложения?
30. Для чего предназначен файл AndroidManifest.xml?
31. Что такое Activity в Android-приложении?
32. Что понимается под жизненным циклом Activity?
33. Какие основные методы жизненного цикла Activity используются в Android?
34. Что такое пользовательский интерфейс Android-приложения?
35. Какие основные элементы интерфейса применяются в Android, например TextView, EditText и Button?
36. Для чего используются контейнеры и разметка интерфейса в Android?
37. Что такое XML-разметка в Android-приложении?
38. Как связаны XML-разметка и Kotlin-код в Android-проекте?
39. Что такое обработка событий в Android-приложении?
40. Как организуется реакция программы на нажатие кнопки?
41. Что такое Intent и для чего он используется?
42. Чем отличаются явные и неявные Intent?
43. Как выполняется переход между экранами Android-приложения?
44. Как можно передавать данные между Activity?
45. Что такое Fragment и для чего он используется?
46. Какие способы хранения данных применяются в Android-приложениях?
47. Для чего используется SharedPreferences?
48. Какова роль файлового хранения и локальных баз данных в Android-приложении?
49. Какие средства отладки предоставляет Android Studio?
50. Почему тестирование и отладка являются обязательной частью разработки Android-приложений?

Критерии и шкала оценивания устного опроса

отлично	1) студент полно излагает материал, дает правильное определение основных понятий; 2) обнаруживает понимание материала, может обосновать свои суждения, применить знания на практике, привести необходимые примеры не только из учебника, но и самостоятельно составленные; 3) излагает материал последовательно и правильно с точки зрения норм литературного языка.
хорошо	студент дает ответ, удовлетворяющий тем же требованиям, что и для отметки, но допускает 1–2 ошибки, которые сам же исправляет, и 1–2 недочета в последовательности и языковом оформлении излагаемого.
удовлетворительно	студент обнаруживает знание и понимание основных положений данной темы, но:

	1) излагает материал неполно и допускает неточности в определении понятий или формулировке правил; 2) не умеет достаточно глубоко и доказательно обосновать свои суждения и привести свои примеры; 3) излагает материал непоследовательно и допускает ошибки в языковом оформлении излагаемого.
неудовлетворительно	студент обнаруживает незнание большей части соответствующего вопроса, допускает ошибки в формулировке определений и правил, искажающие их смысл, беспорядочно и неуверенно излагает материал. Оценка «неудовлетворительно» отмечает такие недостатки в подготовке, которые являются серьезным препятствием к успешному овладению последующим материалом.

Типовые практические задания

Введение в язык Kotlin и базовые конструкции

1. Разработать консольную программу на языке Kotlin, которая выводит в структурированном виде сведения о студенте: фамилию, имя, учебную группу, направление подготовки и наименование дисциплины. В программе необходимо использовать несколько отдельных инструкций вывода, а текст оформить так, чтобы результат выглядел как небольшая информационная карточка.

2. Создать программу, в которой объявляются переменные для хранения имени пользователя, его возраста, среднего балла и признака наличия академической стипендии. Необходимо вывести значения всех переменных на экран и сопроводить каждое значение понятной подписью. Задание направлено на закрепление работы с базовыми типами данных `String`, `Int`, `Double` и `Boolean`.

3. Написать программу, демонстрирующую различие между изменяемыми и неизменяемыми значениями в Kotlin. В программе необходимо объявить несколько переменных через `var` и несколько констант через `val`, затем изменить только допустимые значения и вывести результат до и после изменения. Следует показать в отчете, какие данные логично хранить как константы, а какие как переменные.

4. Разработать программу, которая запрашивает у пользователя имя и год рождения, после чего вычисляет примерный возраст пользователя и выводит персонализированное сообщение. При выполнении задания необходимо использовать ввод данных с клавиатуры, преобразование строкового значения в число и строковые шаблоны Kotlin.

5. Создать программу для вычисления суммы, разности, произведения, частного и остатка от деления двух целых чисел, вводимых пользователем. Результаты всех операций должны быть выведены в удобочитаемой форме, чтобы по выводу программы было понятно, какое именно действие было выполнено.

6. Написать программу, которая получает от пользователя длину и ширину прямоугольника, затем вычисляет его площадь и периметр. Для хранения исходных данных и результатов необходимо использовать отдельные переменные, а итог оформить в виде краткого текстового отчета.

7. Разработать программу, которая запрашивает стоимость единицы товара и количество единиц, после чего вычисляет общую стоимость покупки. Дополнительно необходимо вывести поясняющий текст с указанием исходных данных и полученного результата, используя строковые шаблоны для встраивания значений в текст.

8. Создать программу, которая переводит введенное пользователем количество минут в секунды. В решении необходимо показать использование целочисленных переменных, арифметической операции умножения и форматированного текстового вывода результата.

9. Написать программу, которая запрашивает температуру в градусах Цельсия и вычисляет соответствующее значение в градусах Фаренгейта. В решении следует использовать вещественные типы данных, арифметические выражения и осмысленные имена переменных, отражающие физический смысл величин.

10. Разработать программу для вычисления среднего арифметического трех чисел, вводимых пользователем. Необходимо показать, как выполняется чтение значений с клавиатуры, их преобразование к числовому типу и вычисление итогового результата с использованием отдельной переменной для среднего значения.

11. Создать программу, которая запрашивает размер файла в байтах и выводит его размер в килобайтах и мегабайтах. Требуется использовать вещественные вычисления и кратко пояснить в комментариях, почему в данном случае удобнее применять тип `Double`.

12. Написать программу, которая принимает от пользователя длину ребра куба и вычисляет его объем и площадь полной поверхности. В решении необходимо продемонстрировать использование базовых арифметических операций и вывод нескольких связанных результатов в одном логически завершенном блоке.

13. Разработать программу, которая запрашивает количество часов, минут и секунд, а затем переводит введенное время в общее количество секунд. Важно правильно организовать структуру вычислений, выделив отдельные переменные для каждого этапа расчета.

14. Создать программу, которая вычисляет стоимость поездки на автомобиле. Пользователь должен ввести расстояние в километрах, расход топлива на 100 км и стоимость одного литра топлива. Программа должна определить, сколько топлива потребуется и какова будет итоговая стоимость поездки.

15. Написать программу, которая запрашивает имя студента, название дисциплины и количество набранных баллов, а затем формирует короткий отчет о результатах текущего контроля. В задании необходимо сделать акцент на корректной работе со строками и на использовании строковых шаблонов для построения итогового текста.

16. Разработать программу, в которой демонстрируется различие между целочисленным и вещественным делением. Пользователь вводит два числа, программа сначала выполняет целочисленное деление, а затем вещественное, если одно из значений явно преобразовано к типу `Double`. В выводе следует показать оба результата и пояснить различие.

17. Создать программу, которая запрашивает цену товара и процент скидки, после чего вычисляет величину скидки и итоговую цену после ее применения. В программе необходимо использовать вещественные типы данных и последовательность вычислений через промежуточные переменные.

18. Написать программу, которая получает от пользователя количество дней и переводит это значение в количество полных недель и оставшихся дней. В решении необходимо использовать целочисленное деление и операцию нахождения остатка от деления, а результат представить в понятной текстовой форме.

19. Разработать программу, которая запрашивает радиус окружности и вычисляет длину окружности и площадь круга. Значение числа π необходимо задать в виде константы. Следует обратить внимание на корректный выбор типа данных и читаемое оформление вычислительных выражений.

20. Создать программу, которая запрашивает у пользователя количество товаров, цену одного товара и процент налога, после чего вычисляет стоимость без налога, сумму налога и полную стоимость покупки. В программе требуется использовать несколько переменных для промежуточных вычислений и вывести итог в развернутом виде.

21. Написать программу, которая формирует текстовую визитку разработчика. Необходимо использовать заранее заданные переменные для имени, специализации, предпочитаемого языка программирования и учебного заведения, а затем вывести эти сведения в аккуратно оформленном виде с переносами строк.

22. Разработать программу, которая запрашивает два вещественных числа и вычисляет их сумму, разность, произведение, частное и среднее арифметическое. Особое внимание следует уделить выбору подходящего типа данных и понятному представлению результатов вычислений.

23. Создать программу, которая запрашивает размер заработной платы за месяц и количество месяцев, после чего вычисляет общий доход за указанный период. Дополнительно программа должна вывести поясняющее сообщение, содержащее исходные данные и итоговую сумму дохода.

24. Написать программу, которая получает от пользователя массу тела и рост, а затем вычисляет индекс массы тела по стандартной формуле. В решении необходимо использовать вещественные числа, математическое выражение с возведением роста в квадрат и подробный вывод результата.

25. Разработать программу, которая запрашивает название товара, его цену в иностранной валюте, курс обмена и количество единиц товара, после чего вычисляет полную стоимость покупки в национальной валюте. Итог должен быть выведен в виде небольшого текстового отчета, содержащего все ключевые параметры расчета.

Управляющие конструкции и функции в Kotlin

1. Разработать программу на языке Kotlin, которая запрашивает у пользователя целое число и определяет, является ли оно положительным, отрицательным или равным нулю. В решении необходимо использовать условный оператор `if`, а результат вывести в виде развернутого поясняющего сообщения.

2. Создать программу, которая получает от пользователя возраст и определяет, относится ли введенное значение к категории ребенка, подростка, взрослого или пожилого человека. Для решения необходимо использовать последовательность условных конструкций и продумать такие границы диапазонов, чтобы классификация выглядела логично и последовательно.

3. Написать программу, которая запрашивает два числа и определяет, какое из них больше, меньше либо равны ли они между собой. При выводе необходимо не просто указать результат сравнения, но и отобразить сами введенные значения в итоговом сообщении.

4. Разработать программу, которая проверяет, является ли введенное пользователем число четным или нечетным. Для решения требуется применить операцию нахождения остатка от деления и условный оператор, а вывод оформить в понятной текстовой форме.

5. Создать программу, которая запрашивает оценку студента в баллах и на ее основе определяет текстовую характеристику результата, например неудовлетворительно, удовлетворительно, хорошо или отлично. В задании необходимо использовать условные конструкции и продемонстрировать грамотную работу с числовыми диапазонами.

6. Написать программу, которая получает номер месяца и выводит его название. Для реализации следует использовать конструкцию `when`, поскольку она позволяет удобно организовать выбор одного из нескольких вариантов по значению переменной.

7. Разработать программу, которая по введенному номеру дня недели выводит его наименование и сообщает, относится ли он к рабочим дням или к выходным. В решении рекомендуется использовать `when` и строковые шаблоны для формирования завершенного текстового сообщения.

8. Создать программу, которая запрашивает у пользователя три числа и определяет наибольшее из них. При выполнении задания необходимо использовать условные операторы и продумать обработку случая, когда некоторые значения совпадают.

9. Написать программу, которая по длинам трех сторон определяет, может ли существовать треугольник с такими сторонами. Для решения нужно использовать проверку условия существования треугольника через сравнение сумм сторон и вывести аргументированный результат.

10. Разработать программу, которая запрашивает логин и пароль, после чего сравнивает их с заранее заданными значениями и выводит сообщение об успешной или неуспешной

аутентификации. В задании необходимо закрепить навыки работы со строками и условными операторами.

11. Создать программу, которая выводит на экран числа от 1 до 10 в возрастающем порядке. Для решения необходимо использовать цикл `for` и обеспечить аккуратный вывод результатов на отдельных строках либо в одной строке через пробел.

12. Написать программу, которая выводит все четные числа от 2 до 20. В решении следует использовать цикл, а также продемонстрировать один из возможных способов отбора нужных значений: либо через шаг цикла, либо через проверку условия.

13. Разработать программу, которая запрашивает натуральное число `n` и вычисляет сумму всех чисел от 1 до `n`. В задании важно показать применение цикла и переменной-накопителя, в которой будет постепенно формироваться итоговый результат.

14. Создать программу, которая получает от пользователя число и вычисляет его факториал. Для реализации требуется использовать цикл и последовательное накопление произведения. Следует также предусмотреть корректную работу для небольших граничных значений, например для единицы.

15. Написать программу, которая выводит таблицу квадратов чисел от 1 до 10. В каждой строке необходимо отображать само число и его квадрат, оформляя вывод в виде простой текстовой таблицы.

16. Разработать программу, которая запрашивает количество чисел, затем в цикле считывает указанное количество значений и вычисляет их сумму. В решении необходимо использовать цикл `for` или `while`, а также организовать повторяющийся ввод данных.

17. Создать программу, которая определяет количество цифр во введенном пользователем целом положительном числе. Для решения требуется использовать цикл `while`, в котором число последовательно уменьшается за счет целочисленного деления.

18. Написать программу, которая получает от пользователя число и выводит его цифры по одной на отдельных строках. В решении необходимо использовать цикл и операции деления и нахождения остатка от деления.

