ФЕДЕРАЛЬНОЕ АГЕНТСТВО МОРСКОГО И РЕЧНОГО ТРАНСПОРТА

ФЕДЕРАЛЬНОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«Морской государственный университет имени адмирала Г.И. Невельского»

Факультет электроники и информационных технологий

Кафедра Вычислительной техники

Дисциплина «Программирование»

**Курсовая работа**

«Векторный редактор. Объектно-ориентированное программирование.»

Выполнил: студент гр. *214.21*

*Гармышев Дмитрий Евгеньевич*

Проверил: преподаватель

Хованец В.А.

Оглавление

[Введение 3](#_Toc73003707)

[Задание 4](#_Toc73003708)

[План разработки проекта 5](#_Toc73003709)

[Набросок тестового проекта 5](#_Toc73003710)

[Разработка новой структуры данных 5](#_Toc73003711)

[Хранение точек в структуре, основанной на «мапах» (Map) 6](#_Toc73003712)

[Общее описание проекта 7](#_Toc73003713)

[Требования к оформлению кода и выбору имен 8](#_Toc73003714)

[Алгоритмы 9](#_Toc73003715)

[Тексты модулей 10](#_Toc73003716)

[Инструкция к приложению 11](#_Toc73003717)

[Заключение 12](#_Toc73003718)

[Список литературы 13](#_Toc73003719)

# Введение

Основная тема данной работы – векторный редактор, который позволяет быстро и удобно создавать изображения, основанные на геометрических фигурах. Его особенность заключается в том, что мы можем легко создавать и редактировать фигуры.

Для создания этого проекта было использовано Объектно-ориентированное программирование (далее ООП). Такой подход позволяет явно представлять объекты (например, объект точки или фигуры), указывать их свойства и функции. Помимо ООП, уделено внимание алгоритмам поиска фигур, точек и их обработки.

Проект написан для ОС Android, на языке Java, с использованием событийно-управляемого программирования (например, нажатие на экран, кнопки и т.д.).

Используемые инструменты включают в себя IDE Android Studio для написания кода, и браузер для поиска полезной информации в процессе разработки ПО.

Основная цель данного проекта – закрепить знания в области ООП, научиться создавать сложные структуры данных и алгоритмы для их обработки.

Структура данных включает в себя две части. Первая часть включает в себя все точки, которые отображаются на экране, и хранит только их координаты. Вторая часть хранит в себе все данные о фигурах, их шаблонах. Фигуры в данной структуре не содержат в себе объекты точек, а только ссылки на них в виде индексов.

# Задание

1. Создать новую фигуру щелчками мышью – суть этого задания заключается в том, чтобы при нажатии на экран, создавались точки и соединялись линиями, тем самым образуя фигуру.
2. Выбрать нужные фигуры или одну фигуру щелчками мышью на фигуре. По нажатию на экран, и по полученным координатам нажатия мы должны правильно распознать, какую фигуру мы выбрали.
3. Удалить выбранные фигуры. Данное задание вытекает из второго. Просто удаляем выбранные фигуры из списка, а также точки, которые принадлежат этой фигуре.
4. Копировать выбранные и поместить в нужное место. Данное задание вытекает из второго. Мы создаём копии выбранных фигур и их точек, но уже смещаем их в другое место, куда нажал пользователь.
5. Переместить выбранные фигуры. Данное задание вытекает из второго. Мы просто перемещаем выбранные фигуры, туда куда нажал пользователь.

# План разработки проекта

## Набросок тестового проекта

Для того чтобы приблизительно понять, что конкретно требуется разработать в этом проекте, было создано тестовое приложение (черновик), в котором использовались самые примитивные структуры данных (довольно неоптимизированные, и не пригодные для дальнейшего развития проекта), не самые лучшие алгоритмы, но зато примерно было понятно куда двигаться дальше. Из функций, это приложение включало в себя создание одной фигуры, редактирование её точек (простое перемещение и удаление их), и элементы событийно-управляемого программирования.

Пускай на это приложение было потрачено время, зато я смог понять с какими «подводными камнями» я могу столкнуться в процессе разработки основного приложения, смог определить для себя самые важные функции приложения, и продумал дальнейшую архитектуру (не структур данных, а моментов касаемо работы самого приложения), тем самым значительно снизив вероятность глобальных переработок в дальнейшем.

## Разработка новой структуры данных

Данный этап является тоже самым немаловажным. Изначально было очень много вариантов создания структуры данных, основанных на разных принципах. Где-то все точки хранились в «мапах» (структура данных Map – ключ-значение), где-то использовалось хранение целых объектов в структурах. У каждой структуры были свои плюсы и минусы. Ниже я опишу варианты структур данных для хранения точек отдельно, и для хранения фигур, разберу их преимущества и недостатки, принципы их работы.

