Ульяновский государственный технический университет
Кафедра «Вычислительная техника»

Выпускная работа бакалавра

«Масштабирование изображений с учётом контента»

Специальность 09.03.01

Студент: **Кузьмин В.А.**
Группа: **ИВТАПбд-41**
Руководитель: **Валюх В.В.**

Ульяновск -
2019

Постановка задачи

Цель данного дипломного проекта заключается в разработке подсистемы для масштабирования растровых изображений с учётом контента.

Цель и задачи работы

Цель работы - спроектировать и реализовать независимую подсистему масштабирования изображений.

Задачи:

- Провести анализ существующих систем обработки изображений.
- Спроектировать и реализовать подсистему масштабирования изображений.
- Провести тестирование и отладку подсистемы масштабирования.

Функциональные требования

1) Наличие интерфейса подпрограммы;
2) Реализация класса изменения изображения;
3) Обеспечение загрузки/сохранения изображения;
4) Отображение загруженного изображения;
5) Выбор алгоритма ресайзинга;
6) Тестирование работающей подсистемы.

Использование алгоритма Seam Carving

Алгоритм состоит из составных частей:

- Нахождение энергии каждой точки;
- Нахождение такой вертикальной цепочки пикселей, чтобы суммарная энергия пикселей, которые входят в эту цепочку была минимальной;
- Удаление минимальной цепочки.

Вычисление энергии пикселя

Двойная функция градиента энергии

(255, 101, 51)	(255, 101, 153)	(255, 101, 255)
(255,153,51)	(255,153,153)	(255,153.255)
(255,203,51)	(255,204,153)	(255,205,255)
(255,255,51)	(255,255,153)	(255,255,255)

195075,0	195075,0	195075,0
195075,0	52225,0	195075,0
195075,0	52024,0	195075,0
195075,0	195075,0	195075,0

Энергия пикселя (1,2) вычисляется как:

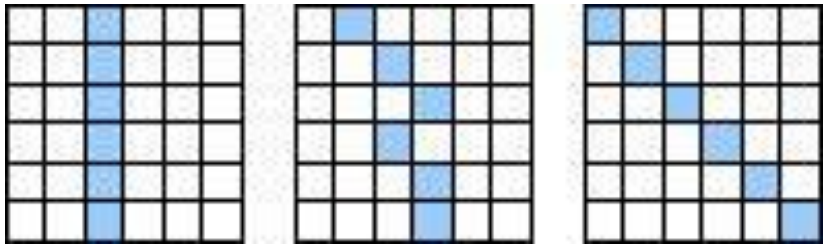
$$R_x(1,2) = 255 - 255 = 0,$$
$$G_x(1,2) = 205 - 203 = 2,$$
$$B_x(1,2) = 255 - 51 = 204,$$
$$\Delta_x^2(1,2) = 2^2 + 204^2 = 41620$$

Энергия пикселя (1,2) вычисляется как:

$$R_x(1,2) = 255 - 255 = 0,$$
$$G_x(1,2) = 205 - 203 = 2,$$
$$B_x(1,2) = 255 - 51 = 204,$$
$$\Delta_x^2(1,2) = 2^2 + 204^2 = 41620$$

Таким образом, энергия пикселя (1, 2) равна 41620 + 10404 = 52024.

