ГУАП

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ _____
ПРЕПОДАВАТЕЛЬ

Ассистент		Н.А. Янковский
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №1

Исследование аналоговых сигналов

Вариант 5

по курсу: Цифровая обработка и передача сигналов

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. № 4128		В.А. Воробьев
	подпись, дата	инициалы, фамилия

Санкт-Петербург 2023

1 Задание

Исходные данные:

$f_1 = 2.7N, f_2 = 1.5N, T = N$, где N – номер по списку.

Проанализировать свойства двух функций $u_1(t) = \sin(2\pi f_1 t)$ и $u_2(t) = \sin(2\pi f_2 t)$ в интервале $[-T/2, T/2]$.

Написать программу, которая позволит:

1. Вычислить все значения функций $u_1(t)$ и $u_2(t)$ на заданном интервале с шагом 10^{-3} и построить график полученных функций.

2. Вычислить приближенное значение скалярного произведения двух функций $(u_1(t), u_2(t))$.

3. Вычислить нормы обеих функций.

4. Определить, являются ли исходные функции ортогональными друг к другу.

5. Как нужно изменить исходные функции, что они могли являться элементами ортонормированного базиса? Выполните данную модификацию и продемонстрируйте результат.

6. Останутся ли исследуемые функции элементами ортонормированного базиса, если:

6.1 частоты f_1 и f_2 удвоятся;

6.2 интервал $[-T/2; T/2]$ увеличится вдвое;

6.3 интервал $[-T/2; T/2]$ уменьшится вдвое.

2 Выполнение работы

Построение графика функций u1 и u2.

