АКТ(ф) СПбГУТ

**ОТЧЕТ**

по практическим работам

по

МДК 01.04 Разработка мобильных приложений

Студент ИСПП-21 20.06.2025 Э. А. Язиков

Преподаватель 20.06.2025 Р. В. Садовский

Архангельск 2025

# Лабораторная работа №1 Установка инструментария и настройка среды для разработки мобильных приложений

1. Цель работы
   1. Изучить процесс установки среды для разработки мобильных

приложений.

1. Ответы на контрольные вопросы
   1. Какие среды разработки поддерживают создание мобильных приложений?  
      Ответ: Android Studio, Xcode, Visual Studio, React Native, Flutter.
   2. На каких языках программирования можно создавать мобильные приложения?  
      Ответ: Java, Kotlin, Swift, Objective-C, C#, Dart, JavaScript.
   3. Какие языки программирования доступны в Android Studio?  
      Ответ: Java и Kotlin.
   4. Какой язык программирования является основным для разработки приложений под ОС Android?  
      Ответ: Kotlin.
   5. Что дополнительно нужно установить для работы Android Studio?  
      Ответ: JDK и SDK Android.
   6. Что такое Java Development Kit и что входит в его состав?  
      Ответ: Комплект для разработки на Java; включает компилятор javac, JVM, инструменты отладки и стандартную библиотеку.
2. Вывод

В ходе проделанной лабораторной работы изучил процесс установки среды для разработки мобильных приложений.

# Лабораторная работа №2 Установка среды разработки мобильных приложений с применением виртуальной машины

1. Цель работы
   1. Изучить процесс установки среды для разработки мобильных приложений с применением виртуальной машины.
2. Ответы на контрольные вопросы
   1. Что такое «виртуальная машина»?  
      Ответ: Программное обеспечение, которое эмулирует работу физического компьютера или устройства, позволяя запускать программы независимо от аппаратной платформы.
   2. Какие IDE позволяют создавать приложения для Android?  
      Ответ: Android Studio, IntelliJ IDEA, Eclipse (с плагинами).
   3. Какие языки программирования доступны в IntelliJ IDEA?  
      Ответ: Java, Kotlin, Groovy, Scala, JavaScript и другие.
   4. Какие языки программирования доступны в Android Studio?  
      Ответ: Java и Kotlin.
   5. На каких языках программирования можно разрабатывать нативные приложения для Android?  
      Ответ: Java и Kotlin.
3. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс установки среды для разработки мобильных приложений с применением виртуальной машины.

# Лабораторная работа №3 Разработка линейных алгоритмов

1. Цель работы
   1. Изучить процесс разработки линейных алгоритмов в приложениях на языке Kotlin.
   2. Изучить процесс ввода и вывода данных в приложениях на языке Kotlin.
2. Ответы на контрольные вопросы
   1. Как объявить переменные на Kotlin?

Ответ: Используя ключевые слова var (для изменяемых) или val (для неизменяемых), например: var x = 5, val y = 10.

* 1. Как выполнить ввод данных на Kotlin?

Ответ: Через функцию readLine(), например: val name = readLine().

* 1. Как выполнить вывод данных на Kotlin?

Ответ: С помощью функции println(), например: println("Hello, World!").

* 1. Как преобразовать значение из строкового в числовой?

Ответ: Используя методы, например: "123".toInt() или "123".toDouble().

* 1. Как округлить данные на Kotlin?

Ответ: Используя функции kotlin.math.round(), например: round(3.6).

* 1. Как сгенерировать случайное число на Kotlin?

Ответ: Через класс Random, например: Random.nextInt(), или с помощью функции kotlin.random.Random.nextInt().

* 1. В чем отличие между ключевыми словами var и val в Kotlin?

Ответ: var — переменная, которую можно изменять; val — неизменяемая переменная (константа).

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс разработки линейных алгоритмов в приложениях на языке Kotlin.
   2. В ходе проделанной лабораторной работы изучил процесс ввода и вывода данных в приложениях на языке Kotlin.

# Лабораторная работа №4 Разработка разветвляющихся алгоритмов

1. Цель работы
   1. Изучить процесс разработки разветвляющихся алгоритмов в приложениях на Kotlin.
2. Ответы на контрольные вопросы
   1. Каков синтаксис условного оператора на Kotlin?

Ответ: if (условие) {

// действия, если условие истинно

} else {

// действия, если условие ложно

}

* 1. Каков синтаксис оператора множественного выбора на Kotlin?

Ответ: when (выражение) {

значение1 -> { // действия }

значение2 -> { // действия }

else -> { // действия по умолчанию }

}

* 1. Каков синтаксис тернарного оператора на Kotlin?

Ответ: val result = if (условие) значение1 else значение2

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс разработки разветвляющихся алгоритмов в приложениях на Kotlin.