19. Разработать программу, которая выводит все числа от 1 до 100, кратные 3. Для выполнения задания следует использовать цикл и проверку условия кратности, а также организовать читаемый вывод результатов.

20. Создать программу, которая по введенному числу определяет, является ли оно простым. В решении требуется организовать цикл проверки возможных делителей и использовать логическую переменную либо досрочный выход из цикла для фиксации результата.

21. Написать программу, которая демонстрирует использование операторов `break` и `continue`. Например, можно организовать цикл вывода чисел от 1 до 20, в котором часть чисел пропускается, а при достижении определенного значения выполнение цикла прекращается. Результат должен наглядно показывать действие обоих операторов.

22. Разработать программу, содержащую функцию для вычисления площади прямоугольника по двум параметрам: длине и ширине. В основной части программы необходимо запросить данные у пользователя, вызвать функцию и вывести результат. Задание направлено на освоение объявления функций, передачи параметров и возврата значения.

23. Создать программу с функцией, которая принимает число и возвращает его квадрат. Основная программа должна запросить исходное значение, вызвать функцию и вывести результат вычисления. Необходимо показать связь между параметром функции и возвращаемым значением.

24. Написать программу, содержащую функцию для перевода температуры из градусов Цельсия в градусы Фаренгейта. В основной части требуется организовать ввод значения температуры, вызов функции и вывод вычисленного результата с пояснением единиц измерения.

25. Разработать программу, в которой реализованы две отдельные функции: одна вычисляет сумму двух чисел, а вторая их произведение. Основная программа должна запросить значения, последовательно вызвать обе функции и вывести результаты. В ходе выполнения

задания необходимо продемонстрировать модульную организацию кода и удобство повторного использования функций.

Коллекции, строки и обработка данных в Kotlin

1. Разработать программу на языке Kotlin, которая запрашивает у пользователя строку текста и выводит количество символов в этой строке. В решении необходимо использовать свойства строкового типа и оформить результат так, чтобы в выводе было видно как исходное значение, так и вычисленная длина строки.

2. Создать программу, которая получает от пользователя имя и фамилию, после чего формирует и выводит полное имя в одном текстовом выражении. Дополнительно необходимо определить и вывести длину полученной строки, чтобы продемонстрировать работу как со строковыми шаблонами, так и со свойством длины строки.

3. Написать программу, которая запрашивает строку и выводит ее в верхнем и нижнем регистрах. В решении следует использовать стандартные строковые методы Kotlin и сопроводить каждый результат понятной подписью.

4. Разработать программу, которая получает от пользователя строку и определяет, содержит ли она указанное слово или символ. Для выполнения задания необходимо организовать отдельный ввод искомого фрагмента и вывести развернутый результат проверки.

5. Создать программу, которая запрашивает строку и выводит ее первый и последний символ. В решении следует предусмотреть корректную работу со строкой ненулевой длины и пояснить в выводе, какие именно позиции были использованы.

6. Написать программу, которая получает от пользователя строку и вычисляет количество пробелов в ней. Для решения требуется организовать перебор символов строки и использовать счетчик, накапливающий количество найденных пробельных символов.

7. Разработать программу, которая запрашивает строку и определяет количество гласных букв в ней. Допускается учитывать как строчные, так и заглавные буквы. В решении необходимо использовать цикл, проверку принадлежности символов заранее заданному набору и переменную-счетчик.

8. Создать программу, которая получает строку текста и заменяет в ней все пробелы символом подчеркивания. Результат необходимо вывести вместе с исходной строкой, чтобы можно было наглядно сравнить оба варианта.

9. Написать программу, которая запрашивает у пользователя строку и определяет, начинается ли она с заданного символа или фрагмента текста. Следует предусмотреть отдельный ввод проверяемого префикса и вывести логически завершенное сообщение о результате проверки.

10. Разработать программу, которая получает строку и определяет, заканчивается ли она заданным фрагментом текста. В решении необходимо использовать стандартные средства работы со строками и обеспечить понятный вывод результата сравнения.

11. Создать программу, которая запрашивает строку и подсчитывает количество слов в ней, ориентируясь на разделение по пробелам. Следует удалить лишние пробелы по краям строки и предусмотреть обработку ситуации, когда между словами встречается несколько пробелов подряд.

12. Написать программу, которая получает строку из нескольких слов и выводит каждое слово на отдельной строке. Для решения требуется использовать операцию разбиения строки на части и перебор полученной коллекции элементов.

13. Разработать программу, которая формирует массив из пяти целых чисел, вводимых пользователем, а затем выводит все элементы массива в исходном порядке. В решении

необходимо показать создание массива, запись значений по индексам и последующий перебор элементов.

14. Создать программу, которая запрашивает у пользователя элементы массива целых чисел, после чего вычисляет сумму всех его элементов. Для решения необходимо использовать цикл и переменную-накопитель, а результат вывести в развернутой текстовой форме.

15. Написать программу, которая получает массив чисел и определяет максимальный и минимальный элементы. В решении можно использовать либо стандартные функции Kotlin, либо самостоятельный перебор массива с поэтапным сравнением значений.

16. Разработать программу, которая запрашивает массив из нескольких целых чисел и вычисляет среднее арифметическое его элементов. Требуется продемонстрировать сочетание работы с массивом, суммирования значений и вычисления итогового среднего.

17. Создать программу, которая формирует список строк, содержащий названия учебных дисциплин, и выводит их в виде нумерованного перечня. В задании необходимо использовать тип `List` или `MutableList`, а также перебор элементов с одновременным отображением их порядкового номера.

18. Написать программу, которая запрашивает у пользователя несколько чисел, сохраняет их в изменяемый список `MutableList`, а затем выводит только четные элементы этого списка. Для решения можно использовать либо цикл с условием, либо стандартные средства фильтрации.

19. Разработать программу, которая формирует список чисел и создает на его основе новый список, содержащий квадраты всех исходных элементов. В решении необходимо использовать средства преобразования коллекций и затем вывести оба списка для сравнения.

20. Создать программу, которая получает список строк и формирует новый список, содержащий только те элементы, длина которых превышает заданное значение. Пользователь должен ввести минимальную длину, а программа должна выполнить фильтрацию и вывести результат.

21. Написать программу, которая демонстрирует использование множества `Set`. Необходимо запросить у пользователя несколько строковых значений, добавить их в множество и вывести итоговую коллекцию без повторяющихся элементов. В выводе следует обратить внимание на то, что одинаковые значения сохраняются в единственном экземпляре.

22. Разработать программу, которая формирует словарь `Map`, где ключами являются названия товаров, а значениями их цены. После этого программа должна запросить название товара и вывести его стоимость, если такой ключ присутствует в словаре. В решении необходимо показать базовую работу с ассоциативными коллекциями.

23. Создать программу, которая получает список целых чисел и определяет, сколько среди них положительных, отрицательных и равных нулю. Для решения требуется организовать перебор элементов коллекции и использовать несколько счетчиков для разных категорий значений.

24. Написать программу, которая запрашивает строку из чисел, разделенных пробелами, преобразует ее в список целых значений и вычисляет сумму этих чисел. В задании необходимо совместить разбиение строки, преобразование элементов и обработку коллекции числовых данных.

25. Разработать программу, которая формирует список оценок студента, а затем выводит сами оценки, их количество, сумму и средний балл. В решении следует использовать коллекцию чисел и показать, как на ее основе можно получить несколько сводных характеристик набора данных.

Основы объектно-ориентированного программирования в Kotlin

1. Разработать программу на языке Kotlin, в которой создается класс `Student` со свойствами, отражающими имя студента, номер группы и средний балл. В основной части программы необходимо создать объект этого класса, присвоить ему конкретные значения и вывести сведения о студенте в виде связного текстового сообщения.

2. Создать программу, содержащую класс `Book` со свойствами для хранения названия книги, имени автора и количества страниц. После создания объекта данного класса необходимо вывести на экран краткое описание книги, используя значения его свойств и строковые шаблоны Kotlin.

3. Написать программу, в которой реализован класс `Rectangle` со свойствами `width` и `height`, а также методом для вычисления площади прямоугольника. В основной программе требуется создать объект класса, вызвать метод и вывести результат вычисления вместе с исходными параметрами фигуры.

4. Разработать программу, содержащую класс `Circle` со свойством радиуса и методом, вычисляющим длину окружности. Значение числа π следует задать в виде константы внутри класса или в основной программе. После создания объекта необходимо продемонстрировать вызов метода и вывод результата.

5. Создать программу, в которой класс `Employee` содержит свойства для хранения фамилии сотрудника, должности и размера заработной платы. Необходимо реализовать метод, формирующий краткую информационную справку о сотруднике, а затем создать объект этого класса и вывести соответствующее сообщение.

6. Написать программу, в которой реализован класс `Product` со свойствами названия товара, цены и количества. В классе необходимо определить метод, вычисляющий полную стоимость партии товара. В основной программе следует создать объект, вызвать метод и вывести итог в понятной форме.

7. Разработать программу, содержащую класс `Car` со свойствами марки автомобиля, года выпуска и пробега. В основной части требуется создать несколько объектов данного класса и вывести сведения о каждом из них, чтобы показать различие между классом как шаблоном и объектами как конкретными экземплярами.

8. Создать программу с классом `Temperature`, в котором хранится значение температуры в градусах Цельсия. Необходимо реализовать метод для перевода температуры в градусы Фаренгейта и продемонстрировать его использование после создания объекта класса.

9. Написать программу, в которой используется первичный конструктор класса `User` для инициализации имени пользователя и возраста. В основной программе нужно создать объект с помощью конструктора и вывести значения его свойств, показав, как передаются параметры при создании экземпляра класса.

10. Разработать программу, содержащую класс `Point` с координатами `x` и `y`, а также метод для вычисления расстояния от начала координат до данной точки. После создания объекта класса необходимо вызвать метод и вывести результат с пояснением.

11. Создать программу, в которой класс `BankAccount` содержит свойства номера счета и текущего баланса, а также методы пополнения и снятия средств. В основной части программы требуется создать объект счета, выполнить несколько операций изменения баланса и вывести состояние счета после каждой операции.

12. Написать программу, демонстрирующую использование модификаторов доступа. Например, можно создать класс `Person`, в котором часть свойств будет доступна извне, а часть скрыта. В задании необходимо показать, какие элементы класса могут использоваться в основной программе, а какие предназначены только для внутренней логики самого класса.

13. Разработать программу, в которой реализован класс `Triangle` со свойствами длин сторон и методом, вычисляющим периметр треугольника. Дополнительно можно добавить метод

проверки возможности существования треугольника с такими сторонами. После этого необходимо создать объект и вывести результат работы методов.

14. Создать программу, содержащую `dataclass` с именем `StudentRecord`, включающую свойства имени студента, группы и количества баллов. В основной программе необходимо создать несколько объектов этого класса и вывести их содержимое, чтобы продемонстрировать удобство использования `dataclass` для хранения структурированных данных.

15. Написать программу, в которой используется сравнение объектов типа `dataclass`. Например, можно создать два объекта с одинаковыми значениями свойств и один объект с отличающимися значениями, после чего проверить результаты сравнения и вывести их на экран.

16. Разработать программу, демонстрирующую наследование. Необходимо создать базовый класс `Transport`, содержащий общее свойство, например название модели, и производный класс `Bus`, дополняющий его количеством пассажирских мест. В основной части программы следует создать объект производного класса и вывести все его свойства.

17. Создать программу, в которой реализован базовый класс `Animal` с методом вывода общей характеристики, а затем производные классы `Dog` и `Cat`, переопределяющие этот метод. После этого нужно создать объекты производных классов и показать различие в результатах работы переопределенного метода.

18. Написать программу, содержащую класс `Shape` и производные классы `Square` и `Rectangle`, каждый из которых имеет метод вычисления площади. В основной программе требуется создать объекты обеих фигур и вывести результаты вычислений, показав применение наследования и переопределения поведения.

19. Разработать программу, в которой класс `Course` содержит свойства названия дисциплины и количества часов, а также метод вывода краткой информации. Необходимо создать список объектов этого класса и организовать вывод сведений обо всех курсах, чтобы показать сочетание объектного подхода и коллекций.

20. Создать программу, демонстрирующую работу с nullable-типами. Например, можно объявить переменную типа `String?`, присвоить ей значение `null`, а затем безопасно вывести длину строки с использованием оператора безопасного вызова. В выводе следует показать, чем nullable-тип отличается от обычного строкового типа.

21. Написать программу, в которой пользователю предлагается ввести строковое значение, которое затем преобразуется в число с помощью безопасных средств Kotlin, например `toIntOrNull()`. Если преобразование прошло успешно, программа должна вывести число, иначе сообщить о невозможности преобразования. Задание направлено на закрепление приемов безопасной работы с потенциально некорректными данными.