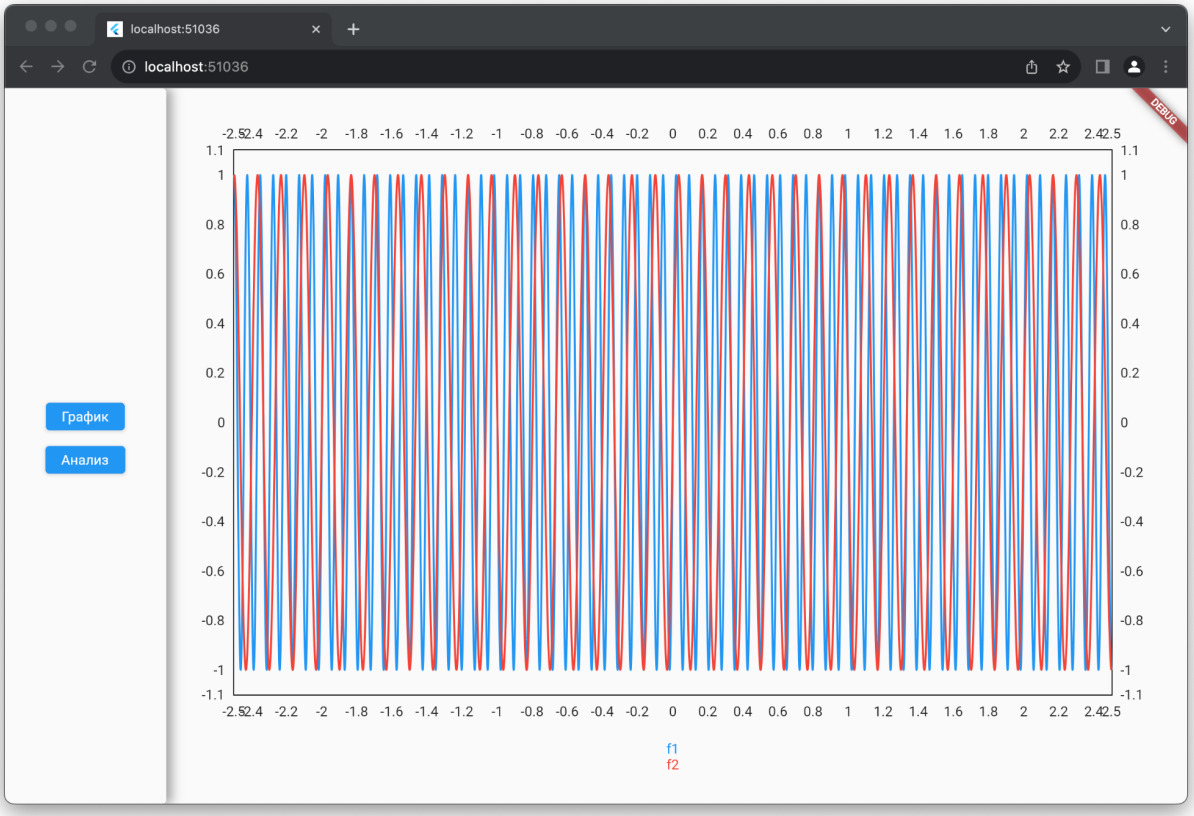

Рисунок 1 - Графики функций u1 и u2

Для улучшения визуализации был увеличен шаг на заданном интервале.

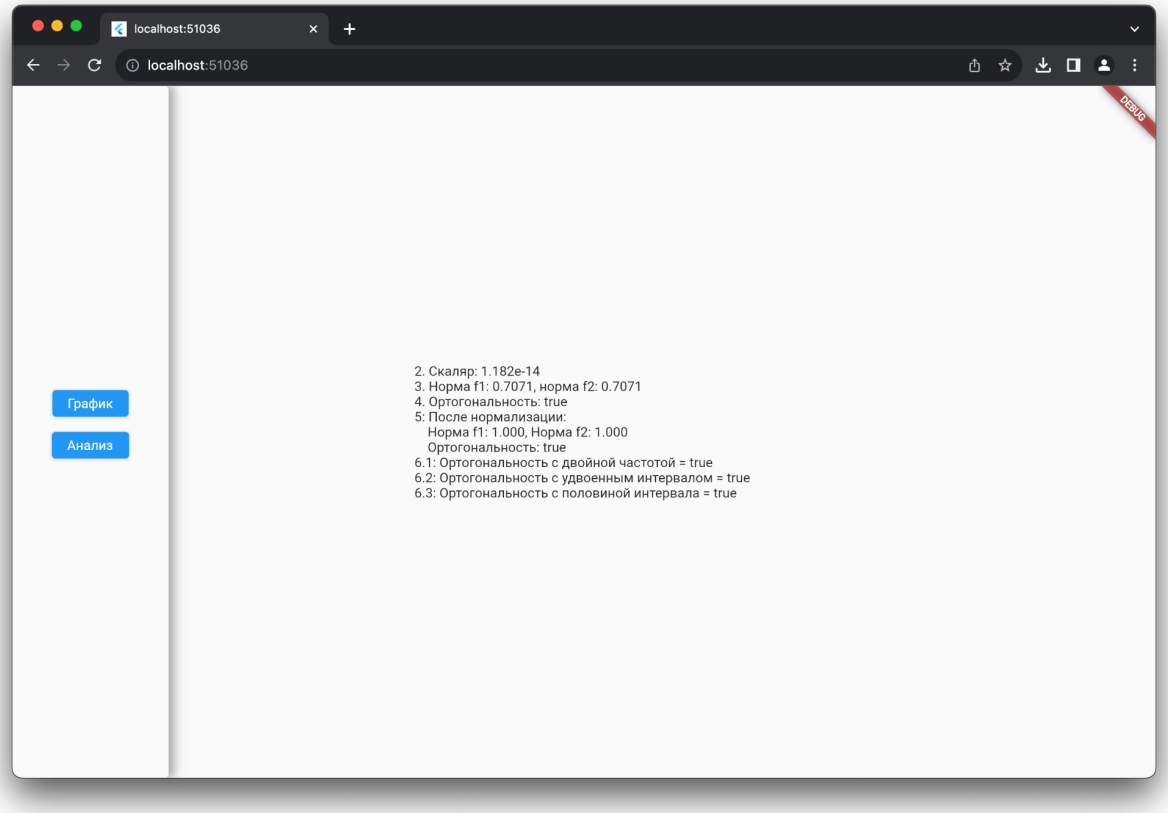

Рисунок 2 - Результат выполнения программы

3 Вывод

В ходе выполнения лабораторной работы мы приобрели практические навыки вычисления и визуализации математических функций, эти навыки в дальнейшем могут быть полезны в анализе данных, обработке сигналов, а также в других областях, где важно понимание и работа с математическими функциями.