# Лабораторная работа №5 Разработка циклических алгоритмов

1. Цель работы
   1. Изучить процесс разработки циклических алгоритмов в приложениях на Kotlin.
2. Ответы на контрольные вопросы
   1. Каков синтаксис оператор for на Kotlin?

Ответ: for (элемент in коллекция) {

// действия с элементом

}

* 1. Каков синтаксис оператора while на Kotlin?

Ответ: while (условие) {

// действия

}

* 1. Каков синтаксис оператора do-while на Kotlin?

Ответ: do {

// действия

} while (условие)

* 1. Какие операторы досрочного выхода из цикла применяются в Kotlin?

Ответ: break — для выхода из цикла, continue — для пропуска текущей итерации и перехода к следующей.

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс разработки циклических алгоритмов в приложениях на Kotlin.

# Лабораторная работа №6 Разработка и вызов функций

1. Цель работы
   1. Изучить процесс разработки функций в приложениях на Kotlin.
2. Ответы на контрольные вопросы
   1. Каков синтаксис создания функций на Kotlin?

Ответ: fun имяФункции(параметры): типВозврата {

// тело функции

}

* 1. Как указать для параметров методов значения по умолчанию на Kotlin?

Ответ: fun имяФункции(параметр: тип = значение по умолчанию) { ... }

* 1. Как указать для параметров переменное количество аргументов на Kotlin?

Ответ: fun имяФункции(vararg аргументы: тип) { ... }

* 1. Что такое функции высокого порядка в Kotlin?

Ответ: Функции, которые принимают другие функции в качестве параметров или возвращают функции в качестве результата.

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс разработки функций в приложениях на Kotlin.

# Лабораторная работа №7 Разработка классов

1. Цель работы
   1. Изучить процесс разработки и применения классов на языке Kotlin.
2. Ответы на контрольные вопросы
   1. Как объявить класс на Kotlin?

Ответ: class ИмяКласса {

// свойства и методы

}

* 1. Как объявить свойства в Kotlin?

Ответ: var имяСвойства: Тип = значение // изменяемое свойство

val имяСвойства: Тип = значение // неизменяемое свойство

* 1. Каков синтаксис объявления геттера и сеттера свойства в Kotlin?

Ответ: var свойство: Тип

get() = поле

set(value) { поле = value }

* 1. Как объявить конструктор в Kotlin?

Ответ: class ИмяКласса constructor(параметры) {

// тело класса

}

* 1. Как создать объект в Kotlin?

Ответ: val объект = ИмяКласса(параметры)

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс разработки и применения классов на языке Kotlin.

# Лабораторная работа №8 Наследование классов

1. Цель работы
   1. Изучить процесс разработки дочерних классов в приложениях на Kotlin.
2. Ответы на контрольные вопросы
   1. Что такое «наследование»?

Ответ: Механизм, позволяющий создавать новый класс на основе существующего, наследуя его свойства и методы, что способствует повторному использованию кода и расширению функциональности.

* 1. Сколько родительских классов может быть у класса в Kotlin?

Ответ: В Kotlin класс может иметь только один родительский класс (одинарное наследование).

* 1. Сколько интерфейсов может реализовывать класс в Kotlin?

Ответ: Неограниченное количество интерфейсов.

* 1. Как указать родительский класс на Kotlin?

Ответ: В объявлении класса после двоеточия указывается имя родительского класса и интерфейсы

* 1. Как переопределить реализацию метода в дочернем классе на Kotlin?

Ответ: Используя ключевое слово override

* 1. Как вызвать реализацию родительского метода в дочернем классе на Kotlin?

Ответ: Через ключевое слово super

* 1. Как указать, что класс может быть родительским?

Ответ: Объявить его как open

* 1. Как указать, что у класса нельзя создавать дочерние классы?

Ответ: Объявить его как final (по умолчанию все классы финальны в Kotlin), или явно указать final (по умолчанию).

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс разработки дочерних классов в приложениях на Kotlin.

# Лабораторная работа №9 Обработка коллекций

1. Цель работы
   1. Изучить процесс разработки дочерних классов в приложениях на Kotlin.
2. Ответы на контрольные вопросы
   1. Какие стандартные типы данных коллекций имеются в Kotlin?

Ответ: В Kotlin существуют основные типы коллекций:

List — список (упорядоченная коллекция, допускающая дублирование элементов)

Set — множество (неупорядоченная коллекция уникальных элементов)

Map — словарь (коллекция пар ключ-значение)

* 1. Как объявить коллекцию в Kotlin?

Ответ: val список = listOf(элементы) // неизменяемый список

val mutableСписок = mutableListOf(элементы) // изменяемый список

* 1. Как добавить элемент в список в Kotlin?

Ответ: mutableСписок.add(элемент)

* 1. Как добавить элемент в словарь в Kotlin?