22. Разработать программу, демонстрирующую использование оператора Элвиса `?:`. Например, можно объявить nullable-переменную с именем пользователя и в случае отсутствия значения подставлять строку по умолчанию. Итоговое сообщение должно явно показывать, какое значение было использовано в программе.

23. Создать программу, в которой выполняется обработка исключения при вводе числового значения. Пользователь должен ввести число, программа пытается преобразовать введенную строку к нужному типу, а при возникновении ошибки выводит понятное сообщение. Для реализации необходимо использовать конструкцию `try-catch`.

24. Написать программу, которая содержит класс `Divider` с методом деления двух чисел. В методе необходимо предусмотреть обработку ситуации деления на ноль и вывести корректное сообщение об ошибке. После этого нужно продемонстрировать работу метода на корректных и некорректных входных данных.

25. Разработать программу, объединяющую несколько изученных элементов объектно-ориентированного подхода. Например, можно создать класс `LibraryBook` с использованием первичного конструктора, метода вывода информации, `dataclass` для хранения сведений о

читателе и безопасной обработки nullable-значения даты возврата книги. В основной программе требуется создать объекты, выполнить несколько операций с их свойствами и вывести результат в виде связного отчета.

Тема 5. Введение в разработку мобильных приложений под Android

1. Разработать в Android Studio первое учебное приложение, которое после запуска отображает на экране текст с названием дисциплины, фамилией студента и номером учебной группы. В ходе выполнения задания необходимо создать новый проект, изучить структуру основных каталогов и файлов, а также обеспечить корректный запуск приложения на эмуляторе или мобильном устройстве.

2. Создать Android-приложение, в котором на главном экране размещены текстовый заголовок и краткое описание назначения программы. Необходимо изменить содержимое стартового экрана, отредактировав XML-разметку и связанный с ней Kotlin-код, а затем продемонстрировать результат работы приложения после запуска.

3. Разработать приложение, в котором необходимо изменить стандартное имя проекта, название приложения и текст, отображаемый на экране. Задание должно быть выполнено с анализом того, в каких файлах Android-проекта задаются данные параметры и каким образом они влияют на внешний вид и идентификацию приложения.

4. Создать учебный Android-проект и проанализировать назначение основных элементов его структуры: каталога `java` или `kotlin`, каталога `res`, файла `AndroidManifest.xml`, каталога с разметками и ресурсов строк. После этого необходимо подготовить небольшое приложение, в котором результаты этого анализа будут подтверждены внесением изменений в соответствующие файлы проекта.

5. Разработать Android-приложение, в котором главный экран содержит текстовый блок с краткой информацией о платформе Android. В процессе выполнения задания необходимо создать новый проект, отредактировать файл разметки, изменить строковые ресурсы и выполнить запуск приложения для проверки отображения результата.

6. Создать приложение, демонстрирующее использование строковых ресурсов Android. Необходимо вынести все текстовые надписи из разметки в файл `strings.xml`, а затем настроить интерфейс так, чтобы все элементы экрана использовали ресурсы, а не жестко заданные строки.

7. Разработать Android-приложение, в котором главный экран содержит информацию о студенте, а все текстовые данные берутся из файла ресурсов. В ходе выполнения задания необходимо использовать не менее трех строковых ресурсов и показать, как они подключаются в XML-разметке.

8. Создать приложение, в котором на стартовом экране выводится список основных этапов разработки мобильного приложения в виде нескольких строк текста. Задание предполагает редактирование XML-разметки, работу с текстовыми ресурсами и проверку итогового внешнего вида интерфейса в запущенном приложении.

9. Разработать Android-приложение, содержащее два отдельных экрана `Activity`, каждый из которых отображает различную текстовую информацию. На первом этапе необходимо создать и зарегистрировать вторую `Activity`, а затем настроить корректный запуск приложения с первой активностью.

10. Создать приложение, в котором необходимо изучить и описать назначение файла `AndroidManifest.xml`, после чего внести в проект изменения, связанные с объявлением второй `Activity`. В качестве практического результата должна быть получена программа, в которой обе активности корректно описаны в манифесте.

11. Разработать Android-приложение, в котором после запуска отображается экран приветствия, а по действию пользователя выполняется переход на второй экран с информацией о

разработчике приложения. Для выполнения задания необходимо создать две `Activity` и реализовать базовый переход между ними.

12. Создать приложение, позволяющее изучить жизненный цикл `Activity`. Для этого необходимо добавить в методы жизненного цикла `onCreate`, `onStart`, `onResume`, `onPause`, `onStop` и `onDestroy` вывод диагностических сообщений в лог или на экран, а затем проследить порядок вызова этих методов при разных действиях пользователя.

13. Разработать Android-приложение, в котором необходимо зафиксировать поведение `Activity` при сворачивании и повторном открытии программы. В ходе выполнения задания следует использовать методы жизненного цикла и средства логирования Android Studio для анализа переходов между состояниями активности.

14. Создать приложение, в котором стартовая `Activity` содержит краткое описание проекта, а в коде реализован вывод сообщений о прохождении этапов жизненного цикла в системный журнал. После запуска и тестирования приложения необходимо определить, какие методы вызываются при первом открытии, при сворачивании и при закрытии экрана.

15. Разработать приложение, в котором главный экран оформлен с использованием нескольких текстовых элементов и отступов, а все настройки интерфейса задаются в XML-разметке. Задание направлено на закрепление навыков базовой работы с файлом `activity_main.xml` и на понимание того, как XML-интерфейс связан с экраном приложения.

16. Создать Android-приложение, в котором используется пользовательский текст, заданный как константа в Kotlin-коде `Activity`, а затем отображаемый в элементе интерфейса после запуска. Для выполнения задания необходимо получить доступ к элементу `TextView` из кода и изменить его содержимое программным способом.

17. Разработать приложение, содержащее экран с названием курса, номером практической работы и кратким описанием задания. Часть информации должна быть задана в XML, а часть установлена из Kotlin-кода при создании `Activity`. Это позволит продемонстрировать различие между статическим и программно изменяемым содержимым интерфейса.

18. Создать приложение, в котором нужно проанализировать процесс сборки проекта в Android Studio и выполнить успешную сборку без запуска на устройстве. После этого необходимо запустить приложение на эмуляторе и убедиться, что собранная версия корректно отображает предусмотренный интерфейс.

19. Разработать Android-приложение, в котором все элементы стартового интерфейса создаются в соответствии с единым стилем: единый размер текста, одинаковые отступы и логическая структура расположения элементов. Задание направлено на формирование навыков аккуратного проектирования простого пользовательского интерфейса на этапе знакомства со средой Android Studio.

20. Создать приложение, в котором на первом экране отображаются сведения о мобильной платформе Android, а во второй `Activity` кратко описываются преимущества использования языка Kotlin в мобильной разработке. Для решения необходимо создать две активности, настроить проект и обеспечить корректную навигацию между экранами.

21. Разработать Android-приложение, в котором необходимо изменить стандартную тему оформления проекта и проверить, как это отражается на внешнем виде главного экрана. В ходе выполнения задания следует изучить базовые параметры темы приложения и продемонстрировать результат после повторной сборки и запуска.

22. Создать учебное приложение, в котором на экране отображается текст, составленный на основе строковых ресурсов и значений, задаваемых в Kotlin-коде. Например, название дисциплины можно хранить в ресурсах, а номер группы задавать в коде `Activity`. Итоговая строка должна формироваться после запуска приложения.

23. Разработать Android-приложение, в котором требуется создать и подключить собственный ресурс строки, изменяемый без редактирования Kotlin-кода. После этого

необходимо показать, как изменение значения в `strings.xml` отражается на интерфейсе приложения при следующем запуске.

24. Создать приложение, содержащее стартовый экран с несколькими разделами текста: заголовком, основной информацией и подписью внизу экрана. Для выполнения задания необходимо проработать структуру XML-разметки, выбрать подходящие контейнеры и обеспечить читаемое расположение всех элементов на экране.

25. Разработать Android-приложение, в котором необходимо объединить несколько базовых навыков: создание проекта, анализ структуры каталогов, настройку строковых ресурсов, редактирование XML-разметки, изменение текста из Kotlin-кода и запуск приложения на эмуляторе. Итогом задания должна стать работоспособная учебная программа с осмысленным содержимым главного экрана.

Разработка пользовательского интерфейса Android-приложений

1. Разработать Android-приложение, в котором на главном экране размещены заголовок, краткое пояснение и кнопка выполнения действия. Для решения необходимо подобрать подходящий контейнер разметки, разместить элементы в логичном порядке и обеспечить аккуратное визуальное оформление с использованием отступов и выравнивания.

2. Создать приложение, содержащее на экране `TextView`, `EditText` и `Button`, где пользователь вводит свое имя, а после нажатия кнопки на экране отображается персонализированное приветствие. В задании необходимо реализовать как XML-разметку интерфейса, так и обработчик нажатия в Kotlin-коде.

3. Разработать Android-приложение, в котором пользователь вводит два числа в текстовые поля, а после нажатия кнопки на экране выводится их сумма. Для выполнения задания требуется правильно организовать интерфейс ввода, получить значения из `EditText`, преобразовать их в числовой тип и отобразить результат в `TextView`.

4. Создать приложение, реализующее простейший экран анкеты, содержащий поля для ввода имени, возраста и названия учебной группы. После нажатия кнопки программа должна выводить введенные сведения в виде оформленного текстового блока. Необходимо показать работу сразу с несколькими полями ввода и обновлением содержимого интерфейса.

5. Разработать приложение, в котором на экране располагаются несколько кнопок, каждая из которых при нажатии изменяет текст одного и того же `TextView`. Задание направлено на освоение обработки событий нажатия и связывания нескольких элементов интерфейса с Kotlin-кодом одной `Activity`.

6. Создать Android-приложение, содержащее текстовое поле для ввода температуры в градусах Цельсия и кнопку преобразования. После выполнения действия на экране должна отображаться температура в градусах Фаренгейта. В решении необходимо использовать элементы `EditText`, `Button`, `TextView` и программную обработку пользовательского ввода.

7. Разработать приложение, в котором пользователь вводит длину и ширину прямоугольника, а программа по нажатию кнопки вычисляет и отображает площадь и периметр. Следует предусмотреть логичное расположение элементов интерфейса и читаемое представление результатов вычисления.

8. Создать приложение, в котором пользователь вводит стоимость товара и процент скидки, а затем получает на экране информацию о размере скидки и итоговой цене. В ходе выполнения задания необходимо продемонстрировать работу с вещественными значениями и вывод нескольких связанных результатов в одном интерфейсном блоке.

9. Разработать Android-приложение с использованием элемента `ImageView`, в котором на экране отображается изображение и подпись к нему. Задание предполагает добавление

графического ресурса в проект, размещение изображения в интерфейсе и настройку его размеров и расположения на экране.

10. Создать приложение, в котором размещены заголовок, изображение и кнопка, по нажатию на которую подпись под изображением изменяется. В ходе выполнения задания необходимо использовать как статические элементы XML-разметки, так и программное изменение текста в Kotlin-коде.

11. Разработать приложение, в котором на экране расположено поле ввода числа и кнопка, а после выполнения действия программа определяет, является ли число четным или нечетным, и выводит результат пользователю. Решение должно объединять обработку событий, получение данных из поля ввода и отображение итогового сообщения.

12. Создать Android-приложение, реализующее простейший калькулятор площади круга. Пользователь вводит радиус, нажимает кнопку и получает рассчитанную площадь. В задании необходимо использовать несколько элементов интерфейса, математическое выражение и понятное оформление результата.

13. Разработать приложение, в котором интерфейс содержит несколько `TextView` с различными размерами шрифта, отступами и выравниванием. Цель задания состоит в том, чтобы научиться управлять внешним видом текстовых элементов в XML-разметке и добиваться читаемой структуры экрана.

14. Создать приложение, в котором на экране выводится список коротких учебных сообщений, оформленных как отдельные блоки текста. Для выполнения задания необходимо проработать размещение нескольких текстовых элементов в контейнере и обеспечить визуальное различие между заголовком, основным текстом и пояснением.

15. Разработать Android-приложение, в котором пользователь вводит число дней, а программа выводит количество полных недель и оставшихся дней. Важной частью задания является продуманная организация пользовательского интерфейса, чтобы поля ввода, кнопка и результат были логично расположены и легко воспринимались.

16. Создать приложение, содержащее кнопку очистки формы. Пользователь должен иметь возможность ввести данные в несколько полей, выполнить вычисление, а затем очистить все поля и область вывода отдельной кнопкой. Задание направлено на закрепление работы с несколькими обработчиками событий.