Нахождение цепочки с минимальной суммарной энергией



5	5	3	5
3	3	4	4
4	6	0	2
1	5	4	0

энергии

5	5	3	5
3	3	4	4
4	6	0	2
1	5	4	0

суммы

5	5	3	5

энергии

5	5	3	5
3	3	4	4
4	6	0	2
1	5	4	0

суммы

5	5	3	5
8	6	7	7

энергии

5	5	3	5
3	3	4	4
4	6	0	2
1	5	4	0

суммы

5	5	3	5
8	6	7	7
10	12	6	9
11	11	10	6

энергии

5	5	3	5
3	3	4	4
4	6	0	2
1	5	4	0

суммы

5	5	3	5
8	6	7	7
10	12	6	9
11	11	10	6

энергии

5	5	3	5
3	3	4	4
4	6	0	2
1	5	4	0

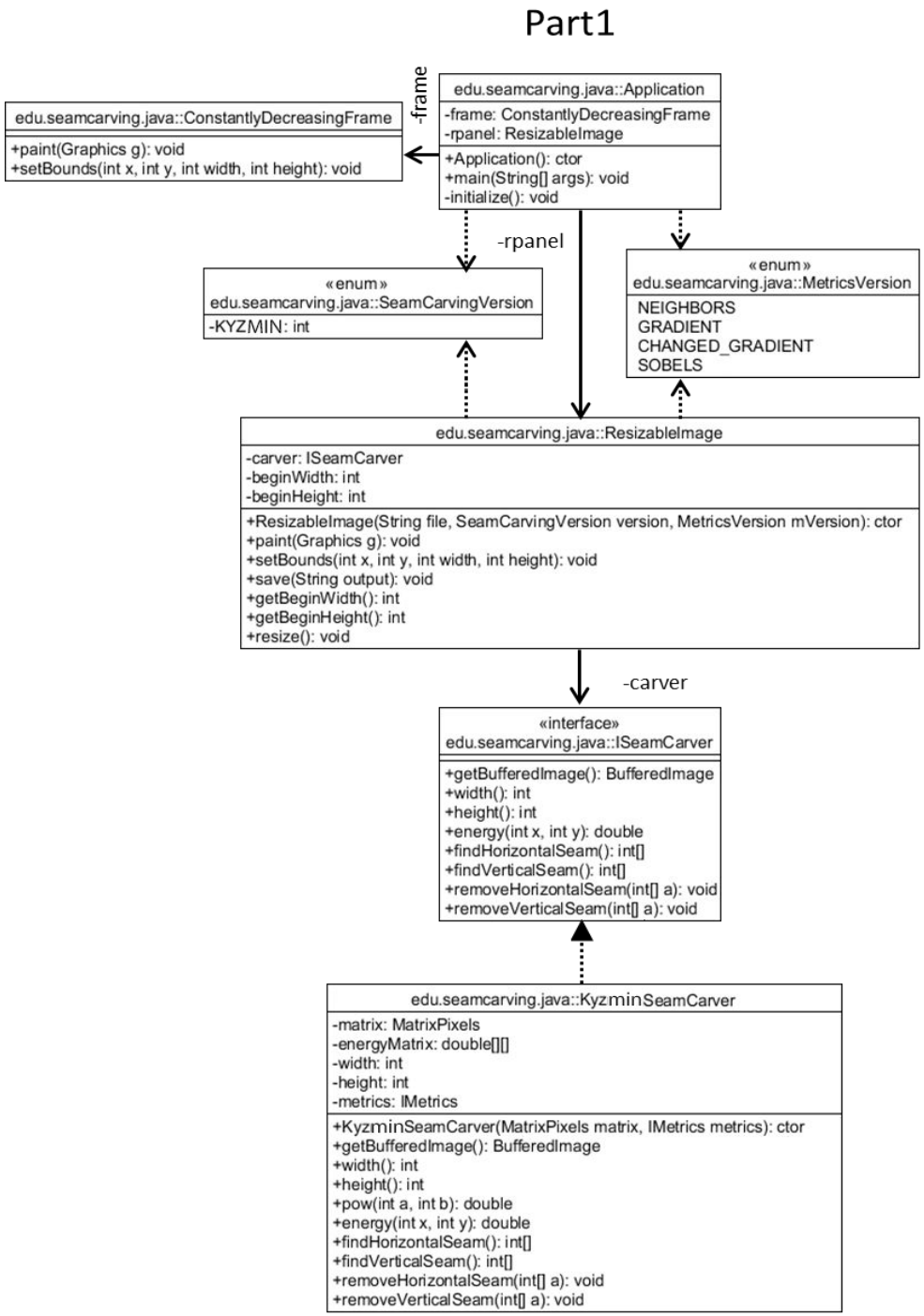
суммы

5	5	3	5
8	6	7	7
10	12	6	9
11	11	10	6

Создание классов

Теперь создаем необходимые классы отвечающие за графический интерфейс и алгоритмическую часть подсистемы. Создаем для наглядности UML-диаграмму классов