Ответ: mutableСловарь[ключ] = значение

* 1. Как перебрать элементы коллекции в Kotlin?

Ответ: for (элемент in коллекция) {

// действия с элементом

}

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс разработки дочерних классов в приложениях на Kotlin.

# Лабораторная работа №10 Использование корутин

1. Цель работы
   1. Изучить процесс создания и использования сопрограмм в приложениях на Kotlin.
2. Ответы на контрольные вопросы
   1. Что такое «корутина»?

Ответ: Корутина — это легковесная единица асинхронного выполнения, которая позволяет выполнять задачи параллельно без блокировки основного потока, обеспечивая эффективное управление асинхронными операциями.

* 1. Для чего используется launch?

Ответ: launch используется для запуска новой корутины, которая выполняется асинхронно и не возвращает результат (возвращает Job), позволяя управлять её выполнением и отменой.

* 1. Какие особенности использования у suspend-функций?

Ответ: suspend-функции могут приостанавливать выполнение без блокировки потока, их можно вызывать только из других suspend-функций или внутри корутин. Они позволяют писать асинхронный код в стиле последовательных вызовов.

* 1. Как приостановить выполнение корутины?

Ответ: Используя функцию delay(), которая приостанавливает выполнение корутины на заданное время без блокировки потока.

* 1. Как отменить выполнение корутины?

Ответ: Вызвав метод cancel() у объекта Job, связанного с корутиной, или используя функции отмены внутри корутины.

* 1. Как подключить библиотеку для работы с корутинами?

Ответ: Добавив зависимость в файл сборки проекта

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс создания и использования сопрограмм в приложениях на Kotlin.

# Лабораторная работа №11 Создание эмуляторов и подключение устройств

1. Цель работы
   1. Изучить процесс создания эмуляторов и подключения устройств для мобильной разработки.
2. Ответы на контрольные вопросы
   1. Что такое AVD?

Ответ: AVD (Android Virtual Device) — это виртуальное устройство, эмулятор Android, которое позволяет запускать и тестировать приложения на виртуальной копии устройства с определенными характеристиками (размер экрана, версия ОС, аппаратные параметры).

* 1. Что такое эмулятор?

Ответ: Эмулятор — программное обеспечение, имитирующее работу физического устройства, позволяющее запускать и тестировать приложения в среде, похожей на реальное устройство.

* 1. Что определяет профиль устройства?

Ответ: Профиль устройства задает параметры виртуального устройства, такие как модель, размер экрана, разрешение, версию Android, аппаратные характеристики и другие настройки.

* 1. Как создать новый эмулятор?

Ответ: В Android Studio через AVD Manager нажатием "Create Virtual Device", выбор модели устройства и настройка параметров.

* 1. Как указать расширенные настройки эмулятора?

Ответ: В настройках AVD при создании или редактировании выбрать "Show Advanced Settings" и настроить параметры, такие как объем памяти, графика, параметры сети и другие.

* 1. Какие эмуляторы могут использоваться вместо эмуляторов Android Studio?

Ответ: Можно использовать сторонние эмуляторы, такие как Genymotion или физические устройства для тестирования приложений.

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс создания эмуляторов и подключения устройств для мобильной разработки.

# Лабораторная работа №12 Настройка режима терминала

1. Цель работы
   1. Изучить процесс использования терминала при разработке мобильных приложений Android.
2. Ответы на контрольные вопросы
   1. Что такое adb?

Ответ: ADB (Android Debug Bridge) — это командная строка и инструмент, позволяющий взаимодействовать с устройствами Android для выполнения команд, отладки, установки приложений, получения информации и управления устройством.

* 1. Как узнать путь к папке с adb?

Ответ: Обычно adb находится в папке SDK Android

* 1. Как запустить утилиту adb?

Ответ: Откройте командную строку или терминал и введите adb — если путь настроен, появится список команд; иначе укажите полный путь к adb.

* 1. Как узнать список доступных команд adb?

Ответ: adb --help

* 1. Как подключиться к shell из adb?

Ответ: adb shell

* 1. Как сохранить скриншот и видеозапись экрана из adb?

Ответ: adb shell screencap /sdcard/screenshot.png

adb pull /sdcard/screenshot.png ./сохранить\_локально.png

* 1. Как выполнить обмен файлами между ПК и устройством на Android из adb?

Ответ: adb push локальный\_файл /путь/на/устройстве/

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс использования терминала при разработке мобильных приложений Android.

# Лабораторная работа №13 Создание нового проекта

1. Цель работы
   1. Изучить процесс создания приложения в Android Studio.
2. Ответы на контрольные вопросы
   1. Что такое Jetpack Compose?

Ответ: Jetpack Compose — это современная декларативная библиотека для создания пользовательских интерфейсов в Android, которая позволяет писать UI-код проще и быстрее, используя Kotlin.

