

UNIWERYSTET JAGIELLOŃSKI

PROJEKT ZALICZENIOWY

Z KURSU INŻYNIERIA OPROGRAMOWANIA

Sudoku Solver

Daniel DOBROWOLSKI

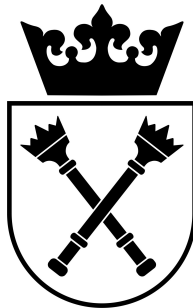
Małgorzata DYMEK

Tomasz JANIŁ

Łukasz KOSMATY

Nikodem KWAŚNIAK

Dawid SZCZERBA



Semestr letni 2018/2019

Spis treści

1	Testy jednostkowe.	2
1.1	Przypadki testowe.	2
1.2	Raporty o błędach.	90
2	Testy systemowe.	99
2.1	Testy poszczególnych scenariuszy przypadków użycia.	99
3	Dodatkowe testy dotyczące obróbki i rozpoznania obrazu.	131
3.1	Testy odczytu Sudoku ze zdjęcia.	131
3.1.1	Faza 1	132
3.1.2	Faza 2	132
3.1.3	Podsumowanie	134
3.2	Klasa DumpManualTest	135
3.2.1	Hierarchia	135
3.2.2	Proces Testowania	136

1 Testy jednostkowe.

Przypadki testowe dla testów jednostkowych zostały opisane według poniższego wzoru:

ID			
Tytuł		Oczekiwany rezultat	
Opis		Poprawność wykonania	
Warunek wstępny		Kroki	

Zaś raporty o wykrytych błędach zostały oparte na poniższym:

Raport o błędzie

ID		Nr commitu	
Tytuł		Oczekiwany rezultat	
Priorytet		Rzeczywisty rezultat	
Warunki wstępne		Kroki	

1.1 Przypadki testowe.

PRZYPADKI TESTOWE DLA WEB – TESTY JEDNOSTKOWE

```
package pl.sudokusolver.solver.utility
```

1. UtilityTest

```
package pl.sudokusolver.solver.utility
```

ID 1.1			
Tytuł	UsedInRow0TestTrue	Oczekiwany rezultat	Każde użycie funkcji usedInRow dla zerowego wiersza zwraca true
Opis	Sprawdzenie czy dana liczba występuje w zerowym wierszu	Poprawność wykonania	Ok
Warunek wstępny	brak	Kroki	---

ID 1.2			
Tytuł	UsedInRowTestIndexOutOfBoundsException	Oczekiwany rezultat	Rzucenie IndexOutOfBoundsException
Opis	Sprawdzenie czy dla wierszy nie występujących w macierzy funkcja rzuci wyjątek	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 1.3

Tytuł	UsedInRow0TestFalse	Oczekiwany rezultat	Każde użycie funkcji usedInRow dla zerowego wiersza zwraca false
Opis	Sprawdzenie czy dana liczba nie występuje w zerowym wierszu	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 1.4			
Tytuł	UsedInRowTestTrue	Oczekiwany rezultat	Każde użycie funkcji usedInRow dla danego wiersza zwraca true
Opis	Sprawdzenie czy dana liczba występuje w danym wierszu	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 1.5			
Tytuł	UsedInRowTestFalse	Oczekiwany rezultat	Każde użycie funkcji usedInRow dla danego wiersza zwraca false
Opis	Sprawdzenie czy dana liczba nie występuje w danym wierszu	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 1.6			
--------	--	--	--

Tytuł	UsedInCol0TestTrue	Oczekiwany rezultat	Każde użycie funkcji usedInRow dla zerowej kolumny zwraca true
Opis	Sprawdzenie czy dana liczba występuje w zerowej kolumnie	Poprawność wykonania	Ok
Warunek wstępny	brak	Kroki	---

ID 1.7			
Tytuł	UsedInColTestIndexOutOfBoundsException	Oczekiwany rezultat	Rzucenie IndexOutOfBoundsException
Opis	Sprawdzenie czy dla kolumn nie występujących w macierzy funkcja rzuci wyjątek	Poprawność wykonania	Ok
Warunek wstępny	brak	Kroki	---