Первым делом, я отдельно начал уделять внимание именно этому моменту, т.к. у нас может быть очень большое количество точек, и нам важен быстрый поиск необходимой точки.

### Хранение точек в структуре, основанной на «мапах» (Map)

Для начала следует разобраться, что из себя представляем Map в Java.

Интерфейс Map<K, V> представляет отображение или иначе говоря словарь, где каждый элемент представляет пару "ключ-значение". При этом все ключи уникальные в рамках объекта Map. Такие коллекции облегчают поиск элемента, если нам известен ключ - уникальный идентификатор объекта.

То есть, данная структура уже подходит по критерию быстрого поиска элементов в ней.

# Общее описание проекта

1. **Структура модулей**

В данном разделе должны быть перечислены и описаны разрабатываемые модули.

1. **Типы данных и имена переменных**

Указываем основные структуры данных и особенности их использования, если они есть. Описываем основные переменные и обязательно глобальные переменные с объяснением того, почему они сделаны глобальными.

1. **Структура меню проекта**

Показываем меню проекта в виде образов экрана. Даем краткое описание пунктов меню и действий в этих пунктах.

1. **Описания основных процедур**

Описываем самостоятельно разработанные в проекте основные процедуры и реализуемые ими алгоритмы.

1. **Примеры графических построений в приложении**

Показываем характерные примеры графических построений в виде образов экранов.

# Требования к оформлению кода и выбору имен

Очень важный раздел. Здесь должны быть приведены основные требования к оформлению кода, способам формирования имен и всё достаточно подробно. Ищите то, что я давал в лекциях и что написано в методических указаниях, ищите информацию в Интернет, чем больше вы напишете, тем лучше. Напишите какие префиксы в именах вы используете, укажите в какой нотации записываются ваши имена в проекте. Также можете объяснить, почему соглашения об оформлении кода и выборе имен так важны.

# Алгоритмы

Здесь размещаются словесные описания алгоритмов для каждого из ваших индивидуальных заданий. Как показывает опыт, написание алгоритмов для вас немалая проблема, поэтому, тренируйтесь, пробуйте, пишите. Обычно пытаются сразу начинать кодирование, а в голове нет даже и понимания, как надо решать задачу. Это важный блок вашего отчета.

# Тексты модулей

Здесь вы должны разместить тексты всех модулей проекта. Предлагается вставлять в отчет тексты модулей в виде копий экрана в приложении Lazarus. Т.е., в редакторе кода выбираете модуль, пролистываете до нужного мета, копируете активное окно при помощи Alt-PrtScr или другим способом (в Win 10, например, есть функция копии в буфер произвольной области экрана) и вставляете копии в отчет так, чтобы не было пропусков в тексте программы. Подсвеченные Лазарусом тексты удобнее читать, да и копии экранов не сканируются Антиплагиатом.

# Инструкция к приложению

Напишите инструкцию, позволяющую освоить разработанный вами векторный редактор.

# Заключение

Сформулируйте, что вы сделали в ходе исполнения проекта. Что разработано, кратко функционал и, по-вашему, какие достоинства имеет этот проект.

# Список литературы

В списке литературы обязательно указываются нижеперечисленные источники. В продолжение списка добавляете ту литературу и те ссылки в Интернете, которые вы использовали в своей работе в дополнение к основным.

1. Хованец В.А. Технология программирования и абстрактные типы данных// Методические указания – Владивосток: Изд-во МГУ им. адм. Г.И. Невельского, 2011. – 36 с.
2. Хованец В.А. Событийно-управляемое программирование// Методические указания – Владивосток: Изд-во МГУ им. адм. Г.И. Невельского, 2011. – 26 с.
3. Хованец В.А. Курс лекций «Программирование» [Текст]: учеб-метод, пособие / В.А. Хованец. – Владивосток: Изд-во МГУ им. адм. Г.И. Невельского, 2014. – 90 с. (электронный вариант).
4. Хованец В.А. Курс «Программирование» в системе электронного обучения МГУ им. адм. Г.И. Невельского, 2 курс ФЭИТ ИВТ, осенний семестр 2019-2020, <http://course.msun.ru/course/view.php?id=10179>.
5. Хованец В.А. Курс «Программирование» в системе электронного обучения МГУ им. адм. Г.И. Невельского, 2 курс ФЭИТ ИВТ, весенний семестр 2019-2020, <http://course.msun.ru/course/view.php?id=10178>.
6. Форум по курсовому проекту «Векторный редактор», <http://course.msun.ru/mod/forum/view.php?id=52933>
7. https://metanit.com/java/tutorial/5.8.php