Листинг программы

side_bar.dart

```
import 'package:flutter/material.dart';

class SideBar extends StatelessWidget {
  static const _buttonsPadding = 16.0;

  final List<SideBarElement> elements;

  const SideBar({Key? key, required this.elements}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Container(
      decoration: BoxDecoration(
        color: const Color.fromARGB(255, 250, 250, 250),
        borderRadius: BorderRadius.circular(4),
        boxShadow: [
          BoxShadow(
            color: Colors.black.withAlpha(85),
            blurRadius: 8,
            offset: const Offset(6, 0),
          )
```

```
        ],
      ),
      padding: const EdgeInsets.all(8),
      alignment: Alignment.center,
      child: Column(
        mainAxisSize: MainAxisSize.min,
        children: elements
            .map(
              (e) => Column(
                children: [
                  ElevatedButton(
                    onPressed: e.onTap,
                    child: Text(e.title),
                  ),
                  const SizedBox(height: _buttonsPadding),
                ],
              ),
            )
            .toList(),
      ),
    );
  }
}
```

```
class SideBarElement {
  final String title;
  final VoidCallback onTap;

  SideBarElement(this.title, this.onTap);
}
```

preview_page.dart

```
import 'package:flutter/material.dart';

import 'components/side_bar.dart';

class PreviewPage extends StatelessWidget {
  static const _sideBarWidth = 156.0;
  static const _sideBarFloat = 8.0;

  final GlobalKey<NavigatorState> navigatorKey;
  final Widget initialPage;
  final List<SideBarElement> sideBarElements;

  const PreviewPage({
    Key? key,
    required this.navigatorKey,
    required this.initialPage,
```

```
  required this.sideBarElements,
}) : super(key: key);

@override
Widget build(BuildContext context) {
  return Stack(
    children: [
      Positioned(
        left: _sideBarWidth,
        top: 0,
        bottom: 0,
        right: 0,
        child: Navigator(
          key: navigatorKey,
          onGenerateRoute: (settings) => MaterialPageRoute(
            builder: (_) => initialPage,
          ),
        ),
      ),
      Positioned(
        left: 0,
        top: 0,
        bottom: 0,
        width: _sideBarWidth + _sideBarFloat,
```

```
          child: SideBar(
            elements: sideBarElements,
          ),
        )
      ],
    );
  }
}
```

first_info_page.dart

```
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:task/solutions/first/first_providers.dart';

class FirstInfoPage extends ConsumerWidget {
  const FirstInfoPage({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context, WidgetRef ref) {
    final manager = ref.watch(FirstProviders.managerProvider);

    return Center(
      child: SizedBox(
        width: 512,
```

```
child: Column(
  mainAxisSize: MainAxisSize.min,
  crossAxisAlignment: CrossAxisAlignment.stretch,
  children: [
    Text("2. Скаляр: ${manager.scalar.strRound}"),
    Text(
      "3. Норма f1: ${manager.function1Norm.strRound}, норма f2: ${manager.function2Norm.strRound}",
    ),
    Text("4. Ортогональность: ${manager.isOrthogonal}"),
    Text("5: После нормализации:\n"
        "  Норма f1: ${manager.function1NormalizeNorm.strRound}, Норма f2: ${manager.function1NormalizeNorm.strRound}\n"
        "  Ортогональность: ${manager.isNormalizedOrthogonal}"),
    Text(
      "6.1: Ортогональность с двойной частотой = ${manager.isOrthogonalWithDoubleHz}",
    ),
    Text(
      "6.2: Ортогональность с удвоенным интервалом = ${manager.isOrthogonalWithBigInterval}",
    ),
    Text(
      "6.3: Ортогональность с половиной интервала = ${manager.isOrthogonalWithShortInterval}",
```

```
            ),
          ],
        ),
      ),
    );
  }
}

extension _DoubleExt on double {
  String get strRound => toStringAsPrecision(4);
}
```

