|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

**Отчет по практической работе №1**

по дисциплине «Разработка мобильных приложений»

|  |  |
| --- | --- |
| **Выполнил:**  студент группыИКБО-28-22 | Некрасов Г.А. |
| **Проверил:**  к.э.н доцент | Степанов П.В. |

Москва 2024 г.

СОДЕРЖАНИЕ

[ПРАКТИЧЕСКОЕ ЗАДАНИЕ 3](#_Toc184744512)

[Код практического задания 4](#_Toc184744513)

[Тестирование практического задания 6](#_Toc184744514)

[Вывод 7](#_Toc184744515)

# ПРАКТИЧЕСКОЕ ЗАДАНИЕ

Используя знания о переменных, циклах, условных операторах, классах, методах и конструкторах в Kotlin создать приложение, которое помогает пользователю отслеживать его личные расходы. Программа должна отвечать следующим требованиям:

1. Наличие класса, содержащего информацию о расходах (сумма расхода, категория, дата). Класс должен содержать метод, выводящий информацию о конкретном расходе.

2. Наличие класса, содержащего информацию о списке всех расходов. Класс должен содержать метод добавления нового расхода в список, метод вывода всех расходов, а также метод подсчета суммы всех расходов по каждой категории

# Код практического задания

В ходе выполнения работы, мы написали код на языке Kotlin (см. Листинг 1)

Листинг 1. MainActivity.kt

|  |
| --- |
| package com.example.practi4eskaya\_1  import java.time.LocalDate import java.time.format.DateTimeParseException import java.util.Scanner  // Класс для хранения информации о конкретном расходе data class Expense(  val amount: Double,  val category: String,  val date: LocalDate ) {  // Метод для вывода информации о расходе  fun displayExpense() {  println("Сумма: $amount, Категория: $category, Дата: $date")  } }  // Класс для работы со списком всех расходов class ExpenseTracker {  private val expenses = mutableListOf<Expense>()   // Метод для добавления нового расхода в список  fun addExpense(amount: Double, category: String, date: LocalDate) {  val expense = Expense(amount, category, date)  expenses.add(expense)  println("Добавлен новый расход: $amount, категория: $category, дата: $date")  }   // Метод для вывода всех расходов  fun displayAllExpenses() {  if (expenses.isEmpty()) {  println("Расходы отсутствуют.")  } else {  println("Все расходы:")  expenses.forEach { it.displayExpense() }  }  }   // Метод для подсчета суммы всех расходов по каждой категории  fun calculateExpensesByCategory() {  if (expenses.isEmpty()) {  println("Расходы отсутствуют.")  return  }  val categoryTotals = expenses.groupBy { it.category }  .mapValues { entry -> entry.value.sumOf { it.amount } }   println("Сумма расходов по категориям:")  categoryTotals.forEach { (category, total) ->  println("Категория: $category, Сумма: $total")  }  } }  // Пример с динамическим меню fun main() {  val expenseTracker = ExpenseTracker()  val scanner = Scanner(System.`in`)   while (true) {  println(  """  Выберите действие:  1. Добавить новый расход  2. Показать все расходы  3. Показать сумму расходов по категориям  4. Выйти  """.trimIndent()  )  print("Введите номер действия: ")  when (scanner.nextLine().trim()) {  "1" -> {  try {  print("Введите сумму расхода: ")  val amount = scanner.nextLine().toDouble()   print("Введите категорию расхода: ")  val category = scanner.nextLine()   print("Введите дату расхода (в формате ГГГГ-ММ-ДД): ")  val date = LocalDate.parse(scanner.nextLine())   expenseTracker.addExpense(amount, category, date)  } catch (e: NumberFormatException) {  println("Ошибка: сумма должна быть числом.")  } catch (e: DateTimeParseException) {  println("Ошибка: дата должна быть в формате ГГГГ-ММ-ДД.")  } catch (e: Exception) {  println("Произошла ошибка: ${e.message}")  }  }   "2" -> {  expenseTracker.displayAllExpenses()  }   "3" -> {  expenseTracker.calculateExpensesByCategory()  }   "4" -> {  println("Выход из программы. До свидания!")  break  }   else -> {  println("Ошибка: выберите корректное действие.")  }  }  println()  } } |

