ГУАП

КАФЕДРА № 42

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | И.Д. Свеженин |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3 |
| Введение в мобильную разработку, Первое приложение в Android Studio |
| по курсу: Кроссплатформенное программирование |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4329 |  |  |  | Д.С. Шаповалова |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2025

Содержание

[1. Цель работы: 3](#_Toc211674421)

[2. Задание: 3](#_Toc211674422)

[4. Скриншоты, иллюстрирующие результаты работы программы: 5](#_Toc211674423)

[5. Вывод: 8](#_Toc211674424)

# **1. Цель работы:**

Выполнить проектирование и разработку мобильного приложения под ОС Android на языке программирования высокого уровня Kotlin.

# **2. Задание:**

Разработайте калькулятор для арифметических операций в соответствии с вариантом. Интерфейс должен быть интуитивно понятным, результат вычислений должен выводиться пользователю в Activity, все цифры и математические операции должны иметь отдельные кнопки.

**Вариант 2**

Тригонометрический калькулятор.

**3. Краткое описание хода разработки, алгоритма работы программы и назначение используемых технологий**

Ход разработки:

1. Разработать визуальную часть приложения
2. Разработать функцию вычисления тригонометрических функций
3. Устранить критические баги

**Используемые технологии:**

* Kotlin – используется для работы написанного кода.
* Jetpack Compose – визуальная часть.

**Описание структуры работы программы:**

1. Основная часть – MainActivity – точка входа
   1. Реакция приложения на событие onCreate()
      1. Вызов UI калькулятора
2. Визуальная часть – UI калькулятора
   1. Используемые переменные
   2. Вертикальный контейнер
      1. Заголовок
      2. Окно ввода
      3. Вызов функции «Цифровая клавиатура»
      4. Выбор функции (горизонтальный контейнер)
      5. Кнопка вычисления
      6. Окно результата
3. Цифровая клавиатура
   1. Визуальная часть клавиатуры
   2. Обработка нажатий
4. Логическая часть
   1. Вычисление тригонометрической функции
5. Функция демонстрация приложения в IDE.

# **4. Скриншоты, иллюстрирующие результаты работы программы:**

В качестве демонстрации работы программы приведём несколько скриншотов с разными значениями углов и выбранной функцией.



