云南大学软件学院 实验报告

课程：移动应用开发及安全　学期：2025 学年 秋季学期　任课教师：许红星

序号：40　学号：20231120231　姓名：张润泽　成绩：

一、 实验目的

开发标准身高计算器

开发成的目标 Android 应用的操作过程是这样的：选择用户的性别，然后输入用户的身高，点查看计算结果的按钮就在结果中显示用户的标准体重。力求操作简单，结果显示清楚。

二、实验过程

1. 需求分析

1.1 功能需求

1. **性别选择功能**：用户可以选择男性或女性
2. **身高输入功能**：用户可以输入身高值（单位：厘米）
3. **计算功能**：根据选择的性别和输入的身高计算标准体重
4. **结果显示功能**：显示计算出的标准体重结果

1.2 非功能需求

1. **易用性**：界面简洁，操作直观
2. **响应性**：计算结果即时显示
3. **兼容性**：支持 Android 7.0 及以上版本
4. **可靠性**：输入验证和错误处理

1.3 计算公式

- 男性标准体重公式：(身高 cm - 80) × 0.7
- 女性标准体重公式：(身高 cm - 70) × 0.6

2. 系统设计

2.1 架构设计

项目采用单层架构设计，由于应用功能简单，不需要复杂的多层架构。主要组件包括：

- **MainActivity**：主活动，负责 UI 展示和用户交互
- **StandardWeightCalculator**：Composable 函数，实现 UI 界面和计算逻辑

2.2 UI 设计

采用 Jetpack Compose 构建响应式 UI，界面元素包括：

- 性别选择区域（单选按钮组）
- 身高输入框（TextField）
- 计算按钮（Button）
- 结果显示区域（Text）

UI 布局采用垂直排列的 Column，所有元素居中对齐，确保界面美观和一致性。

2.3 状态管理

使用 Jetpack Compose 的状态管理机制：

- remember 函数创建可变状态
- mutableStateOf 定义响应式状态变量
- 状态变化自动触发 UI 重组

3. 实现过程

3.1 项目初始化

1. 创建新的 Android 项目
2. 配置项目基本信息（包名、版本等）
3. 设置 Kotlin 和 Compose 支持

3.2 依赖配置

在 app/build.gradle.kts 中配置必要的依赖：

```
dependencies {
    implementation(libs.androidx.core.ktx)
    implementation(libs.androidx.lifecycle.runtime.ktx)
    implementation(libs.androidx.activity.compose)
    implementation(platform(libs.androidx.compose.bom))
    implementation(libs.androidx.compose.ui)
    implementation(libs.androidx.compose.ui.graphics)
    implementation(libs.androidx.compose.ui.tooling.preview)
    implementation(libs.androidx.compose.material3)
    // 测试依赖...
}
```

3.3 UI 实现

使用 Jetpack Compose 实现用户界面：

主界面布局：

```
Column(
    modifier = Modifier.fillMaxSize().padding(16.dp),
    verticalArrangement = Arrangement.Center,
    horizontalAlignment = Alignment.CenterHorizontally
) {
    // UI 组件
}
```

性别选择组件：

```
Row {
    Row(verticalAlignment = Alignment.CenterVertically) {
        RadioButton(
            selected = gender == "Male",
            onClick = { gender = "Male" }
        )
        Text("Male", modifier = Modifier.padding(end = 16.dp))
    }
    // 女性选项...
}
```

身高输入组件：

```
TextField(
    value = height,
    onValueChange = { height = it },
    label = { Text("Enter your height (cm)") },
    singleLine = true
)
```

3.4 计算逻辑实现

在按钮点击事件中实现计算逻辑：

```
Button(onClick = {
    val h = height.toFloatOrNull()
    result = if (h != null) {
        val weight = if (gender == "Male") {
            (h - 80) * 0.7
        } else {
            (h - 70) * 0.6
        }
        "Standard weight: %.2f kg".format(weight)
    } else {
        "Please enter a valid height."
    }
}) {
    Text("View Result")
}
```

3.5 状态管理

使用 Compose 的状态管理机制：

```
var gender by remember { mutableStateOf("Male") }
var height by remember { mutableStateOf("") }
var result by remember { mutableStateOf("") }
```