* 1. Какой тип проекта нужно выбрать, чтобы создать проект с использованием Jetpack Compose?

Ответ: При создании нового проекта в Android Studio нужно выбрать шаблон "Empty Compose Activity" или активировать поддержку Compose при настройке проекта

* 1. Что такое Activity?

Ответ: Activity — это основной компонент Android-приложения, представляющий один экран пользовательского интерфейса, через который пользователь взаимодействует с приложением.

* 1. Для чего используется MainActivity?

Ответ: MainActivity — это обычно стартовая активность приложения, которая запускается при его запуске и содержит основную логику отображения UI.

* 1. Для чего используется функция onCreate?

Ответ: onCreate — это метод жизненного цикла Activity, в котором происходит инициализация компонентов, установка разметки и подготовка интерфейса.

* 1. Что делает функция setContent?

Ответ: setContent задает содержимое пользовательского интерфейса внутри Activity в Jetpack Compose, заменяя традиционный вызов setContentView() и позволяя писать UI на Kotlin.

* 1. Для чего используется функция @Composable?

Ответ: Аннотация @Composable обозначает функцию как компонент UI в Jetpack Compose, который может быть вызван внутри других composable-функций для построения интерфейса.

* 1. Для чего указывается @Preview?

Ответ: @Preview используется для предварительного просмотра composable-компонентов в Android Studio без запуска приложения на устройстве или эмулят

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс создания приложения в Android Studio.

# Лабораторная работа №14 Изучение и комментирование кода

1. Цель работы
   1. Изучить процесс документирования и комментирования приложения Android.
2. Ответы на контрольные вопросы
   1. Как оформляются комментарии в силе KDoc?

Ответ: Комментарии в стиле KDoc оформляются с помощью блоков, начинающихся с /\*\* и заканчивающихся \*/, и содержат описание, а также специальные теги для документации.

* 1. Какая общая форма комментария в стиле KDoc?

Ответ: /\*\*

\* Краткое описание.

\*

\* Подробное описание, если необходимо.

\*

\* @тег1 описание тега1

\* @тег2 описание тега2

\*/

* 1. Как в комментарии KDoc описать свойства класса?

Ответ: В комментарии к свойствам класса пишется краткое описание свойства

* 1. Как в комментарии KDoc описать параметры функции?

Ответ: Используя тег @param

* 1. Как в комментарии KDoc описать возвращаемое значение функции?

Ответ: Используя тег @return

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс документирования и комментирования приложения Android.

# Лабораторная работа №15 Изменение элементов дизайна

1. Цель работы
   1. Изучить процесс изменения элементов дизайна приложения в Android Studio.
2. Ответы на контрольные вопросы
   1. Какой синтаксис у применения модификаторов?

Ответ: Модификаторы применяются через цепочку вызовов с помощью оператора .

* 1. Как указать фон элемента?

Ответ: Используя модификатор background()

* 1. Как указать размер элемента?

Ответ: С помощью модификаторов size(), width(), height()

* 1. Как указать отступы элемента?

Ответ: Используя модифор padding()

* 1. Как указать настройки текста элемента?

Ответ: Через параметры компонента Text

* 1. Как добавить и использовать изображение из ресурсов?

Ответ: Используя функцию painterResource() внутри компонента Image

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс изменения элементов дизайна приложения в Android Studio.

# Лабораторная работа №16 Использование ресурсов

1. Цель работы
   1. Изучить процесс добавления и применения ресурсов при разработке мобильных приложений Android.
2. Ответы на контрольные вопросы
   1. Что хранится в папке res?

Ответ: В папке res хранятся ресурсы приложения, такие как изображения, строки, стили, макеты, цвета и другие ресурсы, используемые в проекте.

* 1. Что хранится в папке values?

Ответ: В папке values находятся файлы с ресурсами типа строк, цветов, размеров, стилей и других данных в виде XML-файлов (например, strings.xml, colors.xml, dimens.xml).

* 1. Что хранится в папке drawable?

Ответ: В папке drawable хранятся графические ресурсы — изображения различных форматов (PNG, JPEG, SVG и т.д.), используемые для отображения в интерфейсе.

* 1. Для чего используется Resource Manager?

Ответ: Resource Manager — это компонент Android, который управляет доступом к ресурсам приложения, обеспечивает их локализацию и позволяет получать ресурсы по идентификаторам в коде.

* 1. Как использовать строковые ресурсы?

Ответ: Через вызов getString(R.string.имя\_строки) или в XML через тег @string/имя\_строки.

* 1. Как использовать ресурсы изображения?

Ответ: В коде через painterResource(id = R.drawable.имя\_изображения) или в XML через android:src="@drawable/имя\_изображения".

* 1. Как использовать массивы из ресурсов?