Рисунок 2.1 – Стартовый экран



Рисунок 3.1 – Синус 45 градусов



Рисунок 3.2 – Косинус 60 градусов

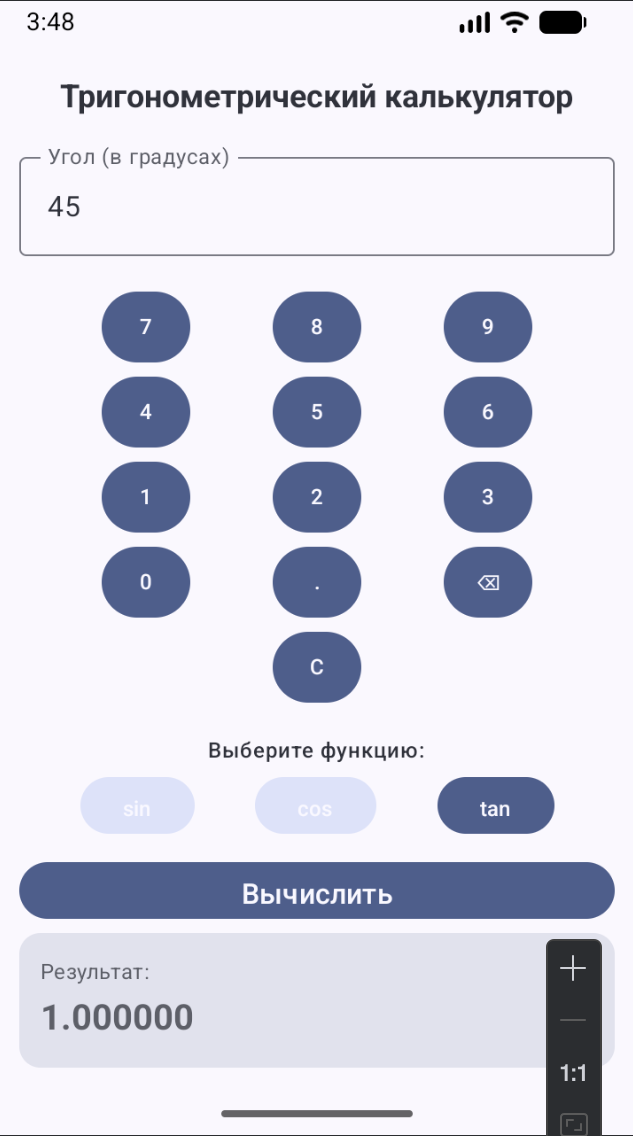


Рисунок 3.3 – Тангенс 45 градусов

Добавлена также обработка ошибок, чтобы предупредить деление на 0, при вычислении тангенса от углов, у которых остаток от деления на 180 равен +-90, то есть где косинус = 0.



Рисунок 3.4 – Попытка вычислить тангенс 90 градусов

# **5. Вывод:**

В данной работе были выполнены проектирование и разработка мобильного приложения под ОС Android на языке высокого уровня – Kotlin. Проектирование выполнено с помощью Jetpack Compose. Были созданы визуальная и логическая части приложения, вычисляющего базовые тригонометрические функции от введённого пользователем угла в градусах, с помощью цифровых кнопок, результат выводится в Activity.

ПРИЛОЖЕНИЕ А

Листинг программы:

package com.example.spasigospodi  
  
import android.os.Bundle  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.activity.enableEdgeToEdge  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.text.KeyboardOptions  
import androidx.compose.material3.\*  
import androidx.compose.runtime.\*  
import androidx.compose.ui.Alignment  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.text.font.FontWeight  
import androidx.compose.ui.text.input.KeyboardType  
import androidx.compose.ui.text.style.TextAlign  
import androidx.compose.ui.tooling.preview.Preview  
import androidx.compose.ui.unit.dp  
import androidx.compose.ui.unit.sp  
import com.example.spasigospodi.ui.theme.SpasiGospodiTheme  
import kotlin.math.cos  
import kotlin.math.sin  
import kotlin.math.tan  
  
class MainActivity : ComponentActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 *setContent* **{** SpasiGospodiTheme **{** Scaffold(modifier = Modifier.*fillMaxSize*()) **{** innerPadding **->** TrigonometryCalculator(  
 modifier = Modifier.*padding*(innerPadding)  
 )  
 **}  
 }  
 }** }  
}  
  
@Composable  
fun TrigonometryCalculator(modifier: Modifier = Modifier) {  
 var angle by remember **{** *mutableStateOf*("") **}** var selectedFunction by remember **{** *mutableStateOf*("") **}** var result by remember **{** *mutableStateOf*<Double?>(null) **}** Column(  
 modifier = modifier  
 .*fillMaxSize*()  
 .*padding*(12.*dp*),  
 verticalArrangement = Arrangement.Top,  
 horizontalAlignment = Alignment.CenterHorizontally  
 ) **{** Text(  
 text = "Тригонометрический калькулятор",  
 style = MaterialTheme.typography.headlineMedium,  
 fontWeight = FontWeight.Bold,  
 modifier = Modifier.*padding*(bottom = 8.*dp*),  
 fontSize = 18.*sp* )  
  
 // Отображение угла  
 OutlinedTextField(  
 value = angle,  
 onValueChange = **{}**,  
 label = **{** Text("Угол (в градусах)") **}**,  
 keyboardOptions = KeyboardOptions(keyboardType = KeyboardType.Decimal),  
 readOnly = true,  
 modifier = Modifier.*fillMaxWidth*()  
 )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 // Цифровая клавиатура  
 NumberPad(  
 onClick = **{** digit **->** when (digit) {  
 "C" -> angle = ""  
 "⌫" -> if (angle.*isNotEmpty*()) angle = angle.*dropLast*(1)  
 "." -> if (!angle.*contains*('.')) angle += "."  
 else -> angle += digit  
 }  
 **}** )  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 // Кнопки выбора функции  
 Text(  
 text = "Выберите функцию:",  
 style = MaterialTheme.typography.bodyLarge,  
 fontWeight = FontWeight.Medium,  
 fontSize = 12.*sp* )  
  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
  