first_chart_page.dart

```
import 'package:fl_chart/fl_chart.dart';
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
import 'package:task/solutions/first/first_providers.dart';

class FirstChartPage extends ConsumerWidget {
  const FirstChartPage({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context, WidgetRef ref) {
    final lineCharData = LineChartBarData(
```

```
  dotData: const FlDotData(show: false),
  spots: [],
  isCurved: true,
);
final manager = ref.watch(FirstProviders.managerProvider);

return Center(
  child: Padding(
    padding: const EdgeInsets.all(32),
    child: Column(
      children: [
        Expanded(
          child: LineChart(
            LineChartData(
              maxY: 1.1,
              minY: -1.1,
              gridData: const FlGridData(show: false),
              lineBarsData: [
                lineCharData.copyWith(
                  color: Colors.blue,
                  spots: manager.function1Dots,
                  aboveBarData: BarAreaData(),
                ),
                lineCharData.copyWith(
```

```
                color: Colors.red,
                spots: manager.function2Dots,
              )
            ],
          ),
        ),
      ),
      const SizedBox(height: 16),
      const Text(
        "f1",
        style: TextStyle(color: Colors.blue),
      ),
      const Text(
        "f2",
        style: TextStyle(color: Colors.red),
      ),
    ],
  ),
 ),
);
 }
}
```

first_calculation_manager.dart

```
import 'package:fl_chart/fl_chart.dart';
import 'package:my_math/my_math.dart';

class FirstCalculationManager {
  final double step;
  final MathInterval interval;
  final Func function1;
  final Func function2;

  FirstCalculationManager({
    required this.step,
    required this.interval,
    required this.function1,
    required this.function2,
  });

  List<FlSpot> get function1Dots {
    return interval.stepped(
      step: step,
      map: (x) => FlSpot(x, function1(x)),
    );
  }

  List<FlSpot> get function2Dots {
```

```
  return interval.stepped(
    step: step,
    map: (x) => FlSpot(x, function2(x)),
  );
}

double get scalar => Algebra.scalarProduct(function1, function2, interval);

double get function1Norm => function1.integral(interval: interval).norm;

double get function2Norm => function2.integral(interval: interval).norm;

Func get _func1Normalize => (x) => function1(x) / function1Norm;

Func get _func2Normalize => (x) => function2(x) / function2Norm;

double get function1NormalizeNorm {
  return _func1Normalize.integral(interval: interval).norm;
}

double get function2NormalizeNorm {
  return _func2Normalize.integral(interval: interval).norm;
}
```

```
bool get isNormalizedOrthogonal => Algebra.isOrthogonal(
      _func1Normalize,
      _func2Normalize,
      interval,
    );

bool get isOrthogonal => Algebra.isOrthogonal(
      function1,
      function2,
      interval,
    );

bool get isOrthogonalWithDoubleHz => Algebra.isOrthogonal(
      (x) => function1(x * 2),
      (x) => function2(x * 2),
      interval,
    );

bool get isOrthogonalWithShortInterval => Algebra.isOrthogonal(
      function1,
      function2,
      interval * 0.5,
    );
```

```
  bool get isOrthogonalWithBigInterval => Algebra.isOrthogonal(
        function1,
        function2,
        interval * 2,
      );
}
```

first_page_solution.dart

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:task/solutions/first/ui/pages/first_chart_page.dart';
import 'package:task/solutions/first/ui/pages/first_info_page.dart';
import 'package:task/ui/components/side_bar.dart';
import 'package:task/ui/preview_page.dart';

class FirstPageSolution extends StatelessWidget {
  final GlobalKey<NavigatorState> _navigatorKey = GlobalKey();

  FirstPageSolution({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return PreviewPage(
      navigatorKey: _navigatorKey,
```

```
  initialPage: const FirstChartPage(),
  sideBarElements: [
    SideBarElement(
      "График",
      () => _navigatorKey.currentState?.pushReplacement(
        PageRouteBuilder(
          pageBuilder: (_, __, ___) => const FirstChartPage(),
          transitionDuration: const Duration(seconds: 0),
          transitionsBuilder: (_, __, ___, child) => child,
        ),
      ),
    ),
    SideBarElement(
      "Анализ",
      () => _navigatorKey.currentState?.pushReplacement(
        PageRouteBuilder(
          pageBuilder: (_, __, ___) => const FirstInfoPage(),
          transitionDuration: const Duration(seconds: 0),
          transitionsBuilder: (_, __, ___, c) => c,
        ),
      ),
    ),
  ],
);
```