Диаграмма классов



Part2

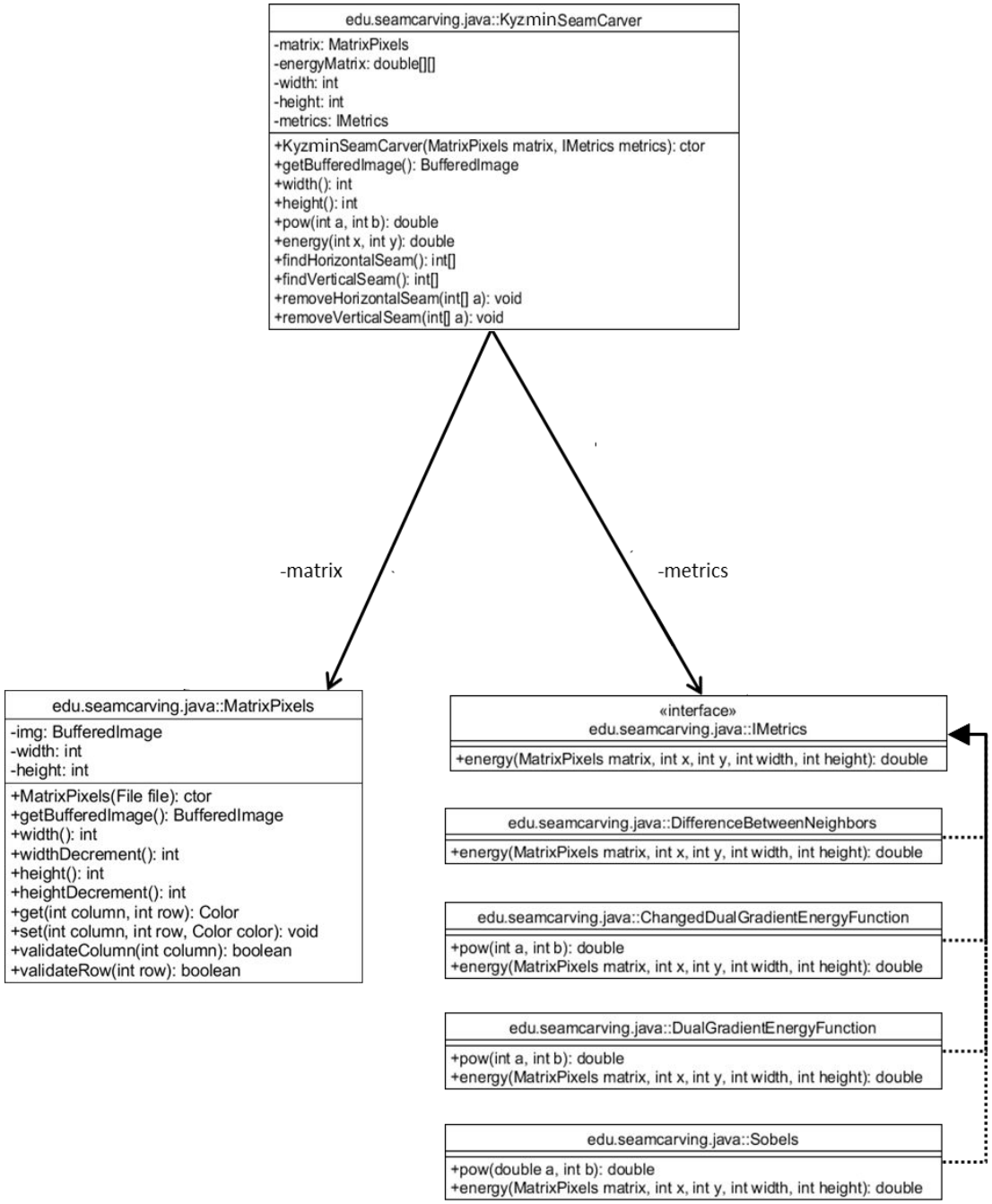
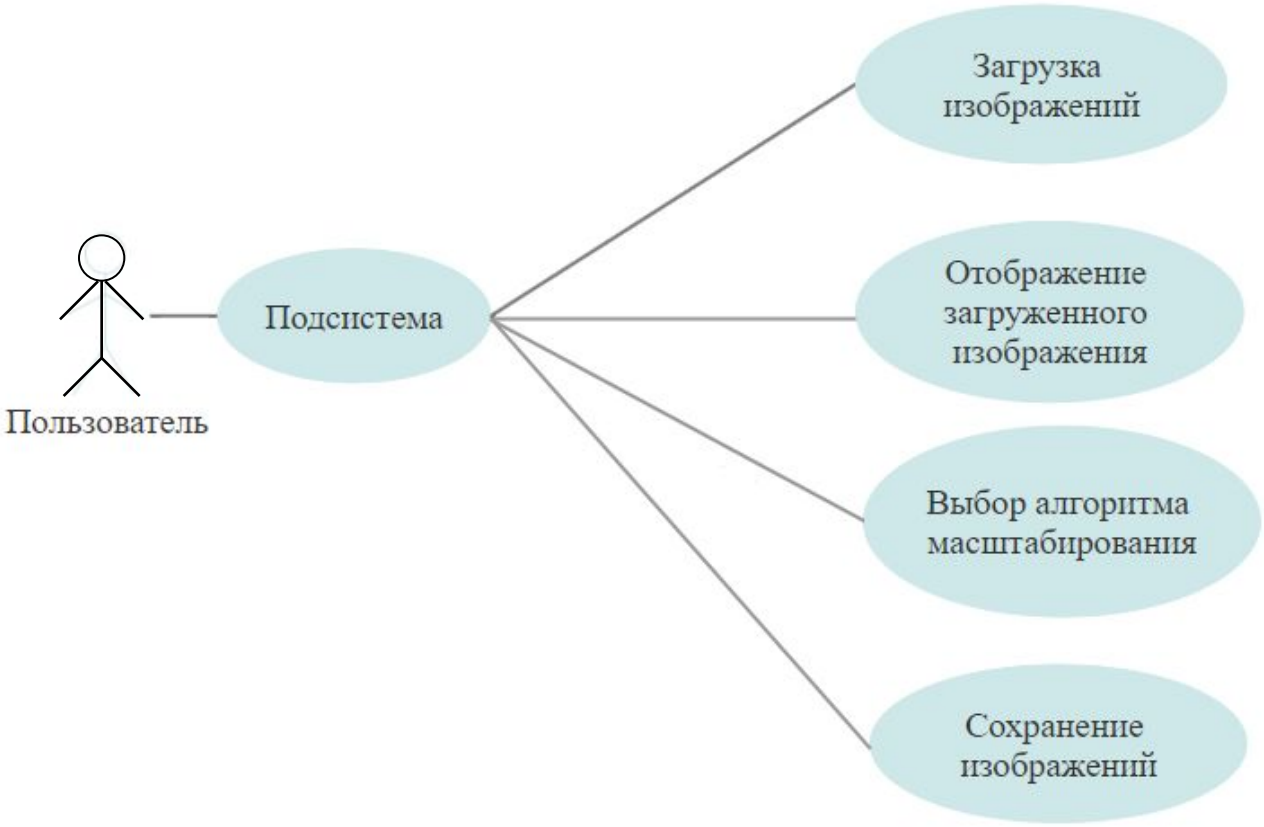