17. Разработать Android-приложение, в котором пользователь вводит три числа, а программа вычисляет их среднее арифметическое и отображает его на экране. В решении необходимо продемонстрировать удобную компоновку интерфейса, извлечение значений из текстовых полей и формирование итогового сообщения.

18. Создать приложение, в котором с помощью `Button` можно изменять внешний текстовый статус на экране, например менять сообщение с "Ожидание ввода" на "Данные обработаны". Это задание ориентировано на закрепление базовой логики взаимодействия между элементами пользовательского интерфейса.

19. Разработать Android-приложение, в котором используются несколько контейнеров разметки для разделения экрана на смысловые области: заголовок, область ввода и область вывода результата. Необходимо подобрать такую структуру XML, чтобы интерфейс выглядел упорядоченно и был удобен для дальнейшего расширения.

20. Создать приложение, в котором пользователь вводит количество минут, а программа по кнопке переводит это значение в секунды и выводит результат на экран. В задании важно продемонстрировать связку между `EditText`, `Button` и `TextView`, а также корректно обработать ввод целого числа.

21. Разработать Android-приложение, содержащее экран с несколькими кнопками выбора действия, например вычисление суммы, произведения или среднего двух чисел. Пользователь вводит два значения, выбирает нужное действие нажатием соответствующей кнопки, после чего

результат отображается на экране. Такое задание позволяет закрепить обработку разных сценариев взаимодействия в рамках одного интерфейса.

22. Создать приложение, в котором необходимо реализовать адаптивное и аккуратное расположение элементов интерфейса с учетом разных размеров экрана. Задание следует выполнить с использованием отступов, выравнивания и осмысленной структуры разметки, чтобы экран оставался понятным и удобным при запуске на эмуляторах с различными параметрами.

23. Разработать Android-приложение, в котором пользователь вводит текст в `EditText`, а программа после нажатия кнопки выводит длину введенной строки. Задание направлено на закрепление работы со строками, пользовательским вводом и выводом результатов через элементы интерфейса Android.

24. Создать приложение, реализующее экран "карточка студента", где часть информации задается в XML-разметке, а часть формируется на основе пользовательского ввода после нажатия кнопки. В результате на экране должна появляться краткая сводка с введенными сведениями, оформленная как единый информационный блок.

25. Разработать Android-приложение, объединяющее несколько базовых элементов интерфейса: текстовые надписи, поля ввода, кнопки и область вывода результата. Например, пользователь может ввести цену и количество товара, нажать кнопку расчета и получить итоговую стоимость. Задание должно продемонстрировать умение проектировать простой, но завершенный пользовательский интерфейс с обработкой действий пользователя.

Навигация и работа с данными в Android-приложениях

1. Разработать Android-приложение, состоящее из двух экранов `Activity`, где на первом экране пользователь вводит свое имя, а после нажатия кнопки переходит на второй экран, на котором отображается персонализированное приветствие. В задании необходимо реализовать переход с помощью явного `Intent` и организовать передачу текстового значения между экранами.

2. Создать приложение, в котором первый экран содержит поля для ввода имени студента, учебной группы и среднего балла, а второй экран отображает эти данные в виде краткой карточки. Для решения необходимо передать между `Activity` сразу несколько значений и корректно извлечь их на принимающем экране.

3. Разработать Android-приложение, в котором пользователь на первом экране вводит два числа и выбирает действие, после чего на втором экране отображается результат вычисления. В ходе выполнения задания необходимо продумать, какие данные следует передавать между экранами, а какие вычисления можно выполнять уже после перехода.

4. Создать приложение с двумя экранами, где первый экран служит формой ввода данных о товаре, а второй показывает краткий чек, содержащий название товара, цену, количество и итоговую стоимость. В решении необходимо использовать `Intent` для передачи данных и обеспечить понятное оформление результата на втором экране.

5. Разработать Android-приложение, в котором на первом экране отображается меню из нескольких кнопок, каждая из которых открывает отдельный экран с разной учебной информацией. Задание направлено на закрепление навигации между экранами и организацию пользовательских сценариев в приложении с несколькими `Activity`.

6. Создать приложение, в котором первый экран содержит краткое описание дисциплины, а по нажатию кнопки открывается второй экран с описанием языка Kotlin. Дополнительно необходимо реализовать кнопку возврата или использовать стандартную навигацию Android, чтобы пользователь мог вернуться к предыдущему экрану.

7. Разработать Android-приложение, в котором пользователь вводит данные на одном экране, а на втором экране эти данные можно отредактировать и вернуть обратно. Задание

предполагает не только переход между `Activity`, но и организацию логики обмена измененными значениями между экранами.

8. Создать приложение, в котором первый экран используется для ввода параметров прямоугольника, а второй экран показывает вычисленные площадь и периметр. В задании необходимо реализовать передачу числовых данных через `Intent`, их извлечение во второй `Activity` и отображение результата в виде развернутого сообщения.

9. Разработать Android-приложение, в котором пользователь вводит на первом экране количество минут, а на втором получает результат перевода в секунды. В решении следует показать полный маршрут работы данных: ввод, передача между экранами, вычисление и визуализация результата.

10. Создать приложение, в котором используется `Fragment` для отображения дополнительной информации на экране. Например, основная `Activity` может содержать форму ввода, а `Fragment` - блок с результатом или справочной информацией. Задание направлено на знакомство с базовой ролью фрагментов в организации интерфейса.

11. Разработать Android-приложение, в котором после выбора действия на главном экране в контейнере той же `Activity` отображается соответствующий `Fragment` с нужным содержимым. В ходе выполнения задания необходимо показать, как фрагменты могут использоваться для переключения частей интерфейса без открытия новых экранов.

12. Создать приложение, в котором один `Fragment` используется для ввода данных, а второй для отображения результата обработки. Взаимодействие между ними должно быть организовано через `Activity` или другой допустимый механизм передачи данных, чтобы пользователь мог видеть связь между вводом и выводом.

13. Разработать Android-приложение, в котором пользователь вводит текст, а после перехода на второй экран приложение сохраняет этот текст и показывает его даже после повторного открытия программы. Для выполнения задания необходимо использовать `SharedPreferences` и реализовать как сохранение, так и считывание значения.

14. Создать приложение, в котором пользователь задает свое имя и название группы, после чего программа сохраняет эти сведения в `SharedPreferences`. При следующем запуске приложения сохраненные данные должны автоматически отображаться на экране без повторного ввода со стороны пользователя.

15. Разработать Android-приложение, в котором пользователь вводит числовой параметр, например размер скидки или налоговую ставку, а программа сохраняет это значение в `SharedPreferences` для дальнейшего использования. После повторного запуска приложения сохраненное значение должно подставляться в соответствующее поле ввода.

16. Создать приложение, в котором на одном экране можно ввести текст заметки, сохранить его локально и затем просмотреть после повторного открытия приложения. Для решения необходимо использовать `SharedPreferences` либо файловое хранение и реализовать базовый сценарий работы с пользовательскими данными.

17. Разработать Android-приложение, в котором пользователь может заполнить небольшую форму и нажать кнопку сохранения. После этого приложение должно вывести подтверждение успешного сохранения, а при следующем открытии показать ранее введенные данные. Задание направлено на закрепление навыков локального хранения информации.

18. Создать приложение, в котором после заполнения формы ввода и перехода на второй экран данные не только отображаются, но и сохраняются для последующего использования. Например, можно сохранить сведения о пользователе или параметры последнего расчета. В решении необходимо объединить навигацию и работу с локальным хранилищем.

19. Разработать Android-приложение, в котором хранится список последних введенных пользователем значений, например последних расчетов или последних строк текста. На первом

этапе достаточно сохранять одно последнее значение, но структура программы должна демонстрировать принцип организации локального хранения и последующего чтения данных.

20. Создать приложение, в котором пользователь выбирает один из нескольких разделов с главного экрана, а приложение открывает соответствующий экран и сохраняет последний выбранный раздел. При повторном запуске программы на экране должно отображаться, какой раздел был открыт последним. Для реализации необходимо совместить навигацию и `SharedPreferences`.

21. Разработать Android-приложение, в котором первый экран выполняет роль меню, второй экран роль формы ввода, а третий экран роль итогового отчета. Между экранами необходимо организовать последовательную передачу данных и продумать навигацию так, чтобы пользователь мог вернуться к предыдущим шагам без потери логики работы приложения.

22. Создать приложение, в котором пользователь на первом экране вводит данные о товаре, на втором подтверждает их, а на третьем получает итоговый чек. В ходе выполнения задания необходимо реализовать многошаговый пользовательский сценарий, включающий переходы между несколькими `Activity` и передачу данных через `Intent`.

23. Разработать Android-приложение, в котором после вычисления результата на втором экране пользователь может сохранить этот результат и затем увидеть его на главном экране приложения при следующем запуске. Такое задание объединяет навигацию, вычислительную логику и постоянное хранение данных.

24. Создать приложение, в котором пользователь вводит несколько строковых параметров, а затем программа отображает их на другом экране в виде структурированного текстового отчета. После этого данные должны быть сохранены локально, чтобы отчет можно было повторно открыть без нового ввода. В задании необходимо объединить передачу данных, визуализацию и локальное хранение.

25. Разработать Android-приложение, объединяющее базовые элементы темы: несколько экранов, переходы между ними, передачу текстовых и числовых данных, использование `SharedPreferences` и организацию логичного пользовательского сценария. Например, приложение может содержать экран ввода сведений о пользователе, экран подтверждения и экран итоговой информации с сохранением введенных данных для последующего запуска программы.

Тестирование, отладка и итоговая разработка мобильного приложения

1. Разработать Android-приложение с простейшей вычислительной логикой, например переводом минут в секунды, а затем выполнить его пошаговую отладку в Android Studio. В ходе выполнения задания необходимо установить точки останова, проследить изменение значений переменных и зафиксировать, как именно проходит выполнение программы при нажатии кнопки.

2. Создать приложение, в котором пользователь вводит два числа для вычисления суммы, после чего необходимо проверить корректность работы программы на нескольких наборах входных данных. Следует подобрать как обычные значения, так и граничные случаи, например ноль, отрицательные числа и большие значения, и убедиться, что результат вычислений отображается верно.

3. Разработать Android-приложение с формой ввода текстовых данных, например имени и группы студента, а затем провести его отладку с помощью логирования. В процессе выполнения задания необходимо добавить вывод диагностических сообщений в ключевые участки программы и проследить, как данные проходят путь от ввода пользователя до отображения результата на экране.

4. Создать приложение, в котором имеется намеренно допущенная ошибка преобразования данных из `EditText`, и выполнить ее поиск и устранение. Например, программа может аварийно завершаться при пустом вводе. Необходимо выявить причину проблемы, исправить код и проверить, что приложение стало устойчиво работать как при корректных, так и при некорректных входных данных.

5. Разработать Android-приложение для вычисления площади прямоугольника, после чего провести его тестирование на нескольких вариантах ввода. Требуется проверить корректность расчетов при целых и вещественных значениях, а также предусмотреть ситуацию, когда пользователь не ввел одно из значений или ввел недопустимый текст.

6. Создать приложение, содержащее две `Activity`, между которыми передаются данные пользователя, а затем проверить корректность этой передачи. В ходе выполнения задания необходимо отследить, какие значения отправляются в `Intent`, какие принимаются на втором экране, и убедиться, что при переходе между экранами информация не теряется и не искажается.

7. Разработать приложение с использованием `SharedPreferences`, которое сохраняет введенные пользователем данные, а затем выполнить тестирование сценария повторного запуска. Необходимо убедиться, что данные корректно сохраняются, повторно загружаются и отображаются после закрытия и нового открытия программы.

8. Создать приложение, в котором реализована форма ввода параметров товара, а результат отображается как краткий чек. После разработки необходимо провести серию проверок, включая корректный ввод, пустые поля, ввод отрицательных значений и слишком больших чисел, а затем устранить обнаруженные недостатки интерфейса и логики.

9. Разработать Android-приложение с несколькими кнопками действий, например сложение, умножение и вычисление среднего, и затем выполнить его функциональное тестирование. Требуется составить набор сценариев, при которых каждая кнопка приводит к ожидаемому результату, а также проверить, что интерфейс остается понятным и не вводит пользователя в заблуждение.

10. Создать приложение, в котором используются `TextView`, `EditText`, `Button` и `ImageView`, а затем проверить корректность отображения интерфейса на эмуляторах с различными размерами экрана. Задание направлено на анализ удобства пользовательского интерфейса, выявление возможных проблем с размещением элементов и их последующее устранение.

11. Разработать Android-приложение, в котором необходимо отследить поведение методов жизненного цикла `Activity` в процессе запуска, сворачивания, возврата к приложению и закрытия. Для выполнения задания следует использовать логирование, а затем на основе наблюдений сделать вывод о последовательности вызова методов и влиянии этих переходов на состояние интерфейса.