```
  }
}
```

first_providers.dart

```
import 'dart:math';

import 'package:my_math/my_math.dart';
import 'package:riverpod/riverpod.dart';
import 'package:task/solutions/first/logic/first_calculation_manager.dart';

abstract final class FirstProviders {
  // Variant providers
  static final _variantProvider = Provider<int>((ref) => 5);

  static final _mult1 = Provider<double>((ref) => 2.7);

  static final _mult2 = Provider<double>((ref) => 1.5);

  static final _stepProvider = Provider<double>((ref) => 0.001);

  static final _funcProvider = Provider.family<Func, double>((ref, mult) {
    final n = ref.watch(_variantProvider);
    return (x) => sin(2 * pi * n * mult * x);
  });
```

```
  static final _intervalProvider = Provider<MathInterval>((ref) {
    final variant = ref.watch(_variantProvider);
    return MathInterval(-variant / 2, variant / 2);
  });

  // Logic providers

  static final managerProvider = Provider<FirstCalculationManager>((ref) {
    final f1 = ref.watch(_mult1);
    final f2 = ref.watch(_mult2);

    return FirstCalculationManager(
      step: ref.watch(_stepProvider),
      interval: ref.watch(_intervalProvider),
      function1: ref.watch(_funcProvider(f1)),
      function2: ref.watch(_funcProvider(f2)),
    );
  });
}
```

main.dart

```
import 'package:flutter/material.dart';
import 'package:flutter_riverpod/flutter_riverpod.dart';
```

```
import 'package:task/solutions/first/first_page_solution.dart';

void main() {
  runApp(const MainApp());
}

class MainApp extends StatelessWidget {
  const MainApp({super.key});

  @override
  Widget build(BuildContext context) {
    return ProviderScope(
      child: MaterialApp(
        home: Scaffold(
          body: FirstPageSolution(),
        ),
      ),
    );
  }
}
```

my_math.dart

```
library my_math;
```

```
export 'src/utils/typedefs.dart';
export 'src/extensions/interval_extension.dart';
export 'src/extensions/integral_extension.dart';
export 'src/math/algebra.dart';
export 'src/models/math_interval.dart';
```

typedefs.dart

```
typedef Func = double Function(double x);
```

const.dart

```
abstract final class Const {
  static const int integralN = 100;
  static const epsilon = 0.001;
}
```

definite_integral.dart

```
import 'dart:math';

import 'package:my_math/src/utils/const.dart';

import '../../my_math.dart';

class DefiniteIntegral {
```

```
  final MathInterval interval;
  final Func f;

  DefiniteIntegral(this.f, {required this.interval});

  double get norm {
    final start = interval.start;
    final end = interval.end;

    int n = Const.integralN;
    double h = (end - start) / n;

    double result = 0.0;

    for (int i = 0; i < n; i++) {
      double x = start + i * h;
      result += pow(f(x), 2);
    }

    result *= h * (1 / (end - start));
    return sqrt(result);
  }
}
```

math_interval.dart

```
class MathInterval {
  final double start;
  final double end;

  MathInterval(this.start, this.end);

  MathInterval operator *(double multiplier) {
    return MathInterval(start * multiplier, end * multiplier);
  }
}
```

algebra.dart

```
import 'package:my_math/src/utils/const.dart';

import '../models/math_interval.dart';
import '../utils/typedefs.dart';

abstract final class Algebra {
  static double scalarProduct(Func f1, Func f2, MathInterval interval) {
    final start = interval.start;
    final end = interval.end;

    int n = Const.integralN;
```

```
    double h = (end - start) / n;

    double result = 0.0;

    for (int i = 0; i < n; i++) {
      double x = start + i * h;
      result += f1(x) * f2(x);
    }

    return result * h;
  }

  static bool isOrthogonal(Func f1, Func f2, MathInterval interval) =>
      areSimilar(scalarProduct(f1, f2, interval), 0);

  static bool areSimilar(double a, double b) {
    return (a - b).abs() < Const.epsilon;
  }
}
```

integral_extension.dart

```
import 'package:my_math/my_math.dart';
import 'package:my_math/src/models/definite_integral.dart';
```

```
extension FuncExtensionExt on Func {
  DefiniteIntegral integral({required MathInterval interval}) {
    return DefiniteIntegral(this, interval: interval);
  }
}
```

interval_extension.dart

```
import '../models/math_interval.dart';

extension IntervalExt on MathInterval {
  double get length => end - start;

  List<T> stepped<T>({required final step, required Function(double x) map}) {
    return [for (var x = start; x < end; x += step) map(x)];
  }
}
```