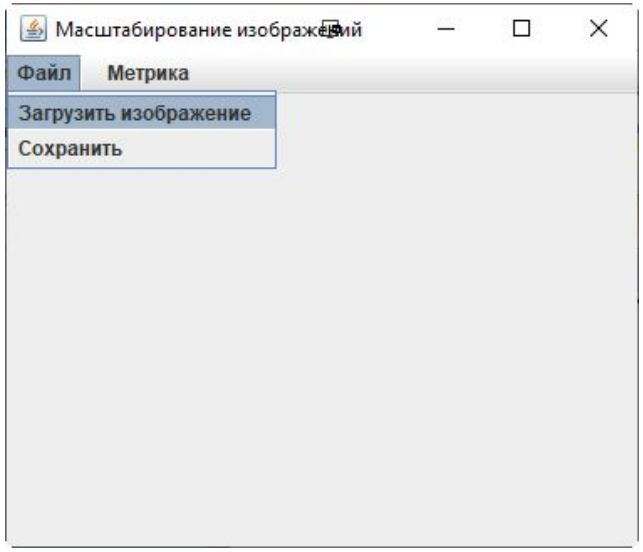
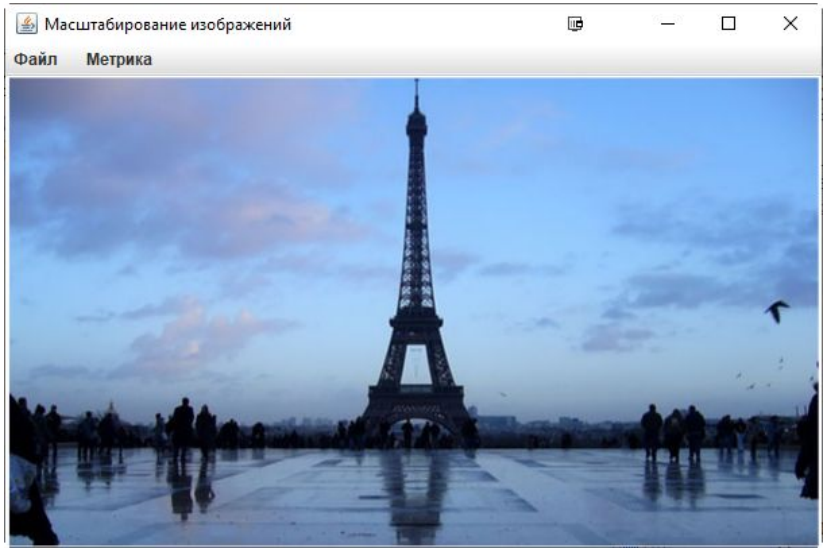