Ответ: Создайте массив в файле arrays.xml, например:

<string-array name="my\_array">

<item>Первый</item>

<item>Второй</item>

</string-array>

Затем получайте его в коде так:

val myArray = resources.getStringArray(R.array.my\_array)

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс добавления и применения ресурсов при разработке мобильных приложений Android.

# Лабораторная работа №17 Настройка элементов управления для ввода данных

1. Цель работы
   1. Изучить процесс настройки и использования элементов управления для ввода данных приложения Android.
2. Ответы на контрольные вопросы
   1. Какие элементы позволяют ввести текстовые значения?

Ответ: Элементы, позволяющие вводить текстовые значения, включают EditText (в классическом Android) и TextField или OutlinedTextField в Jetpack Compose.

* 1. Как настроить тип клавиатуры?

Ответ: В EditText через свойство inputType.

* 1. Как связать значение из элемента управления с переменной?

Ответ: В классическом Android — через слушатель TextWatcher. В Jetpack Compose — через состояние (State) и привязку value и onValueChange.

* 1. Для чего используется combobox, какие его основные настройки?

Ответ: Combobox (или Spinner в классическом Android) используется для выбора одного варианта из списка. Основные настройки включают список элементов (Adapter), выбранное значение и обработчик выбора.

* 1. Для чего используется radiobutton, какие его основные настройки?

Ответ: RadioButton позволяет выбрать один вариант из группы. Основные настройки — текст, состояние (checked), группа (RadioGroup) для управления взаимным исключением.

* 1. Для чего используется slider, какие его основные настройки?

Ответ: Slider (или SeekBar) используется для выбора значения из диапазона. Основные параметры — минимальное и максимальное значение (valueRange), текущие значение (value), шаг (steps).

* 1. Для чего используется switch, какие его основные настройки?

Ответ: Switch — это переключатель (вкл/выкл). Основные настройки — состояние (checked) и обработчик изменения (onCheckedChange).

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс настройки и использования элементов управления для ввода данных приложения Android.

# Лабораторная работа №18 Настройка кнопок

1. Цель работы
   1. Изучить процесс настройки и использования элементов управления кнопок в приложениях Android.
2. Ответы на контрольные вопросы
   1. Как объявить button?

Ответ: В Jetpack Compose — с помощью компонента Button, например:

Button(onClick = { /\* действие \*/ }) {

Text("Нажми меня")

}

* 1. Что такое «IconButton»?

Ответ: IconButton — это кнопка, отображающая только иконку без текста, предназначенная для быстрого выполнения действия при нажатии.

* 1. Что такое «IconToggleButton»?

Ответ: IconToggleButton — это кнопка с иконкой, которая может находиться в двух состояниях (включено/выключено), позволяющая переключать состояние.

* 1. Что такое «FloatingActionButton»?

Ответ: FloatingActionButton — плавающая круглая кнопка, обычно используемая для важного действия, расположенная поверх интерфейса.

* 1. Что такое «ExtendedFloatingActionButton»?

Ответ: ExtendedFloatingActionButton — расширенная версия FAB с иконкой и текстом, занимает больше места и предоставляет больше информации о действии.

* 1. Какие параметры нужно указать при создании объекта Icon?

Ответ: Обычно указывается ресурс иконки или объект ImageVector.

1. Вывод

В ходе проделанной лабораторной работы изучил процесс настройки

# Лабораторная работа №19 Отображение списков и таблиц

1. Цель работы
   1. Изучить процесс настройки и использования элементов управления для отображения списков в приложении Android.
2. Ответы на контрольные вопросы
   1. Что такое «LazyColumn»?

Ответ: LazyColumn — это компонент в Jetpack Compose, который отображает вертикальный список элементов, загружаемых по мере прокрутки, что обеспечивает эффективное использование памяти при работе с большими наборами данных.

* 1. Что такое «LazyRow»?

Ответ: LazyRow — это компонент, отображающий горизонтальный список элементов с ленивой загрузкой, позволяющий прокручивать содержимое по горизонтали.

* 1. Что такое «LazyVerticalGrid»?

Ответ: LazyVerticalGrid — это компонент для отображения сетки элементов в вертикальной ориентации с ленивой загрузкой, позволяющий располагать элементы в несколько колонок.

* 1. Что такое «LazyHorizontalGrid»?

Ответ: LazyHorizontalGrid — это компонент для отображения сетки элементов в горизонтальной ориентации с ленивой загрузкой, позволяющий располагать элементы в несколько строк при горизонтальной прокрутке.

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс настройки и использования элементов управления для отображения списков в приложении Android.

# Лабораторная работа №20 Обработка событий: переключение между экранами

1. Цель работы
   1. Изучить процесс настройки и реализации перехода между экранами в приложении Android.
2. Ответы на контрольные вопросы
   1. Какую зависимость нужно добавить в Gradle для поддержки навигации?

