|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатики и систем управления

КАФЕДРА Теоретической информатики и компьютерных технологий

**Контрольная работа № 1**

по курсу «Разработка мобильных приложений»

Выполнил:

студент группы ИУ9-71Б

Митрошкин Алексей

Проверил:

Посевин Д.П.

Москва, 2024

1. **Цель**

Целью контрольной работы является создание интерактивного приложения для визуализации радиус-векторов тел с заданными координатами и массами. В рамках работы студенты научатся:

1. **Задание**

|  |  |
| --- | --- |
| **Входные данные** | **Результат** |
| **Массы и координаты центра масс пяти тел в двухмерном пространстве.** | **Радиус-векторы точек отрисовать зелёным.**  **Радиус-вектор координаты центра масс системы – красным.** |

1. **Реализация**

Листинг 1. Мобильное приложение

|  |
| --- |
| import 'package:flutter/material.dart';  import 'dart:math';  void main() => runApp(MyApp());  class MyApp extends StatelessWidget {  @override  Widget build(BuildContext context) {  return MaterialApp(  home: VectorPainterScreen(),  );  }  }  class VectorPainterScreen extends StatefulWidget {  @override  \_VectorPainterScreenState createState() => \_VectorPainterScreenState();  }  class \_VectorPainterScreenState extends State<VectorPainterScreen> {  // Задаем начальные значения масс и координат  List<double> masses = [1.0, 2.0, 3.0, 4.0, 5.0];  List<Offset> coordinates = [  Offset(50, 100),  Offset(-70, 150),  Offset(120, -50),  Offset(-90, -80),  Offset(30, 60),  ];  int repaintCounter = 0; // Счётчик для обновления CustomPaint  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: Text('Радиус-векторы тел'),  ),  body: Column(  children: [  Expanded(  child: Padding(  padding: const EdgeInsets.all(8.0),  child: ListView.builder(  itemCount: 5,  itemBuilder: (context, index) {  return VectorInput(  index: index,  mass: masses[index],  coordinate: coordinates[index],  onMassChanged: (value) {  setState(() {  masses[index] = value;  });  },  onCoordinateChanged: (offset) {  setState(() {  coordinates[index] = offset;  });  },  );  },  ),  ),  ),  ElevatedButton(  onPressed: () {  setState(() {  repaintCounter++; // Увеличение счётчика для перерисовки CustomPaint  });  },  child: Text('Нарисовать векторы'),  ),  Expanded(  child: CustomPaint(  size: Size.infinite,  painter: VectorPainter(coordinates, repaintCounter), // Передача счётчика  ),  ),  ],  ),  );  }  }  class VectorInput extends StatelessWidget {  final int index;  final double mass;  final Offset coordinate;  final Function(double) onMassChanged;  final Function(Offset) onCoordinateChanged;  VectorInput({  required this.index,  required this.mass,  required this.coordinate,  required this.onMassChanged,  required this.onCoordinateChanged,  });  @override  Widget build(BuildContext context) {  return Row(  children: [  Expanded(  child: TextFormField(  initialValue: mass.toString(),  decoration: InputDecoration(labelText: 'Масса ${index + 1}'),  keyboardType: TextInputType.number,  onChanged: (value) => onMassChanged(double.tryParse(value) ?? 1.0),  ),  ),  SizedBox(width: 8),  Expanded(  child: TextFormField(  initialValue: coordinate.dx.toString(),  decoration: InputDecoration(labelText: 'X${index + 1}'),  keyboardType: TextInputType.number,  onChanged: (value) {  double x = double.tryParse(value) ?? 0;  onCoordinateChanged(Offset(x, coordinate.dy));  },  ),  ),  SizedBox(width: 8),  Expanded(  child: TextFormField(  initialValue: coordinate.dy.toString(),  decoration: InputDecoration(labelText: 'Y${index + 1}'),  keyboardType: TextInputType.number,  onChanged: (value) {  double y = double.tryParse(value) ?? 0;  onCoordinateChanged(Offset(coordinate.dx, y));  },  ),  ),  ],  );  }  }  class VectorPainter extends CustomPainter {  final List<Offset> coordinates;  final int repaintCounter; // Добавлен счётчик для обновления  VectorPainter(this.coordinates, this.repaintCounter);  @override  void paint(Canvas canvas, Size size) {  final origin = Offset(size.width / 2, size.height / 2);  final greenPaint = Paint()  ..color = Colors.green  ..strokeWidth = 2;  final redPaint = Paint()  ..color = Colors.red  ..strokeWidth = 2;  // Нарисовать зеленые радиус-векторы  for (var point in coordinates) {  canvas.drawLine(origin, origin + point, greenPaint);  }  // Вычислить центр масс  double totalMass = coordinates.length.toDouble();  Offset centerOfMass = coordinates.fold(Offset.zero, (sum, offset) => sum + offset) / totalMass;  // Нарисовать красный радиус-вектор центра масс  canvas.drawLine(origin, origin + centerOfMass, redPaint);  }  @override  bool shouldRepaint(covariant CustomPainter oldDelegate) {  return true; // Обновление при любом изменении  }  } |

1. **Результаты**

Результат показан на рисунке 1.