edu.seamcarving.java::KyzminSeamCarver
-matrix: MatrixPixels
-energyMatrix: double[][]
-width: int
-height: int
-metrics: IMetrics
+KyzminSeamCarver(MatrixPixels matrix, IMetrics metrics): ctor
+getBufferedImage(): BufferedImage
+width(): int
+height(): int
+pow(int a, int b): double
+energy(int x, int y): double
+findHorizontalSeam(): int[]
+findVerticalSeam(): int[]
+removeHorizontalSeam(int[] a): void
+removeVerticalSeam(int[] a): void
-matrix
-metrics
edu.seamcarving.java::MatrixPixels
-img: BufferedImage
-width: int
-height: int
+MatrixPixels(File file): ctor
+getBufferedImage(): BufferedImage
+width(): int
+widthDecrement(): int
+height(): int
+heightDecrement(): int
+get(int column, int row): Color
+set(int column, int row, Color color): void
+validateColumn(int column): boolean
+validateRow(int row): boolean
«interface»
edu.seamcarving.java::IMetrics
+energy(MatrixPixels matrix, int x, int y, int width, int height): double
edu.seamcarving.java::DifferenceBetweenNeighbors
+energy(MatrixPixels matrix, int x, int y, int width, int height): double
edu.seamcarving.java::ChangedDualGradientEnergyFunction
+pow(int a, int b): double
+energy(MatrixPixels matrix, int x, int y, int width, int height): double
edu.seamcarving.java::DualGradientEnergyFunction
+pow(int a, int b): double
+energy(MatrixPixels matrix, int x, int y, int width, int height): double
edu.seamcarving.java::Sobels
+pow(double a, int b): double
+energy(MatrixPixels matrix, int x, int y, int width, int height): double