12. Создать приложение, в котором пользователь вводит данные в несколько полей, а затем может очистить форму отдельной кнопкой. После реализации необходимо протестировать все пользовательские сценарии, убедиться, что очистка затрагивает все поля и результат вычисления, и при необходимости доработать обработчики событий.

13. Разработать Android-приложение с вычислением индекса массы тела, после чего провести проверку корректности математической формулы и вывода результата. В ходе тестирования необходимо подобрать несколько примеров с заранее известными вычислениями и убедиться, что приложение выдает ожидаемое значение без ошибок округления, критичных для учебной задачи.

14. Создать приложение, в котором отображается карточка студента на втором экране, а затем проверить устойчивость работы программы при частичном отсутствии данных. Например, следует протестировать случаи, когда не заполнено имя, группа или другой параметр, и

доработать приложение так, чтобы оно не завершалось аварийно и информировало пользователя о проблеме.

15. Разработать Android-приложение для учета стоимости покупки, включающее поля для ввода цены, количества и процента скидки. После завершения разработки необходимо провести комплексное тестирование приложения, охватывающее корректность вычислений, отображение результата, реакцию на пустой ввод и общее удобство пользовательского интерфейса.

16. Создать приложение, в котором пользователь вводит текст заметки, сохраняет его и может повторно открыть при следующем запуске программы. После реализации следует провести проверку всего сценария: ввод, сохранение, закрытие приложения, повторный запуск и восстановление данных. Дополнительно необходимо убедиться, что новое сохранение корректно заменяет старое значение.

17. Разработать Android-приложение с тремя экранами, где первый используется для ввода данных, второй для подтверждения, а третий для вывода итогового результата. После завершения разработки необходимо выполнить отладку всей цепочки переходов и протестировать правильность передачи данных на каждом шаге пользовательского сценария.

18. Создать приложение, в котором вычисления выполняются после нажатия кнопки, а результат выводится в отдельный текстовый блок. В ходе отладки необходимо использовать точки останова и пошаговое выполнение, чтобы проследить последовательность обработки пользовательского ввода, преобразования данных и формирования итогового текста.

19. Разработать Android-приложение, в котором часть данных задается через `strings.xml`, а часть формируется в Kotlin-коде. После этого необходимо проверить, как изменение строковых ресурсов влияет на итоговый интерфейс приложения, и убедиться, что приложение корректно использует ресурсы без жестко заданных текстов в коде там, где это не требуется.

20. Создать приложение, в котором необходимо реализовать простую защиту от неверного ввода данных. Например, если пользователь оставил поле пустым или ввел текст вместо числа, приложение должно вывести понятное сообщение об ошибке. После реализации требуется протестировать несколько вариантов некорректного ввода и подтвердить устойчивую работу программы.

21. Разработать итоговое мини-приложение учебного характера, объединяющее форму ввода, вычислительную логику, вывод результата и сохранение последнего состояния. После реализации необходимо выполнить полную проверку приложения: протестировать пользовательский интерфейс, корректность расчетов, сохранение данных и устойчивость работы при повторных запусках.

22. Создать итоговое Android-приложение типа мини-калькулятора, содержащее несколько действий и экран с результатом. После завершения основной реализации необходимо провести поэтапную отладку и документированное тестирование, проверяя работу каждого действия отдельно и анализируя возможные ошибки в пользовательских сценариях.

23. Разработать приложение справочного характера, содержащее несколько экранов с учебной информацией и переходами между ними, а затем проверить корректность навигации. Необходимо удостовериться, что каждый экран открывается в нужный момент, кнопки возврата работают корректно, а отображаемый текст соответствует назначению соответствующего раздела.

24. Создать итоговое учебное приложение, в котором пользователь заполняет данные о себе, получает итоговую карточку на отдельном экране и может сохранить эти сведения для следующего запуска программы. После реализации следует провести комплексную проверку: корректность интерфейса, передачу данных между экранами, сохранение в `SharedPreferences`, повторную загрузку и отсутствие аварийного завершения при неполном вводе.

25. Разработать итоговое мини-приложение под Android, объединяющее ключевые элементы, изученные в курсе: пользовательский интерфейс, обработку событий, переходы между

экранами, передачу данных, локальное хранение, отладку и тестирование. После завершения разработки необходимо провести итоговую проверку работоспособности приложения, выявить и устранить замеченные недостатки, а затем подготовить приложение к демонстрации как завершенный учебный программный продукт.

Типовые задания для промежуточного контроля

Перечень типовых контрольных вопросов для устного опроса на промежуточной аттестации (зачет)

1. Каково назначение языка программирования Kotlin и каковы его основные особенности?
2. В чем состоят основные преимущества языка Kotlin в современной программной разработке?
3. По каким причинам язык Kotlin получил широкое распространение в разработке программного обеспечения?
4. Каково назначение функции `main()` в программе на языке Kotlin?
5. В чем состоит различие между функциями `print()` и `println()`?
6. В чем состоит различие между ключевыми словами `var` и `val` в языке Kotlin?
7. Что понимается под переменной в языке Kotlin?
8. Что понимается под константой в языке Kotlin?
9. Почему при разработке программного кода необходимо использовать осмысленные имена переменных и констант?
10. Что понимается под типом данных в языке Kotlin?
11. Какие базовые числовые типы данных используются в языке Kotlin?
12. В чем состоит различие между типами данных `Int` и `Long`?
13. В чем состоит различие между типами данных `Float` и `Double`?
14. Каково назначение типа данных `Boolean`?
15. В чем состоит различие между типами данных `Char` и `String`?
16. Что понимается под выводом типа в языке Kotlin?
17. В каких случаях целесообразно явно указывать тип переменной?
18. Какие арифметические операции поддерживаются в языке Kotlin?
19. Почему результат деления двух целочисленных значений может отличаться от вещественного результата?
20. Каково назначение операции нахождения остатка от деления?
21. Что понимается под строковыми шаблонами в языке Kotlin?
22. Каково назначение функции `readln()`?
23. По какой причине данные, вводимые пользователем, требуют преобразования типов?
24. Каким образом в языке Kotlin выполняется преобразование строки в числовой тип данных?
25. Каково назначение комментариев в программном коде?
26. Почему соблюдение правил форматирования программного кода является важным требованием при разработке программ?
27. Какие условные конструкции используются в языке Kotlin?
28. В чем состоит различие между конструкциями `if` и `when`?
29. Какие циклические конструкции поддерживаются в языке Kotlin?
30. В каких случаях целесообразно использовать цикл `for`?
31. В каких случаях целесообразно использовать цикл `while`?
32. Каково назначение операторов `break` и `continue`?
33. Что понимается под функцией в языке Kotlin?
34. Что понимается под параметрами функции?
35. Что понимается под возвращаемым значением функции?

36. Каким образом использование функций способствует структурированию программного кода?

37. Что понимается под областью видимости переменной?

38. Какие основные виды коллекций используются в языке Kotlin?

39. В чем состоит различие между `List` и `MutableList`?

40. Каково назначение множества `Set`?

41. Каково назначение словаря `Map`?

42. Какие основные операции могут выполняться над строками в языке Kotlin?

43. Что понимается под классом в языке Kotlin?

44. В чем состоит различие между классом и объектом?

45. Каково назначение свойств и методов класса?

46. Каково назначение конструктора класса?

47. Что понимается под наследованием в языке Kotlin?

48. Каково назначение `dataclass` в языке Kotlin?

49. Что понимается под null-безопасностью в языке Kotlin?

50. Какие средства языка Kotlin используются для обработки исключений?

51. Что представляет собой мобильная платформа Android?

52. По каким причинам платформа Android получила широкое распространение в области мобильной разработки?

53. Каково назначение среды разработки Android Studio?

54. Из каких основных структурных элементов состоит проект Android-приложения?

55. Каково назначение файла `AndroidManifest.xml`?

56. Каково назначение каталога `res` в структуре Android-проекта?

57. Каково назначение файла `strings.xml`?

58. Что понимается под `Activity` в Android-приложении?

59. Что понимается под жизненным циклом `Activity`?

60. Какие основные методы жизненного цикла `Activity` используются в Android-приложении?

61. Каково назначение метода `onCreate()`?

62. В каких ситуациях вызываются методы `onStart()` и `onResume()`?

63. В каких ситуациях вызываются методы `onPause()` и `onStop()`?

64. В каких случаях может вызываться метод `onDestroy()`?

65. Что понимается под пользовательским интерфейсом Android-приложения?

66. Что представляет собой XML-разметка в Android-приложении?

67. Каким образом XML-разметка связана с Kotlin-кодом в Android-приложении?

68. Какие основные элементы пользовательского интерфейса наиболее часто применяются в Android-приложениях?

69. Каково назначение элементов `TextView`, `EditText` и `Button`?

70. Каково назначение контейнеров разметки в Android-приложении?

71. Почему при разработке Android-приложения необходимо учитывать принципы адаптивности пользовательского интерфейса?

72. Что понимается под обработкой событий в Android-приложении?

73. Каким образом организуется обработка нажатия кнопки в Android-приложении?

74. Что понимается под `Intent` в Android?

75. Каково назначение явных `Intent`?

76. В чем состоит различие между явными и неявными `Intent`?

77. Каким образом осуществляется переход между экранами Android-приложения?

78. Каким образом осуществляется передача данных между `Activity`?

79. Что понимается под `Fragment` в Android-приложении?

80. В чем состоит различие между `Activity` и `Fragment`?

81. Какие основные способы локального хранения данных используются в Android-приложениях?

82. Каково назначение механизма `SharedPreferences`?

83. В каких случаях в Android-приложениях используется файловое хранение данных?

84. Какова роль локальных баз данных в Android-приложениях?

85. Что понимается под пользовательским вводом в Android-приложении?

86. Почему необходимо выполнять проверку корректности данных, вводимых пользователем?

87. Какие типичные ошибки могут возникать при обработке пользовательского ввода в Android-приложении?

88. Каково назначение логирования в процессе разработки Android-приложений?

89. Какие возможности предоставляет отладчик Android Studio?

90. Что понимается под точкой останова и каково ее назначение?

91. Почему тестирование является обязательным этапом разработки Android-приложения?

92. Какие виды проверок целесообразно проводить при тестировании учебного Android-приложения?

93. Почему необходимо проверять Android-приложение на различных пользовательских сценариях?

94. В чем состоит значение проверки пользовательского интерфейса на устройствах с различными размерами экранов?

95. Какие проблемы могут возникать при реализации переходов между экранами Android-приложения?

96. Почему необходимо контролировать корректность передачи данных между `Activity`?

97. Какие ошибки могут возникать при использовании `SharedPreferences`?

98. Что включает в себя итоговая отладка Android-приложения перед его демонстрацией?

99. Какие основные этапы включает процесс разработки Android-приложения от проектирования интерфейса до тестирования?

Ситуационные задачи для промежуточной аттестации

1. Разработчику необходимо создать консольную программу на языке Kotlin, которая запрашивает у пользователя стоимость товара и количество единиц, а затем вычисляет итоговую стоимость покупки. В процессе тестирования выяснилось, что программа корректно работает только при вводе целых чисел, тогда как пользователь может вводить цену с дробной частью. Необходимо определить причину проблемы, предложить способ ее устранения и обосновать выбор подходящих типов данных.

2. В программе на Kotlin, предназначенной для вычисления среднего арифметического двух чисел, разработчик использовал переменные типа Int. После проверки результатов было установлено, что при вводе значений 5 и 2 программа выводит 3 вместо 3.5. Необходимо объяснить причину такого результата и предложить корректный вариант реализации вычисления.

3. При разработке программы на Kotlin студент объявил все значения с помощью var, включая те, которые в ходе выполнения программы не изменяются. Преподаватель указал на неудачность такого решения. Необходимо объяснить, в чем состоит проблема, и определить, какие значения в подобной программе следует объявлять как неизменяемые.

4. В ходе выполнения практического задания студент написал программу, которая запрашивает у пользователя целое число и должна определять, является ли оно четным. Однако при вводе текстового значения программа аварийно завершается. Необходимо определить причину ошибки и предложить способ повышения надежности программы.

5. Необходимо разработать консольную программу на Kotlin, которая принимает от пользователя три числа и выводит наибольшее из них. Следует предложить алгоритм решения задачи, определить, какие управляющие конструкции целесообразно использовать, и пояснить, как должна обрабатываться ситуация равенства нескольких значений.

6. В программе на Kotlin реализован ввод оценки студента в баллах. По введенному числу программа должна вывести словесную характеристику результата: неудовлетворительно, удовлетворительно, хорошо или отлично. Необходимо определить, какую конструкцию языка Kotlin предпочтительно использовать для решения этой задачи, и обосновать выбор.