 Row(  
 horizontalArrangement = Arrangement.SpaceEvenly,  
 modifier = Modifier.*fillMaxWidth*()  
 ) **{** *listOf*("sin", "cos", "tan").*forEach* **{** func **->** Button(  
 onClick = **{** selectedFunction = func **}**,  
 colors = ButtonDefaults.buttonColors(  
 containerColor = if (selectedFunction == func)  
 MaterialTheme.colorScheme.primary  
 else  
 MaterialTheme.colorScheme.secondaryContainer  
 ),  
 modifier = Modifier  
 .*height*(32.*dp*)  
 ) **{** Text(func, fontSize = 12.*sp*)  
 **}  
 }  
 }** Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 // Кнопка вычисления  
 Button(  
 onClick = **{** result = *calculateTrigFunction*(angle, selectedFunction)  
 **}**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(32.*dp*)  
 ) **{** Text("Вычислить", fontSize = 16.*sp*, textAlign = TextAlign.Center)  
 **}** Spacer(modifier = Modifier.*height*(8.*dp*))  
  
 // Результат  
 Card(modifier = Modifier.*fillMaxWidth*()) **{** Column(modifier = Modifier.*padding*(all = 12.*dp*)) **{** Text(  
 text = "Результат:",  
 style = MaterialTheme.typography.bodyMedium,  
 color = MaterialTheme.colorScheme.onSurfaceVariant,  
 fontSize = 12.*sp* )  
 val resultText = when {  
 angle.*isBlank*() -> "Введите угол"  
 selectedFunction.*isBlank*() -> "Выберите функцию"  
 result == null -> "..."  
 result!!.*isNaN*() -> "Ошибка: деление на ноль"  
 else -> String.*format*("%.6f", result)  
 }  
 Text(  
 text = resultText,  
 style = MaterialTheme.typography.headlineSmall,  
 fontWeight = FontWeight.Bold,  
 fontSize = 20.*sp* )  
 **}  
 }  
 }**}  
  
@Composable  
fun NumberPad(onClick: (String) -> Unit) {  
 val rows = *listOf*(  
 *listOf*("7", "8", "9"),  
 *listOf*("4", "5", "6"),  
 *listOf*("1", "2", "3"),  
 *listOf*("0", ".", "⌫"),  
 *listOf*("C")  
 )  
  
 Column(horizontalAlignment = Alignment.CenterHorizontally) **{** rows.*forEach* **{** row **->** Row(  
 horizontalArrangement = Arrangement.SpaceEvenly,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(vertical = 4.*dp*)  
 ) **{** row.*forEach* **{** symbol **->** Button(  
 onClick = **{** onClick(symbol) **}**,  
 modifier = Modifier  
 .*width*(50.*dp*)  
 .*height*(40.*dp*)  
 ) **{** Text(symbol,  
 fontSize = 12.*sp*,  
 textAlign = TextAlign.Center)  
 **}  
 }  
 }  
 }  
 }**}  
  
private fun calculateTrigFunction(angle: String, function: String): Double? {  
 return try {  
 val angleInDegrees = angle.*toDouble*()  
  
 when (function) {  
 "sin" -> *sin*(Math.toRadians(angleInDegrees))  
 "cos" -> *cos*(Math.toRadians(angleInDegrees))  
 "tan" -> {  
 // Если угол ≡ 90° (mod 180°), тангенс не определён  
 val mod180 = angleInDegrees % 180  
 if (mod180 == 90.0 || mod180 == -90.0) {  
 Double.NaN  
 } else {  
 *tan*(Math.toRadians(angleInDegrees))  
 }  
 }  
 else -> Double.NaN  
 }  
 } catch (e: NumberFormatException) {  
 null  
 }  
}  
  
  
@Preview(showBackground = true)  
@Composable  
fun TrigonometryCalculatorPreview() {  
 SpasiGospodiTheme **{** TrigonometryCalculator()  
 **}**}