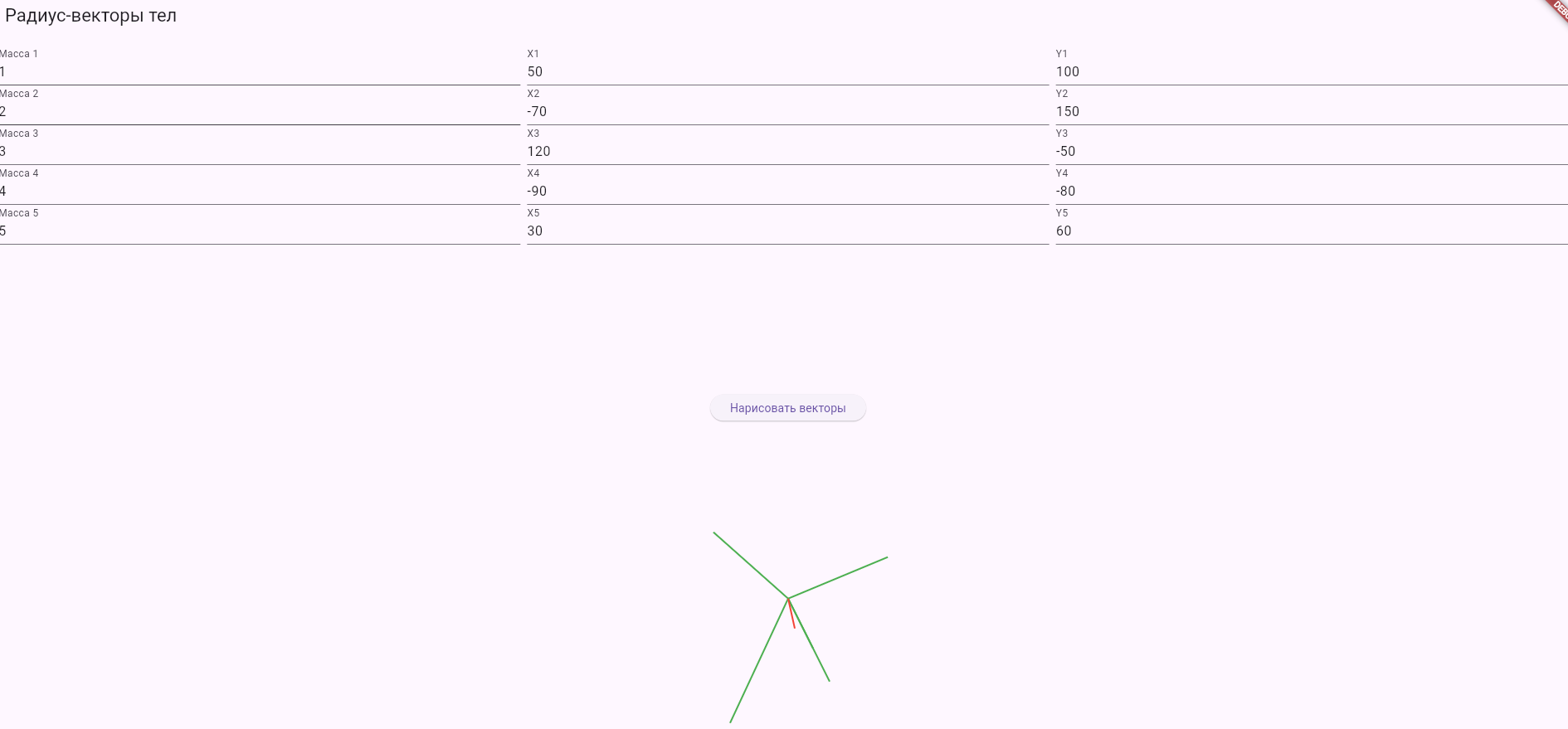


Рисунок 1 - Результаты работы

1. **Вывод**

В результате выполнения данной лабораторной работы было создано интерактивное приложение с использованием Flutter, которое позволяет визуализировать радиус-векторы тел и центра масс на плоскости. Пользователь может задавать координаты и массы объектов через интерфейс, после чего приложение рассчитывает и отображает радиус-векторы для каждого тела и центр масс всех тел.