4. 测试与验证

4.1 单元测试

创建基本的单元测试验证计算逻辑：

- 测试男性标准体重计算
- 测试女性标准体重计算
- 测试边界值处理

4.2 UI 测试

使用 Espresso 和 Compose 测试工具进行 UI 测试：

- 测试用户交互流程
- 验证 UI 组件显示正确性
- 测试状态变化和 UI 更新

4.3 手动测试

在不同设备和 Android 版本上进行手动测试：

- 验证应用启动和基本功能
- 测试不同输入值的处理

- 验证界面响应性和布局适应性

5. 遇到的问题与解决方案

5.1 输入验证问题

问题：用户可能输入非数字字符导致计算错误

解决方案：使用 toFloatOrNull()进行安全转换，并提供友好的错误提示

5.2 UI 布局问题

问题：在不同屏幕尺寸上布局不一致

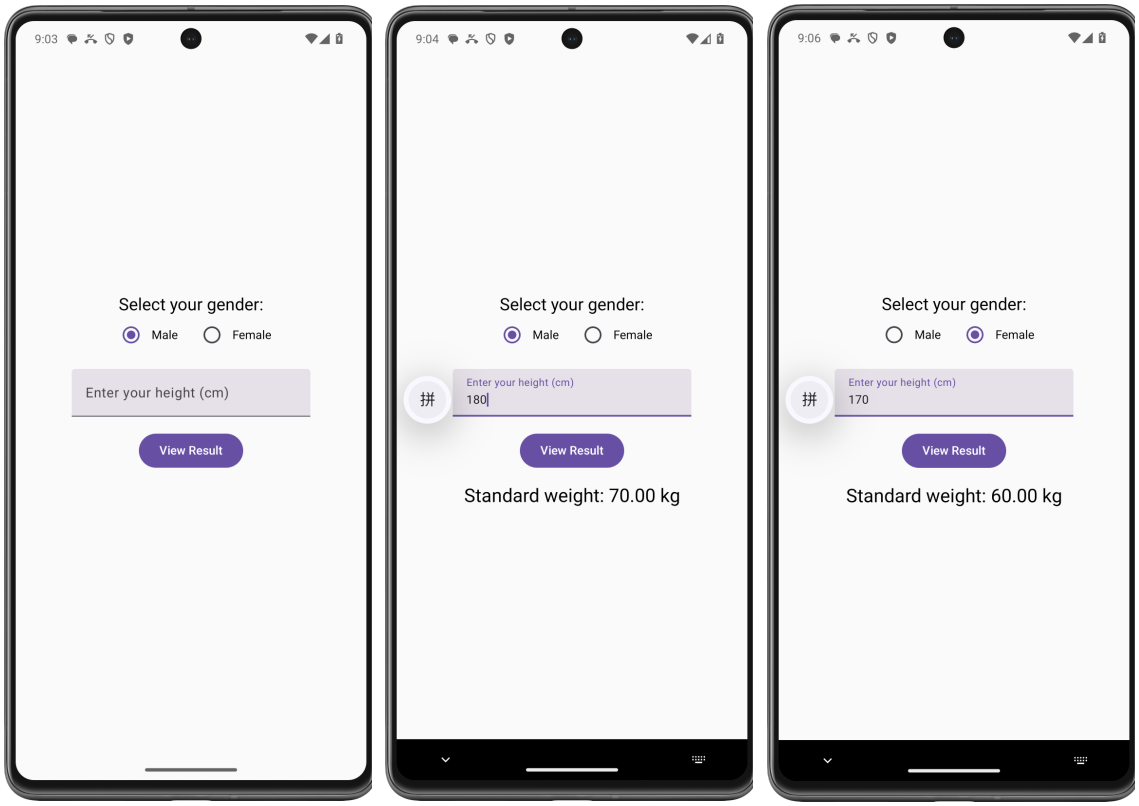
解决方案：使用 Compose 的响应式布局系统，设置适当的 padding 和 spacing

5.3 状态管理问题

问题：状态变化时 UI 不更新

解决方案：确保使用 remember 和 mutableStateOf 正确管理状态

6. 项目成果



主界面　　男性身高计算　　女性身高计算

项目完整代码：https://github.com/godsboy404/AndroidDev/tree/main/2-WeightCalc