Ответ: Для поддержки навигации в Jetpack Compose необходимо добавить следующую зависимость:

implementation "androidx.navigation:navigation-compose:2.5.3"

* 1. Для чего предназначен NavHost?

Ответ: NavHost — это контейнер, который управляет отображением различных экранов (фрагментов или composable) в приложении и обеспечивает навигацию между ними, основываясь на графе навигации.

* 1. Для чего предназначен NavController?

Ответ: NavController — это объект, который управляет навигацией внутри NavHost. Он позволяет программно переходить между экранами, возвращаться назад и передавать параметры.

* 1. Что указывается в графе навигации?

Ответ: В графе навигации указываются все возможные маршруты (экраны) и связи между ними, а также параметры, необходимые для перехода.

* 1. В каких случаях для организации маршрутов используется sealed class?

Ответ: Sealed class используют для определения набора маршрутов с параметрами, чтобы обеспечить типобезопасность и централизованное управление маршрутами.

* 1. В каких случаях для организации маршрутов используется enum?

Ответ: Enum применяют для фиксированного набора статичных маршрутов без параметров, когда маршруты не меняются и не требуют передачи данных.

* 1. Как считать параметры, переданные при навигации?

Ответ: Параметры извлекаются из NavBackStackEntry через arguments, например:

val param = navBackStackEntry.arguments?.getString("paramName")

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс настройки и реализации перехода между экранами в приложении Android.

# Лабораторная работа №21 Обработка событий: подсказки

1. Цель работы
   1. Изучить процесс настройки и использования элементов управления для отображения уведомлений и подсказок.
2. Ответы на контрольные вопросы
   1. Какой класс позволяет описать диалоговое окно?

Ответ: В Jetpack Compose для описания диалогового окна используется функция AlertDialog.

* 1. Какая общая форма описания диалогового окна?

Ответ: Общая форма — это вызов функции AlertDialog, в которой задаются параметры: заголовок (title), содержимое (text или textContent), кнопки (confirmButton, dismissButton) и обработчики.

* 1. Какой класс позволяет описать настройки всплывающего сообщения?

Ответ: Всплывающее сообщение (toast) настраивается через класс Toast, а в Compose — с помощью функции Snackbar.

* 1. Для чего в диалоговых окнах используется функция onDismissRequest?

Ответ: onDismissRequest — это обработчик, вызываемый при попытке закрыть диалог, например, при нажатии вне окна или на кнопку отмены; он управляет закрытием диалога.

* 1. Для чего применяется объект SnackbarHost?

Ответ: SnackbarHost — это контейнер, который отображает Snackbar (всплывающие сообщения) в приложении, управляя их отображением и очередностью.

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс настройки и использования элементов управления для отображения уведомлений и подсказок.

# Лабораторная работа №22 Обработка событий: индикация

1. Цель работы
   1. Изучить процесс изменения интерфейса для информирования пользователей в приложениях Android.
2. Ответы на контрольные вопросы
   1. Какие классы индикаторов прогресса используются в приложениях Android?

Ответ: В Jetpack Compose для отображения индикаторов прогресса используются CircularProgressIndicator и LinearProgressIndicator.

* 1. Как указать шаг в индикаторах прогресса?

Ответ: В LinearProgressIndicator можно задать параметр progress, который принимает значение от 0f до 1f, где шаг определяется изменением этого значения.

* 1. Как указать задержку в индикаторах прогресса?

Ответ: В Compose стандартные индикаторы не имеют встроенной задержки, но можно реализовать задержку с помощью LaunchedEffect и delay, чтобы управлять отображением или запуском индикатора.

* 1. Как настроить цвет в индикаторах прогресса?

Ответ: Через параметр color.

* 1. Для чего используется BadgedBox?

Ответ: BadgedBox — это контейнер, который позволяет добавлять значки (бэджи) к другим компонентам, например, к иконкам или кнопкам, отображая уведомления или статус.

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс изменения интерфейса для информирования пользователей в приложениях Android.

# Лабораторная работа №23 Настройка тем

1. Цель работы
   1. Изучить процесс изменения интерфейса для информирования пользователей в приложениях Android.
2. Ответы на контрольные вопросы
   1. Что такое MaterialTheme?

Ответ: MaterialTheme — это компонент в Jetpack Compose, который обеспечивает стилизацию интерфейса согласно принципам Material Design, включая цвета, типографику и формы, и позволяет применять их ко всему приложению или его частям.

* 1. Как настроить палитру?

Ответ: Для настройки палитры цветов создают объект Colors, задавая основные цвета (primary, secondary, background и др.), и передают его в MaterialTheme.

* 1. Как настроить типографику?

Ответ: Создают объект Typography, задавая шрифты, размеры и стили для различных текстовых элементов, и передают его в MaterialTheme.

* 1. Как сменить тему на светлую/темную?