7. Разработчику необходимо создать программу, которая запрашивает у пользователя натуральное число и вычисляет сумму всех чисел от 1 до этого значения. Требуется определить, каким образом следует организовать цикл, какие переменные необходимо использовать и как должен формироваться итоговый результат.

8. В программе на Kotlin необходимо вычислить факториал введенного пользователем числа. При этом разработчик должен учитывать, что значение факториала быстро возрастает. Необходимо предложить решение задачи, указать возможные ограничения и обосновать выбор числового типа данных.

9. Требуется разработать программу на Kotlin, которая получает строку текста и определяет количество слов в ней. Необходимо предложить способ обработки строки, учитывать возможное наличие лишних пробелов и пояснить, какие стандартные средства языка могут быть использованы при решении.

10. В программе на Kotlin необходимо обработать список оценок студента и определить средний балл, максимальную и минимальную оценки. Требуется объяснить, какую коллекцию целесообразно использовать, какие операции необходимо выполнить над ее элементами и как оформить вывод результата.

11. Студент разработал класс Student, содержащий имя, группу и средний балл, однако все действия по выводу данных о студенте выполняются во внешнем коде, а сам класс содержит только свойства. Необходимо оценить данную ситуацию с точки зрения объектно-ориентированного подхода и предложить способ улучшения структуры программы.

12. В программе на Kotlin необходимо реализовать класс Rectangle, который должен хранить длину и ширину прямоугольника, а также предоставлять возможность вычисления площади и периметра. Требуется определить, какие свойства и методы должны быть включены в класс, и обосновать выбранную структуру.

13. При проектировании приложения разработчик использовал nullable-переменную типа String?, но затем попытался напрямую обратиться к ее длине без проверки значения. Необходимо объяснить, почему такое решение является некорректным, и предложить безопасные способы работы с nullable-типами в Kotlin.

14. В программе на Kotlin пользователь вводит строковое значение, которое затем преобразуется в число. Если введено некорректное значение, приложение не должно завершаться с ошибкой. Необходимо предложить два возможных подхода к решению этой задачи: с использованием безопасного преобразования и с использованием обработки исключений.

15. Разработчику необходимо создать простое Android-приложение, содержащее один экран с текстовым заголовком и описанием назначения программы. После создания проекта возник вопрос, в каких файлах следует задавать текст интерфейса и почему не рекомендуется жестко прописывать все строки непосредственно в Kotlin-коде. Необходимо дать развернутое пояснение.

16. В Android-приложении реализован экран с полями EditText для ввода двух чисел и кнопкой вычисления суммы. В процессе тестирования выявлено, что при нажатии кнопки приложение аварийно завершается, если одно из полей остается пустым. Необходимо определить причину такого поведения и предложить способ корректной обработки данной ситуации.

17. В Android-приложении необходимо организовать переход с первого экрана на второй после нажатия кнопки. При этом на втором экране должно отображаться имя пользователя, введенное на первом экране. Требуется описать общий способ решения задачи, указав, какие механизмы Android необходимо использовать для перехода и передачи данных.

18. Разработчик создал вторую Activity в проекте Android, но при попытке перехода на нее приложение завершает работу с ошибкой. Необходимо определить одну из наиболее

вероятных причин подобной ситуации и указать, какие элементы проекта следует проверить в первую очередь.

19. В Android-приложении необходимо сохранить имя пользователя так, чтобы после повторного запуска программы оно автоматически отображалось на экране без повторного ввода. Требуется определить, какой механизм локального хранения данных целесообразно использовать для этой задачи, и объяснить причины такого выбора.

20. Пользователь Android-приложения вводит в форму параметры товара: название, цену и количество. После нажатия кнопки программа должна открыть второй экран и показать краткий чек. Необходимо определить, как организовать передачу нескольких значений между Activity, и пояснить, какие требования следует учитывать при извлечении этих данных на принимающем экране.

21. В Android-приложении с несколькими экранами пользователь после возврата на предыдущий экран обнаруживает, что ранее введенные данные утрачены. Необходимо объяснить, с какими особенностями жизненного цикла Activity может быть связана данная ситуация, и предложить способы сохранения состояния интерфейса.

22. При отладке Android-приложения разработчик заметил, что методы onPause() и onStop() вызываются чаще, чем ожидалось, например при переходе на другой экран или при сворачивании приложения. Необходимо объяснить, почему это происходит, и охарактеризовать место этих методов в жизненном цикле Activity.

23. В Android-приложении реализован экран с несколькими кнопками, каждая из которых должна выполнять отдельное вычислительное действие над введенными числами. В ходе проверки выяснилось, что интерфейс работает, но код обработчиков стал громоздким и трудно поддерживаемым. Необходимо предложить способ структурирования логики приложения и обосновать его с точки зрения качества программного кода.

24. После разработки Android-приложения с пользовательским интерфейсом на одном эмуляторе программа выглядит корректно, однако на устройстве с другим размером экрана элементы интерфейса частично перекрывают друг друга. Необходимо объяснить возможные причины подобной проблемы и указать, какие принципы построения адаптивного интерфейса следует учитывать при разработке.

25. Необходимо разработать итоговое учебное Android-приложение, которое должно включать форму ввода данных, обработку пользовательских действий, переход между экранами, передачу данных и сохранение результатов для повторного запуска. Требуется определить основные этапы проектирования и реализации такого приложения, а также обозначить, какие аспекты должны быть обязательно проверены в ходе итогового тестирования.

Типовые тестовые задания

1. Какое ключевое слово используется в Kotlin для объявления изменяемой переменной?
   а) `let`
   б) `val`
   в) `var`
   г) `const`
   Правильный ответ: в
2. Какое ключевое слово используется в Kotlin для объявления неизменяемого значения?
   а) `val`
   б) `var`
   в) `static`
   г) `final`
   Правильный ответ: а

3. Какая функция является точкой входа в программу на Kotlin?
   а) `start()`
   б) `run()`
   в) `main()`
   г) `init()`
   Правильный ответ: в
4. Какой тип данных в Kotlin предназначен для хранения целых чисел?
   а) `Double`
   б) `Boolean`
   в) `String`
   г) `Int`
   Правильный ответ: г
5. Какой тип данных в Kotlin предназначен для хранения вещественных чисел повышенной точности?
   а) `Float`
   б) `Double`
   в) `Long`
   г) `Char`
   Правильный ответ: б
6. Какой тип данных используется в Kotlin для хранения логических значений?
   а) `Bool`
   б) `Logic`
   в) `Boolean`
   г) `Binary`
   Правильный ответ: в
7. Какая функция используется в Kotlin для вывода текста с переходом на новую строку?
   а) `printLine()`
   б) `println()`
   в) `echo()`
   г) `writeLine()`
   Правильный ответ: б
8. Какое из следующих выражений корректно считывает строку, введенную пользователем?
   а) `input()`
   б) `scan()`
   в) `read()`
   г) `readln()`
   Правильный ответ: г
9. Как преобразовать строку в целое число в Kotlin?
   а) `toInteger()`
   б) `parseInt()`
   в) `toInt()`
   г) `asInt()`
   Правильный ответ: в
10. Что будет результатом выражения `10 / 3`, если оба операнда имеют тип `Int`?
    а) `3.33`
    б) `3`
    в) `4`
    г) ошибка компиляции
    Правильный ответ: б
11. Какая конструкция используется в Kotlin для выбора одного из нескольких вариантов по значению выражения?

а) `switch`
б) `select`
в) `case`
г) `when`

Правильный ответ: г

12. Какой цикл используется в Kotlin для перебора диапазона значений?
    а) `repeat`
    б) `foreach`
    в) `for`
    г) `loop`

    Правильный ответ: в
13. Какой оператор используется для досрочного выхода из цикла?
    а) `skip`
    б) `stop`
    в) `break`
    г) `exit`

    Правильный ответ: в
14. Какой оператор используется для перехода к следующей итерации цикла?
    а) `next`
    б) `continue`
    в) `break`
    г) `pass`

    Правильный ответ: б
15. Что используется в Kotlin для объявления функции?
    а) `def`
    б) `function`
    в) `fun`
    г) `proc`

    Правильный ответ: в
16. Какая коллекция в Kotlin является изменяемым списком?
    а) `List`
    б) `MutableList`
    в) `Set`
    г) `Map`

    Правильный ответ: б
17. Какой тип коллекции не допускает хранения повторяющихся элементов?
    а) `List`
    б) `MutableList`
    в) `Set`
    г) `Array`

    Правильный ответ: в
18. Что представляет собой `dataclass` в Kotlin?
    а) класс для хранения только числовых данных
    б) специальный класс для компактного описания объектов-данных
    в) класс, который нельзя наследовать
    г) класс для работы с файлами

    Правильный ответ: б
19. Какой оператор используется для безопасного обращения к nullable-значению?
    а) `!!`
    б) `?.`
    в) `::`

г) `=>`

Правильный ответ: б

20. Какая конструкция используется в Kotlin для обработки исключений?
    а) `try-catch`
    б) `check-except`
    в) `error-handle`
    г) `guard-case`
    Правильный ответ: а

21. Какая среда разработки наиболее часто используется для создания Android-приложений?
    а) Visual Studio
    б) Android Studio
    в) Eclipse CDT
    г) NetBeans
    Правильный ответ: б
22. Какой файл содержит сведения о компонентах Android-приложения?
    а) `build.gradle`
    б) `settings.xml`
    в) `AndroidManifest.xml`
    г) `MainActivity.kt`
    Правильный ответ: в
23. Как называется основной компонент Android-приложения, представляющий отдельный экран?
    а) `Fragment`
    б) `Service`
    в) `Intent`
    г) `Activity`
    Правильный ответ: г
24. Какой метод жизненного цикла `Activity` вызывается при ее создании?
    а) `onStart()`
    б) `onCreate()`
    в) `onPause()`
    г) `onStop()`
    Правильный ответ: б
25. В каком каталоге Android-проекта обычно размещаются XML-файлы разметки экранов?
    а) `res/layout`
    б) `res/xml`
    в) `assets/layout`
    г) `src/layout`
    Правильный ответ: а
26. Какой файл обычно используется для хранения строковых ресурсов приложения?
    а) `text.xml`
    б) `labels.xml`
    в) `strings.xml`
    г) `resources.xml`
    Правильный ответ: в
27. Какой элемент интерфейса используется для отображения текста?
    а) `EditText`
    б) `Button`
    в) `TextView`

г) `ImageButton`

Правильный ответ: в

28. Какой элемент интерфейса используется для ввода текста пользователем?
    а) `TextView`
    б) `EditText`
    в) `Label`
    г) `TextInput`
    Правильный ответ: б
29. Какой элемент интерфейса используется для обработки нажатия пользователем?
    а) `Button`
    б) `ImageView`
    в) `ScrollView`
    г) `ConstraintLayout`
    Правильный ответ: а
30. Как называется механизм Android, применяемый для перехода между `Activity`?
    а) `Intent`
    б) `BundleView`
    в) `Navigator`
    г) `Router`
    Правильный ответ: а
31. Какой `Intent` используется для открытия конкретной `Activity` внутри приложения?
    а) неявный
    б) системный
    в) явный
    г) глобальный
    Правильный ответ: в
32. Какой метод чаще всего используется для запуска новой `Activity`?
    а) `openActivity()`
    б) `runIntent()`
    в) `startActivity()`
    г) `showActivity()`
    Правильный ответ: в
33. Как можно передать данные из одной `Activity` в другую?
    а) только через файл `AndroidManifest.xml`
    б) через `Intent`extras
    в) только через `SharedPreferences`
    г) только через XML-разметку
    Правильный ответ: б
34. Какой механизм Android подходит для хранения простых пар ключ-значение?
    а) `SQLiteDatabase`
    б) `SharedPreferences`
    в) `IntentFilter`
    г) `RecyclerView`
    Правильный ответ: б
35. Какой компонент Android чаще всего используется для отображения части интерфейса внутри `Activity`?
    а) `Fragment`
    б) `Service`
    в) `BroadcastReceiver`
    г) `Provider`
    Правильный ответ: а

36. Какой метод жизненного цикла `Activity` вызывается, когда экран становится видимым для пользователя?
    а) `onDestroy()`
    б) `onStop()`
    в) `onStart()`
    г) `onRestart()`
    Правильный ответ: в
37. Какой метод жизненного цикла `Activity` вызывается при потере приложением фокуса?
    а) `onPause()`
    б) `onCreate()`
    в) `onResume()`
    г) `onAttach()`
    Правильный ответ: а
38. Для чего используется Logcat в Android Studio?
    а) для рисования интерфейса
    б) для просмотра диагностических сообщений и ошибок
    в) для создания эмулятора
    г) для хранения ресурсов
    Правильный ответ: б
39. Что позволяет сделать точка останова при отладке Android-приложения?
    а) удалить ошибку автоматически
    б) приостановить выполнение программы в нужном месте
    в) скомпилировать проект без ошибок
    г) изменить XML-разметку во время запуска
    Правильный ответ: б
40. Что является обязательным этапом перед демонстрацией учебного Android-приложения?
    а) только переименование проекта
    б) только настройка темы оформления
    в) тестирование и отладка приложения
    г) только публикация в магазине приложений
    Правильный ответ: в

Критерии оценки на этапе зачета по дисциплине

зачет по дисциплине проводятся в форме устного опроса. Зачет проводится по расписанию в компьютерном классе.