# Тестирование практического задания

Выполним проверку работы нашего кода (см. Рисунки 1-

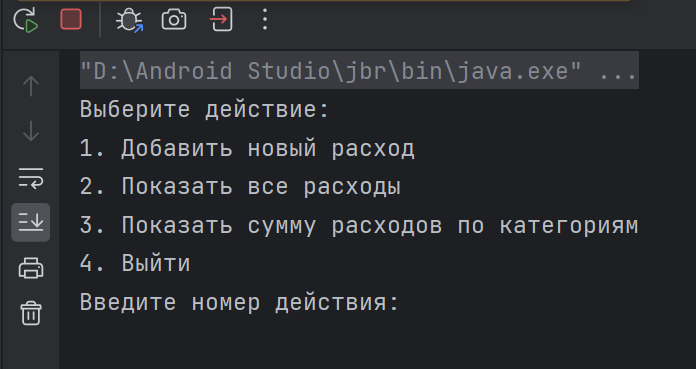


Рисунок 1. Пользовательское меню

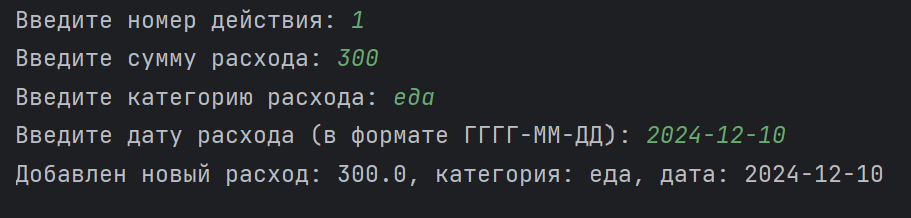


Рисунок 2. Добавление нового расхода

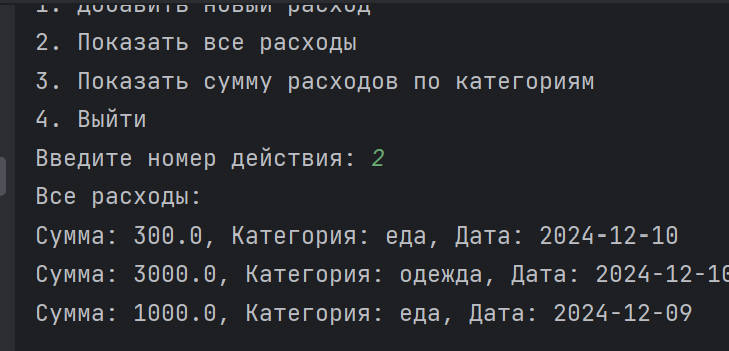


Рисунок 3. Вывод всех расходов

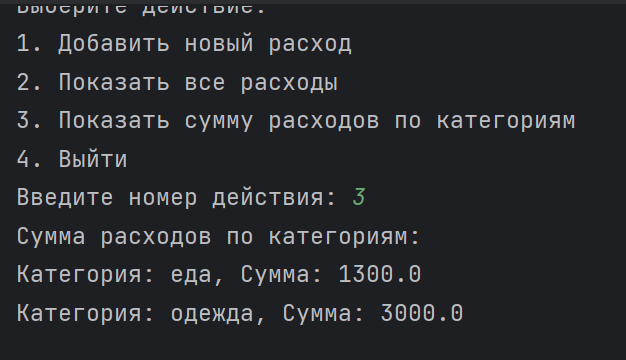


Рисунок 4. Вывод сумм расходов по категориям

# Вывод

В процессе выполнения работы мы создали приложение, которое помогает пользователю отслеживать его личные расходы