Ответ: Можно использовать встроенные темы MaterialTheme — например, применяя isSystemInDarkTheme() для определения текущей системы темы и переключая цвета или создавая собственные темы для светлой и темной версии. Также можно определить отдельные стили и применять их по условию.

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс изменения интерфейса для информирования пользователей в приложениях Android.

# Лабораторная работа №24 Настройка анимации

1. Цель работы
   1. Изучить процесс настройки анимации интерфейса приложений Android.
2. Ответы на контрольные вопросы
   1. Для чего используется анимация в приложениях?

Ответ: Анимация в приложениях используется для улучшения пользовательского опыта, привлечения внимания, плавного перехода между состояниями интерфейса, а также для визуализации изменений и создания более живого и интерактивного дизайна.

* 1. На какие виды делится анимация в Jetpack Compose?

Ответ: В Jetpack Compose анимация делится на основные виды:

Анимации с помощью Animate\*AsState (например, animateFloatAsState, animateColorAsState) — простые анимации состояний.

Transition-анимации с UpdateTransition — для сложных последовательных или параллельных анимаций.

Бесконечные анимации с InfiniteTransition — для циклических эффектов.

* 1. Для чего используется анимация Animate\*AsState?

Ответ: Animate\*AsState используется для плавного изменения значения свойства (например, цвета, размера, прозрачности) при изменении состояния компонента, обеспечивая автоматическую интерполяцию между начальным и конечным значениями.

* 1. Для чего используется анимация UpdateTransition?

Ответ: UpdateTransition применяется для управления несколькими связанными анимациями одновременно или по очереди, позволяя создавать сложные переходы и синхронизировать изменения нескольких свойств.

* 1. Для чего используется анимация InfiniteTransition?

Ответ: InfiniteTransition предназначена для создания циклических или бесконечных анимаций, например, вращений, пульсаций или бегущих эффектов, которые повторяются без остановки.

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс настройки анимации интерфейса приложений Android.

# Лабораторная работа №25 Работа с файловой системой

1. Цель работы
   1. Изучить процесс работы с файлами в приложениях Android.
2. Ответы на контрольные вопросы
   1. Как добавить папку assets в проект?

Ответ: В Android Studio папка assets создается вручную внутри папки src/main. Для этого:

В проводнике проекта перейдите в app/src/main.

Щелкните правой кнопкой мыши, выберите New → Folder → Assets Folder.

Назовите её assets и подтвердите создание.

* 1. Как открыть файл из assets?

Ответ: Используйте AssetManager, например:

val assetManager = context.assets

val inputStream = assetManager.open("filename.txt")

* 1. Как получить путь к папке приложения?

Ответ: В Android прямого пути к внутренним папкам приложения обычно не используют, вместо этого используют API. Для внутреннего хранилища:

val filesDir = context.filesDir

* 1. Как считать данные из файла?

Ответ: Открываете файл через AssetManager или файловую систему и читаете поток данных, например:

val inputStream = context.assets.open("filename.txt")

val text = inputStream.bufferedReader().use { it.readText() }

* 1. Как записать данные в файл?

Ответ: Для записи используйте FileOutputStream, например:

val file = File(context.filesDir, "myfile.txt")

file.writeText("Ваши данные")

* 1. Как вызвать диалог выбора файла?

Ответ: В Android стандартных диалогов выбора файла нет, нужно использовать сторонние библиотеки или встроенные системы, например, через Intent.

* 1. Как вызвать диалог сохранения файла?

Ответ: Аналогично — через Intent с действием ACTION\_CREATE\_DOCUMENT.

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс работы с файлами в приложениях Android.

# Лабораторная работа №26 Подготовка стандартных модулей

1. Цель работы
   1. Изучить процесс создания стандартных модулей в приложении Android.
2. Ответы на контрольные вопросы
   1. Для чего используется модульный подход?

Ответ: Модульный подход используется для разделения приложения на независимые, легко управляемые и переиспользуемые части (модули), что повышает масштабируемость, облегчает поддержку, тестирование и повторное использование кода.

* 1. Что такое data class и какие у него особенности?

Ответ: data class — это специальный класс в Kotlin, предназначенный для хранения данных. Особенности: автоматически генерируются методы equals(), hashCode(), toString(), copy() и компоненты componentN(). Такие классы удобны для моделирования данных без необходимости писать стандартный код.

* 1. Что такое MVVM?

Ответ: MVVM (Model-View-ViewModel) — архитектурный паттерн, разделяющий приложение на три части:

Model — данные и бизнес-логика;

View — пользовательский интерфейс;

ViewModel — посредник между Model и View, управляет состоянием UI и обрабатывает логику взаимодействия.

Это способствует более чистой архитектуре, тестируемости и разделению ответственности.

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс создания стандартных модулей в приложении Android.