Критерии и шкала оценки зачета по дисциплине.

Оценка	Характеристики ответа студента
Зачтено	- студент глубоко и всесторонне усвоил программный материал; - уверенно, логично, последовательно и грамотно его излагает; - опираясь на знания основной и дополнительной литературы, тесно привязывает усвоенные научные положения с практической деятельностью IT-специалиста; - умело обосновывает и аргументирует выдвигаемые им идеи; - делает выводы и обобщения; - аргументирует научные положения; - допускает несущественные ошибки и неточности;
Не зачтено	- студент не усвоил значительной части программного материала; - допускает существенные ошибки и неточности при рассмотрении задач в сфере деятельности IT-специалиста; - испытывает трудности в практическом применении знаний; - не может аргументировать научные положения;

	- не формулирует выводов и обобщений

7.2. Методические материалы, определяющие процедуры оценивания

Методические материалы, определяющие процедуры оценивания в рамках текущего контроля успеваемости

С целью определения уровня овладения компетенциями, закрепленными за дисциплиной, в заданные преподавателем сроки проводится текущий и промежуточный контроль знаний, умений и навыков каждого обучающегося.

Краткая характеристика процедуры реализации текущего и промежуточного контроля для оценки компетенций обучающихся представлена в таблице.

Процедура оценивания	Организация деятельности обучающегося
Выполнение практических заданий/творческих заданий	При выполнении практических заданий/творческих заданий обучающимся необходимо выполнить всю работу согласно тексту задания. Результаты работы сохранить в файлах. После выполнения задания необходимо преподавателю продемонстрировать результаты работы и быть готовым ответить на вопросы и продемонстрировать выполнение отдельных пунктов задания. Защита практических работ осуществляется на практических занятиях.
Устный опрос	Средство контроля, организованное как специальная беседа преподавателя с обучающимся на темы, связанные с изучаемой дисциплиной, и рассчитанное на выяснение объема знаний обучающегося по определенному разделу, теме, проблеме и т.п. Развернутый ответ обучающегося должен представлять собой связное, логически последовательное сообщение на заданную тему, показывать его умение применять определения, правила в конкретных случаях. Показатели для оценки устного ответа: 1) знание материала; 2) последовательность изложения; 3) владение речью и профессиональной терминологией; 4) применение конкретных примеров; 5) знание ранее изученного материала; 6) уровень теоретического анализа; 7) степень самостоятельности; 8) степень активности в процессе; 9) выполнение регламента. Уровень знаний обучающегося определяется оценками «отлично», «хорошо», «удовлетворительно», «неудовлетворительно». Критерии и шкала оценки приведены в п. 3. Фонда оценочных средств.
Зачет	**Зачет** проводится в устной форме по расписанию экзаменационной сессии. Зачет по дисциплине включает в себя: собеседование преподавателя со студентами по контрольным вопросам и

	ситуационным задачам. Контрольный вопрос — это средство контроля усвоения учебного материала дисциплины. Процедура проведения данного оценочного мероприятия включает в себя: беседу преподавателя с обучающимся на темы, связанные с изучаемой дисциплиной, и рассчитанное на выяснение объема знаний обучающегося по определенному разделу, теме дисциплины. Ситуационная задача — это оценочное средство, включающее совокупность условий, направленных на решение практически значимой ситуации с целью формирования компетенций, соответствующих основным типам профессиональной деятельности. Процедура проведения данного оценочного мероприятия включает в себя: оценку правильности решения задачи. В случае вариативности решения задачи следует обосновать все возможные варианты решения. Контрольные вопросы и ситуационные задачи к экзамену доводятся до сведения студентов заранее. Билет к экзамену содержит один контрольный вопрос и одну ситуационную задачу. При подготовке к ответу пользование учебниками, учебно-методическими пособиями, средствами связи и электронными ресурсами на любых носителях запрещено. Время на подготовку ответа – от 30 до 45 минут. По истечении времени подготовки ответа, студент отвечает на вопросы экзаменационного билета. На ответ обучающегося по каждому вопросу билета отводится, как правило, 3-5 минут. После ответа обучающегося преподаватель может задать дополнительные (уточняющие) вопросы в пределах предметной области экзаменационного задания. После окончания ответа преподаватель объявляет обучающемуся оценку по результатам экзамена, а также вносит эту оценку в экзаменационную ведомость, зачетную книжку. Уровень знаний, умений и навыков обучающегося определяется оценками «отлично», «хорошо», «удовлетворительно», «неудовлетворительно». Перечень вопросов к экзамену, а также критерии и шкала оценки приведены в разделе 3. Фонда оценочных средств.

8. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

8.1. Основная литература

1. Соколова, В. В. Разработка мобильных приложений : учебник для среднего профессионального образования / В. В. Соколова. — Москва : Издательство Юрайт, 2025. — 160 с. — (Профессиональное образование). — ISBN 978-5-534-16868-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/566082

2. Соколова, В. В. Вычислительная техника и информационные технологии. Разработка мобильных приложений : учебник для вузов / В. В. Соколова. — Москва : Издательство Юрайт, 2025. — 160 с. — (Высшее образование). — ISBN 978-5-534-16302-5. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/561336

8.2. Дополнительная литература

1. *Зыков, С. В.* Программирование : учебник и практикум для вузов / С. В. Зыков. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2026. — 285 с. — (Высшее образование). — ISBN 978-5-534-16031-4. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/583644

2. Тузовский, А. Ф. Объектно-ориентированное программирование : учебник для вузов / А. Ф. Тузовский. — Москва : Издательство Юрайт, 2025. — 213 с. — (Высшее образование). — ISBN 978-5-534-16316-2. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/561394

Периодические издания:

1. Программные продукты и системы : научный журнал / Научно-исследовательский институт «Центрпрограммсистем». – Тверь, 2010-2026. – ISSN 0236-235X. – Текст : электронный. – URL: https://www.iprbookshop.ru/149185.html
2. Прикладная информатика : научный журнал / Университет «Синергия». – 2006. – Москва, 2006–2025. – ISSN 1993-8314. – Текст : электронный. – URL: https://www.iprbookshop.ru/11770.html

8.3. Программное обеспечение

MicrosoftWindows, Яндекс 360, Microsoft Office Professional Plus 2019, GoogleChrome, Яндекс.Браузер.

8.4. Профессиональные базы данных

База данных IT специалиста – https://info-comp.ru/
База данных программного обеспечения Oracle – https://otus.ru/nest/post/1577/
База данных «Стратегическое управление и планирование – http://www.stplan.ru/
База данных нормативно-правовых актов РФ – https://pravo-search.minjust.ru/bigs/portal.html
База данных по бизнес-планированию – https://biznesplan-primer.ru/
База данных по делопроизводству и документообороту – https://clubtk.ru/osnovy-deloproizvodstva-i-dokumentooborota-dlya-novichkov

8.5. Информационные справочные системы

Справочно-правовая система «КонсультантПлюс» - http://www.consultant.ru/

Поисковые системы

Поисковая система Yandex- https://www.yandex.ru/
Поисковая система Rambler – https://www.rambler.ru/

8.6. Интернет-ресурсы

Научная электронная библиотека - http://www.elibrary.ru/
Научная электронная библиотека «Киберленинка» - http://cyberleninka.ru/
Национальная Электронная Библиотека (НЭБ) – https://rusneb.ru
Образовательная платформа ЮРАЙТ - https://urait.ru/
Электронная библиотечная система «СКСИ» - https://www.sksi.ru/Environment/EbsSksi
Онлайн-курс «Цифровая грамотность» – https://oiledu.ru/courses/ugntu/tsifrovaya-gramotnost.html
Цифровой университет 2035 – https://2035.university
Образовательная платформа «Цифровой гражданин» – https://it-gramota.ru/

8.7. Методические указания по освоению дисциплины

Методические указания для подготовки к лекции

Аудиторные занятия планируются в рамках такой образовательной технологии, как проблемно-ориентированный подход с учетом профессиональных и личностных особенностей обучающихся. Это позволяет учитывать исходный уровень знаний обучающихся, а также существующие технические возможности обучения.

Методологической основой преподавания дисциплины являются научность и объективность.

Лекция является первым шагом подготовки обучающихся к практическим занятиям. Проблемы, поставленные в ней, на практическом занятии приобретают конкретное выражение и решение.

Преподаватель на вводной лекции определяет структуру дисциплины, поясняет цели и задачи изучения дисциплины, формулирует основные вопросы и требования к результатам освоения. При проведении лекций, как правило, выделяются основные понятия и определения. При описании закономерностей обращается особое внимание на сравнительный анализ конкретных примеров.

На первом занятии преподаватель доводит до обучающихся требования к текущей и промежуточной аттестации, порядок работы в аудитории и нацеливает их на проведение самостоятельной работы с учетом количества часов, отведенных на нее учебным планом по направлению подготовки 40.03.01 Юриспруденция и рабочей программой по дисциплине (п. 5.5).

Рекомендуя литературу для самостоятельного изучения, преподаватель поясняет, каким образом максимально использовать возможности, предлагаемые библиотекой АНО ВО СКСИ, в том числе ее электронными ресурсами, а также сделает акцент на привлечение ресурсов сети Интернет и профессиональных баз данных для изучения практики.

Выбор методов и форм обучения по дисциплине определяется:

– общими целями образования, воспитания, развития и психологической подготовки обучающихся;

– особенностями учебной дисциплины и спецификой ее требований к отбору дидактических методов;

– целями, задачами и содержанием материала конкретного занятия;

– временем, отведенным на изучение того или иного материала;

– уровнем подготовленности обучающихся;

– уровнем материальной оснащенности, наличием оборудования, наглядных пособий, технических средств.

Лекции дают обучающимся систематизированные знания по дисциплине, концентрируют их внимание на наиболее сложных и важных вопросах.

Лекции обычно излагаются в традиционном или в проблемном стиле (интерактивном). Интерактивный стиль позволяет стимулировать активную познавательную деятельность обучающихся и их интерес к дисциплине, формировать творческое мышление, прибегать к противопоставлениям и сравнениям, делать обобщения, активизировать внимание обучающихся

путем постановки проблемных вопросов, поощрять дискуссию. Во время лекционных занятий рекомендуется вести конспектирование учебного материала, обращать внимание на формулировки и категории, раскрывающие суть того или иного явления или процессов, выводы и практические рекомендации.

В конце лекции делаются выводы и определяются задачи на самостоятельную работу. Во время лекционных занятий рекомендуется вести конспектирование учебного материала, обращать внимание на формулировки и категории, раскрывающие суть того или иного явления или процессов, научные выводы и практические рекомендации. В случае недопонимания какой-либо части предмета следует задать вопрос в установленном порядке преподавателю.

Конспект – это систематизированное, логичное изложение материала источника. Различаются четыре типа конспектов:

План-конспект – это развернутый детализированный план, в котором достаточно подробные записи приводятся по тем пунктам плана, которые нуждаются в пояснении.

Текстуальный конспект – это воспроизведение наиболее важных положений и фактов источника.

Свободный конспект – это четко и кратко сформулированные (изложенные) основные положения в результате глубокого осмысливания материала. В нем могут присутствовать выписки, цитаты, тезисы; часть материала может быть представлена планом.

Тематический конспект – составляется на основе изучения ряда источников и дает более или менее исчерпывающий ответ по какой-то схеме (вопросу).

Подготовленный конспект и рекомендуемая литература используются при подготовке к и практическим занятиям. Подготовка сводится к внимательному прочтению учебного материала, к выводу с карандашом в руках всех утверждений, к решению примеров, задач, к ответам на вопросы. Примеры, задачи, вопросы по теме являются средством самоконтроля.

Методические указания по подготовке к практическим работам

Целью практических работ является углубление и закрепление теоретических знаний, полученных обучающимися на лекциях и в процессе самостоятельного изучения учебного материала, а, следовательно, формирование у них определенных умений и навыков.

В ходе подготовки к практическим работам необходимо прочитать конспект лекции, изучить основную литературу, ознакомиться с дополнительной литературой, выполнить выданные преподавателем задания. При этом учесть рекомендации преподавателя и требования программы. Дорабатывать свой конспект лекции, делая в нем соответствующие записи из литературы. Желательно при подготовке к практическим работам по дисциплине одновременно использовать несколько источников, раскрывающих заданные вопросы.