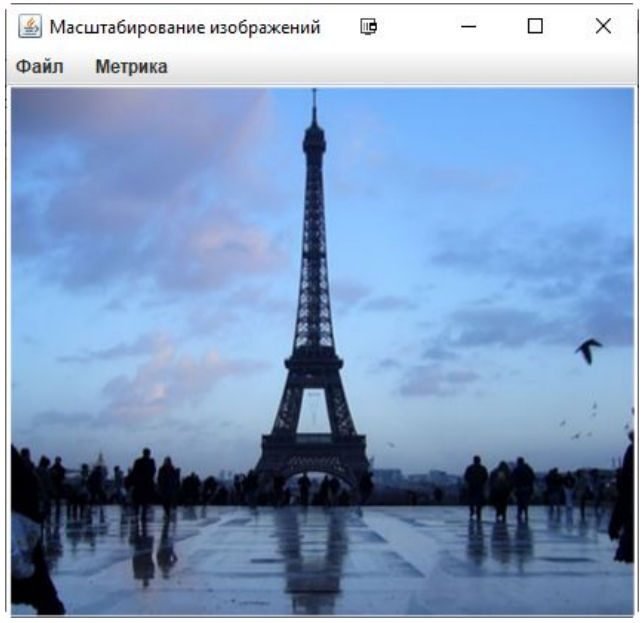
Диаграмма взаимодействия пользователя с системой