# Лабораторная работа №27 Передача данных между модулями

1. Цель работы
   1. Изучить процесс передачи данных между модулями в приложении Android.
2. Ответы на контрольные вопросы
   1. Как передать параметры между модулями?

Ответ: Для передачи параметров между модулями обычно используют механизмы навигации, передачу данных через аргументы или глобальные хранилища (например, ViewModel, SharedPreferences). В контексте Android и Jetpack Compose — через параметры навигационных маршрутов или NavController

* 1. Как передать значение простого типа с использованием навигации?

Ответ: Передача осуществляется через аргументы маршрута

* 1. Как передать объект с использованием навигации?

Ответ: Обычно объекты сериализуются в формат, например, JSON, и передаются как строковые аргументы. Или используют Parcelable/Serializable

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс передачи данных между модулями в приложении Android.

# Лабораторная работа №28 Работа с БД

1. Цель работы
   1. Изучить процесс работы с БД в приложении Android.
2. Ответы на контрольные вопросы
   1. Как создать БД SQLite в приложении Android?

Ответ: Для создания базы данных SQLite в Android можно использовать класс SQLiteOpenHelper, который управляет созданием и обновлением базы данных. Нужно создать подкласс SQLiteOpenHelper, переопределить методы onCreate() и onUpgrade(), и в onCreate() определить структуру таблиц с помощью SQL-запросов.

* 1. Как получить список строк таблицы БД SQLite в приложении Android?

Ответ: Открываете базу данных через SQLiteOpenHelper, вызываете getReadableDatabase(), затем выполняете запрос с помощью метода query() или rawQuery()

* 1. Как добавить новую строку в таблицу БД SQLite в приложении Android?

Ответ: Получаете объект базы данных через getWritableDatabase(), затем используете метод insert()

* 1. Как изменить существующую строку в таблице БД SQLite в приложении Android?

Ответ: Используйте метод update(

* 1. Как добавить удалить строку из таблицы БД SQLite в приложении Android?

Ответ: метод delete()

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс работы с БД в приложении Android.

# Лабораторная работа №29 Работа с API

1. Цель работы
   1. Изучить процесс работы с API в приложениях Android.
2. Ответы на контрольные вопросы
   1. Что такое REST?

Ответ: REST (Representational State Transfer) — архитектурный стиль взаимодействия в распределённых системах, основанный на использовании стандартных HTTP-методов и протокола. Он предполагает работу с ресурсами через уникальные URI и передачу данных в различных форматах, обеспечивая масштабируемость и простоту взаимодействия между клиентом и сервером

* 1. Какие методы являются стандартными для REST?

Ответ: сновные стандартные HTTP-методы в REST:

GET — получение данных;

POST — создание нового ресурса;

PUT — обновление существующего ресурса;

DELETE — удаление ресурса;

PATCH — частичное обновление ресурса.

* 1. Какой формат может быть у данных, возвращаемых при использовании REST?

Ответ: Обычно данные возвращаются в формате JSON или XML, чаще всего — JSON, так как он легче парсится и широко поддерживается.

* 1. Для чего используется библиотека retrofit2?

Ответ: Retrofit2 — это библиотека для Android и Java, которая упрощает выполнение HTTP-запросов к REST API. Она автоматически сериализует и десериализует данные, управляет сетевыми вызовами и обеспечивает удобный интерфейс для работы с API.

* 1. Как в манифесте Android разрешить использование интернет?

Ответ: В файле AndroidManifest.xml нужно добавить разрешение

<uses-permission android:name="android.permission.INTERNET" />

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс работы с API в приложениях Android.

# Лабораторная работа №30 Тестирование и оптимизация мобильного приложения

1. Цель работы
   1. Изучить процесс тестирования и оптимизации приложений Android.
2. Ответы на контрольные вопросы
   1. Что такое модульное тестирование?

Ответ: Модульное тестирование — это процесс проверки отдельных компонентов или модулей программы на правильность их работы. Цель — убедиться, что каждый модуль функционирует независимо и соответствует требованиям, что облегчает обнаружение и исправление ошибок.

* 1. В какой папке проекта хранятся модульные тесты?

Ответ: Обычно модульные тесты хранятся в папке src/test/java или src/test/kotlin внутри структуры проекта Android Studio.

* 1. Как добавить точку останова в приложение?

Ответ: В Android Studio щелкните слева от строки кода в редакторе, где хотите поставить точку останова. Она появится как красная точка, и выполнение программы при запуске в режиме отладки остановится на этой точке.

* 1. Как запустить приложение в режиме отладки?

Ответ: В Android Studio нажмите кнопку Debug (иконка жука) вместо Run или выберите Run → Debug. Также можно выбрать конфигурацию и нажать кнопку Debug для запуска приложения в режиме отладки.

1. Вывод
   1. В ходе проделанной лабораторной работы изучил процесс тестирования и оптимизации приложений Android.