Методические указания для выполнения самостоятельной работы

Самостоятельная работа обучающихся заключается:

В целях наиболее эффективного изучения дисциплины подготовлены различные задания, различающиеся по преследуемым целям.

Задания представлены – 1) контрольными вопросами, предназначенными для самопроверки; 2) письменными заданиями, включающими задачи и задание.

Задачи самостоятельной внеаудиторной работы обучающихся заключаются в продолжении изучения теоретического материала дисциплины и в развитии навыков самостоятельного анализа литературы.

I. Самостоятельное теоретическое обучение предполагает освоение студентом во внеаудиторное время рекомендуемой преподавателем основной и дополнительной литературы. С этой целью обучающимся рекомендуется постоянно знакомиться с классическими теоретическими источниками по темам дисциплины, а также с новинками литературы, статьями в периодических изданиях, справочных правовых системах.

Для лучшего понимания материала целесообразно осуществлять его конспектирование с возможным последующим его обсуждением на практических занятиях, на научных семинарах и в индивидуальных консультациях с преподавателем. Формы конспектирования материала могут быть различными:

1) обобщение – при подготовке такого конспекта студентом осуществляется анализ и обобщение всех существующих в доктрине подходов по выбранному дискуссионному вопросу раздела, в том числе, дореволюционных ученых, ученых советского и современного периода развития. Основная задача обучающегося заключается не только в изложении точек зрения по исследуемому вопросу, но и в выражении собственной позиции с соответствующим развернутым теоретическим обоснованием.

2) рецензия – при подготовке такого конспекта студентом осуществляется рецензирование выбранного источника по изучаемому дискуссионному вопросу, чаще всего, статьи и периодическом издании, тезисов выступления на конференции либо главы из монографии. Для этого студентом дается оценка содержанию соответствующего источника по следующим параметрам: актуальность выбранной темы, в том числе убедительность обоснования актуальности исследования автором; соответствие содержания работы ее названию; логичность, системность и аргументированность (убедительность) выводов автора; научная добросовестность (наличие ссылок на использованные источники, самостоятельность исследования, отсутствие фактов недобросовестных заимствований текстов, идей и т.п.); научная новизна и др.

Формами контроля за самостоятельным теоретическим обучением являются теоретические опросы, которые осуществляются преподавателем на практических занятиях в устной форме, преследующие цель проверки знаний обучающихся по основным понятиям и терминам по теме дисциплины. В случае представления студентом выполненного им в письменном виде конспекта по предложенным вопросам темы, возможна его защита на практическом занятии или в индивидуальном порядке.

II. Ключевую роль в планировании индивидуальной траектории обучения по дисциплине играет *опережающая самостоятельная работа* (ОПС). Такой тип обучения предлагается в замену традиционной репродуктивной самостоятельной работе (самостоятельное повторение учебного материала и рассмотренных на занятиях алгоритмов действий, выполнение по ним аналогичных заданий). ОПС предполагает следующие виды самостоятельных работ:

познавательно-поисковая самостоятельная работа, предполагающая подготовку докладов, выступлений на практических занятиях, подбор литературы по конкретной проблеме, написание рефератов и др.;

творческая самостоятельная работа, к которой можно отнести выполнение специальных творческих и нестандартных заданий. Задача преподавателя на этапе планирования самостоятельной работы – организовать ее таким образом, чтобы максимально учесть индивидуальные способности каждого обучающегося, развить в нем познавательную потребность и готовность к выполнению самостоятельных работ все более высокого уровня. Студенты, приступая к изучению тем, должны применить свои навыки работы с библиографическими источниками и рекомендуемой литературой, умение четко формулировать свою собственную точку зрения и навыки ведения научных дискуссий. Все подготовленные и представленные тексты должны являться результатом самостоятельной информационно-аналитической работы обучающихся. На их основе студенты готовят материалы для выступлений в ходе практических занятий.

Подготовка к устному опросу

Самостоятельная работа обучающихся включает подготовку к устному опросу на практических занятиях. Для этого студент изучает лекции, основную и дополнительную литературу, публикации, информацию из Интернет-ресурсов.Кроме того, изучению должны быть подвергнуты различные источники права, как регламентирующие правоотношения, возникающие в рамках реализации основ права, так и отношения, что предопределяют реализацию их, либо следуют за ними.

Тема и вопросы к практическим занятиям по дисциплине доводятся до обучающихся заранее. Эффективность подготовки обучающихся к устному опросу зависит от качества ознакомления с рекомендованной литературой. Для подготовки к устному опросу студенту необходимо ознакомиться с материалом, посвященным теме практического занятия, в

рекомендованной литературе, записях с лекционного занятия, обратить внимание на усвоение основных понятий дисциплины, выявить неясные вопросы и подобрать дополнительную литературу для их освещения, составить тезисы выступления по отдельным проблемным аспектам. В среднем, подготовка к устному опросу по одному практическому занятию занимает от 2 до 4 часов в зависимости от сложности темы и особенностей организации студентом своей самостоятельной работы.

Методические указанияк подготовке и проведению лекции с элементами дискуссии, постановкой проблем

Правильно организованная дискуссия проходит три стадии развития: ориентация, оценка и консолидация.

На первой стадии вырабатывается определенная установка на решение поставленной проблемы. При этом перед преподавателем (организатором дискуссии) ставятся следующие задачи:

1. Сформулировать проблему и цели дискуссии. Для этого надо объяснить, что обсуждается, что должно дать обсуждение.
2. Создать необходимую мотивацию, т.е. изложить проблему, показать ее значимость, выявить в ней нерешенные и противоречивые вопросы, определить ожидаемый результат (решение).
3. Установить регламент дискуссии, а точнее, регламент выступлений, так как общий регламент определяется продолжительностью практического занятия.
4. Сформулировать правила ведения дискуссии, основное из которых — выступить должен каждый.
5. Добиться однозначного семантического понимания терминов, понятий и т п.

Вторая стадия — стадия оценки — обычно предполагает ситуацию сопоставления, конфронтации и даже конфликта идей. На этой стадии перед преподавателем ставятся следующие задачи:

1. Начать обмен мнениями, что предполагает предоставление слова конкретным участникам.
2. Собрать максимум мнений, идей, предложений. Для этого необходимо активизировать каждого обучающегося. Выступая со своим мнением, студент может сразу внести свои предложения, а может сначала просто выступить, а позже сформулировать свои предложения.
3. Не уходить от темы, что требует некоторой твердости организатора, а иногда даже авторитарности. Следует тактично останавливать отклоняющихся, направляя их в заданное «русло»,
4. Поддерживать высокий уровень активности всех участников. Не допускать чрезмерной активности одних за счет других, соблюдать регламент, останавливать затянувшиеся монологи, подключать к разговору всех присутствующих обучающихся.
5. Оперативно проводить анализ высказанных идей, мнений, позиций, предложений перед тем, как переходить к следующему витку дискуссии. Такой анализ, предварительные выводы или резюме целесообразно делать через определенные интервалы (каждые 10—15 минут), подводя при этом промежуточные итоги.
6. В конце дискуссии предоставить право обучающимся самим оценить свою работу (рефлексия).

Третья стадия — стадия консолидации — предполагает выработку определенных единых или компромиссных мнений, позиций, решений. На этом этапе осуществляется контролирующая функция. Задачи, которые должен решить преподаватель, можно сформулировать следующим образом:

1. Проанализировать и оценить проведенную дискуссию, подвести итоги, результаты. Для этого надо сопоставить сформулированную в начале дискуссии цель с полученными результатами, сделать выводы, вынести решения, оценить результаты, выявить их положительные и отрицательные стороны.

2. Помочь участникам дискуссии прийти к согласованному мнению, чего можно достичь путем внимательного выслушивания различных толкований, поиска общих тенденций для принятия решений.

3. Принять групповое решение совместно с участниками. При этом следует подчеркнуть важность разнообразных позиций и подходов.

4. В заключительном слове подвести группу к конструктивным выводам, имеющим познавательное и практическое значение.

Составной частью любой дискуссии является процедура *вопросов и ответов*.

С функциональной точки зрения, все вопросы можно разделить на две группы:

- *Уточняющие (закрытые)* вопросы, направленные на выяснение истинности или ложности высказываний, грамматическим признаком которых обычно служит наличие в предложении частицы «ли», например: «Верно ли что?», «Правильно ли я понял, что?». Ответить на такой вопрос можно только «да» или «нет».
- *Восполняющие (открытые)* вопросы, направленные на выяснение новых свойств или качеств интересующих нас явлений, объектов. Их грамматический признак — наличие вопросительных слов: *что, где, когда, как, почему* и т.д.

Методические указания по подготовке к промежуточной аттестации

Промежуточная аттестация по дисциплине проводится в форме экзамена.

Для допуска к экзамену студенту необходимо выполнить и успешно сдать практические работы (практические задания) по каждой теме.

При подготовке к экзамену необходимо повторить конспекты лекций по всем разделам дисциплины. До экзамена обычно проводится консультация, но она не может возместить отсутствия систематической работы в течение триместра и помочь за несколько часов освоить материал, требующийся к экзамену. На консультации студент получает лишь ответы на трудные или оставшиеся неясными вопросы. Польза от консультации будет только в том случае, если студент до нее проработает весь материал.

На экзамене студент должен подтвердить усвоение учебного материала, предусмотренного рабочей программой дисциплины, а также продемонстрировать приобретенные навыки адаптации полученных теоретических знаний к своей профессиональной деятельности. Экзамен проводится в форме устного собеседования по контрольным вопросам, а также обучающемуся необходимо решить ситуационную задачу.

9. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Для реализации дисциплины необходимо следующее материально-техническое обеспечение:

- для проведения занятий лекционного типа - аудитория, оборудованная мультимедийными средствами обучения: проектором, ПК, экраном, доской;

- для проведения лабораторных занятий - компьютерный класс с предустановленным программным обеспечением, указанным в п.8.3.

- для проведения промежуточной аттестации - компьютерный класс с предустановленным программным обеспечением, указанным в п.8.3.

- практическая подготовка - компьютерный класс с предустановленным программным обеспечением, указанным в п.8.3.

- для самостоятельной работы: помещение для самостоятельной работы с возможностью подключения к информационно-коммуникационной сети «Интернет» и обеспечением доступав электронную информационно-образовательную среду.

10.ОСОБЕННОСТИ ОСВОЕНИЯ ДИСЦИПЛИНЫ ЛИЦАМИ С ОГРАНИЧЕННЫМИ ВОЗМОЖНОСТЯМИ ЗДОРОВЬЯ

Обучающимся с ограниченными возможностями здоровья предоставляются специальные учебники, учебные пособия и дидактические материалы, специальные технические средства обучения коллективного и индивидуального пользования, услуги ассистента (тьютора), оказывающего обучающимся необходимую техническую помощь, а также услуги сурдопереводчиков и тифлосурдопереводчиков.

Освоение дисциплины обучающимися с ограниченными возможностями здоровья и инвалидами может быть организовано совместно с другими обучающимися, а также в отдельных группах.

Освоение дисциплины обучающимися с ограниченными возможностями здоровья и инвалидами осуществляется с учетом особенностей психофизического развития, индивидуальных возможностей и состояния здоровья.

В целях доступности получения среднего профессионального образования по образовательной программе лицами с ограниченными возможностями здоровья при освоении дисциплины обеспечивается:

1) для лиц с ограниченными возможностями здоровья по зрению:

– присутствие тьютора, оказывающий студенту необходимую техническую помощь с учетом индивидуальных особенностей (помогает занять рабочее место, передвигаться, прочитать и оформить задание, в том числе, записывая под диктовку),

– письменные задания, а также инструкции о порядке их выполнения оформляются увеличенным шрифтом,

– специальные учебники, учебные пособия и дидактические материалы (имеющие крупный шрифт или аудиофайлы),

– индивидуальное равномерное освещение не менее 300 люкс,

– при необходимости студенту для выполнения задания предоставляется увеличивающее устройство;

2) для лиц с ограниченными возможностями здоровья по слуху:

– присутствие ассистента, оказывающий студенту необходимую техническую помощь с учетом индивидуальных особенностей (помогает занять рабочее место, передвигаться, прочитать и оформить задание, в том числе, записывая под диктовку),

– обеспечивается наличие звукоусиливающей аппаратуры коллективного пользования, при необходимости обучающемуся предоставляется звукоусиливающая аппаратура индивидуального пользования;

– обеспечивается надлежащими звуковыми средствами воспроизведения информации;

3) для лиц с ограниченными возможностями здоровья, имеющих нарушения опорно-двигательного аппарата:

– письменные задания выполняются на компьютере со специализированным программным обеспечением или надиктовываются тьютору;

– по желанию студента задания могут выполняться в устной форме.