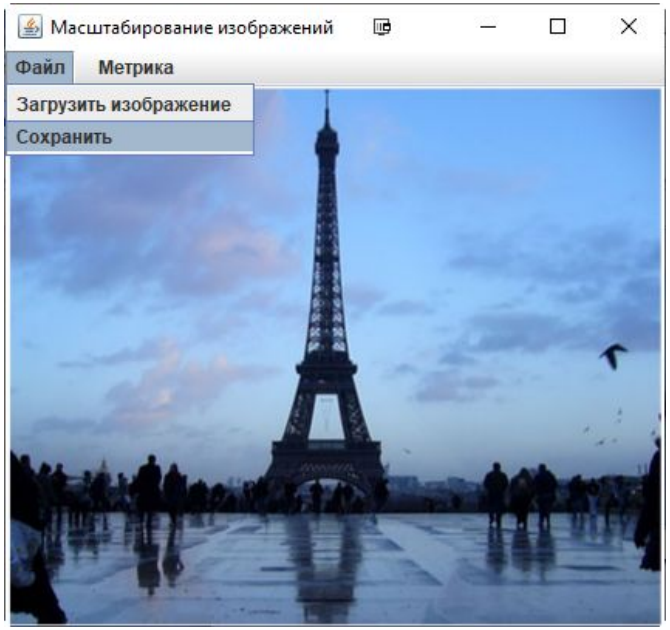


Работоспособность подсистемы





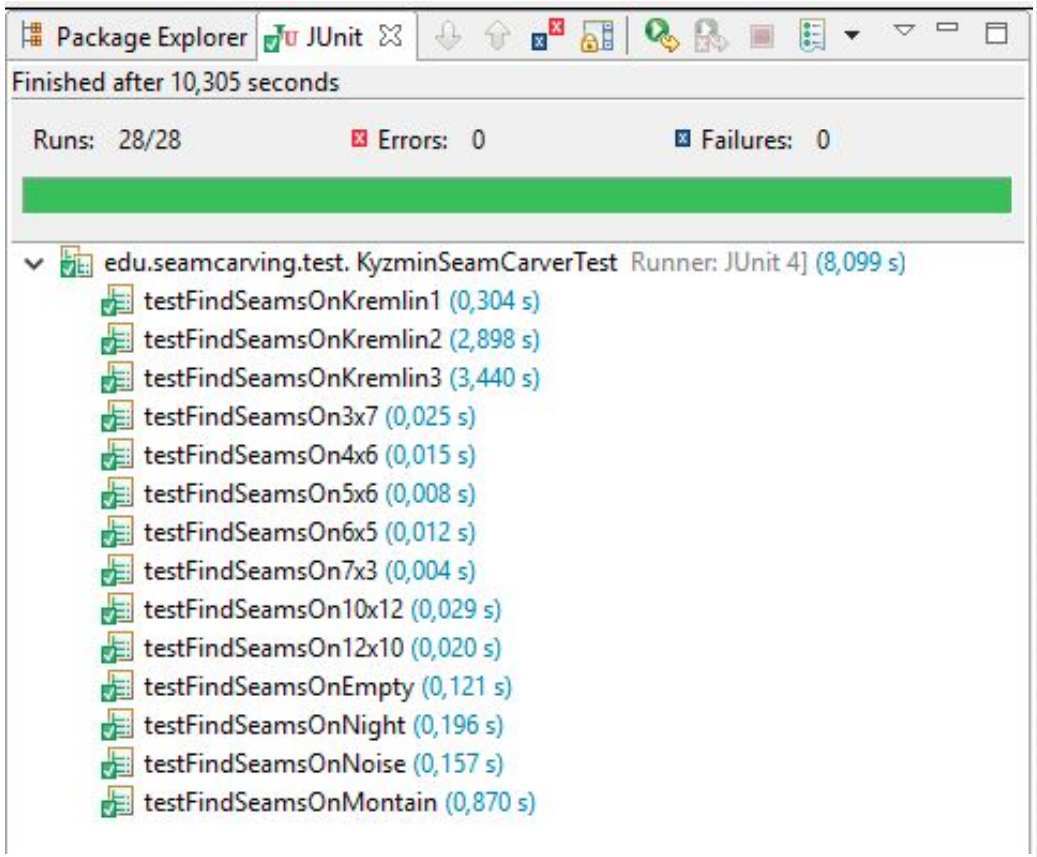




Тестирование

Для тестирования класса было создано 14 unit-тестов:

1. testFindSeamsOn3x7
2. testFindSeamsOn4x6
3. testFindSeamsOn5x6
4. testFindSeamsOn6x5
5. testFindSeamsOn7x3
6. testFindSeamsOn10x12
7. testFindSeamsOn12x10
8. testFindSeamsOnMountains
9. testFindSeamsOnBridge
10. testFindSeamsOnLake
11. testFindSeamsOnSea
12. testFindSeamsOnBlackWhite
13. testFindSeamsOnRed
14. testFindSeamsOnGreen



Заключение

В ходе выполнения работы был рассмотрен алгоритм под названием «Seam Carving». Работа потребовала изучения большого количества литературного материала. На основе него были получены знания в работе с такими структурами данных, как графы и усовершенствованы навыки в области динамического программирования. Так же в процессе выполнения проекта мы научились работать с Unit-тестами, т.е. писать их и составлять.

Интерфейс созданной подсистемы удобен, прост, наглядно отображает ее возможности. Главное меню редактора содержит команды работы с файлами, команды выбора метрик для масштабирования.

Тестирование подтвердило, что программа корректно выполняет обработку данных и демонстрацию результатов.

Спасибо за внимание!