ID 1.8			
Tytuł	UsedInCol0TestFalse	Oczekiwany rezultat	Każde użycie funkcji usedInCol dla zerowej kolumny zwraca false
Opis	Sprawdzenie czy dana liczba nie występuje w zerowej kolumnie	Poprawność wykonania	Ok

Warunek wstępny	brak	Kroki	---
------------------------	------	--------------	-----

ID 1.9			
Tytuł	UsedInColTestTrue	Oczekiwany rezultat	Każde użycie funkcji usedInCol dla danej kolumny zwraca true
Opis	Sprawdzenie czy dana liczba występuje w danej kolumnie	Poprawność wykonania	Ok
Warunek wstępny	brak	Kroki	---

ID 1.10			
Tytuł	UsedInColTestFalse	Oczekiwany rezultat	Każde użycie funkcji usedInCol dla danej kolumny zwraca false
Opis	Sprawdzenie czy dana liczba nie występuje w kolumnie	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 1.11			
Tytuł	UsedInBox1TestTrue	Oczekiwany rezultat	Funkcja usedInBox zwraca true

Opis	Sprawdzenie czy dana liczba występuje w bloku 3x3, gdzie górnym lewym rogiem jest row=0,col=0	Poprawność wykonania	Ok
Warunek wstępny	brak	Kroki	---

ID 1.12			
Tytuł	UsedInBoxTestIndexOutOfBoundsException	Oczekiwany rezultat	Rzucenie IndexOutOfBoundsException
Opis	Sprawdzenie czy funkcja rzuci wyjątek, jeżeli nasz blok 3x3 będzie wychodził poza macierz	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 1.13			
Tytuł	UsedInBox1TestFalse	Oczekiwany rezultat	Funkcja usedInBox zwraca false
Opis	Sprawdzenie czy dana liczba nie występuje w bloku 3x3, gdzie górnym lewym rogiem jest row=0,col=0	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 1.14			
Tytuł	UsedInBoxTestTrue	Oczekiwany rezultat	Każde użycie funkcji usedInBox dla danego danego bloku zwraca true
Opis	Sprawdzenie czy dana liczba występuje w danym bloku 3x3	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 1.15			
Tytuł	UsedInBoxTestFalse	Oczekiwany rezultat	Każde użycie funkcji usedInBox dla danego bloku zwraca false
Opis	Sprawdzenie czy dana liczba nie występuje w danym bloku 3x3	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 1.16			
Tytuł	CanPlaceDigitTestTrue	Oczekiwany rezultat	Dla zadanych wartości funkcja CanPlaceDigit zwraca true

Opis	Sprawdzenie czy dana liczba <u>nie występuje</u> w danej kolumnie, wierszu i bloku 3x3	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 1.17			
Tytuł	CanPlaceDigitIndexOutOfBoundsExceptionTest	Oczekiwany rezultat	Rzucenie IndexOutOfBoundsException
Opis	Sprawdzenie czy funkcja rzuci wyjątek IndexOutOfBoundsException	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 1.18			
Tytuł	CanPlaceDigitTestFalse	Oczekiwany rezultat	Dla zadanych wartości funkcja CanPlaceDigit zwraca false
Opis	Sprawdzenie czy dana liczba <u>występuje</u> w danej kolumnie, wierszu i bloku 3x3	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 1.19			
Tytuł	getUnassignedLocationTrue	Oczekiwany rezultat	Para liczb zwracana przez funkcję oraz podana przez testera jest taka sama
Opis	Sprawdzenie czy dla danych macierzy(testowane na trzech macierzach) funkcja zwróci dobrą pierwszą lokalizację, do której nie została przypisana liczba	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 1.20			
Tytuł	getUnassignedLocationFalse	Oczekiwany rezultat	Para liczb zwracana przez funkcję oraz podana przez testera(inna wolna lokalizacja, ale nie pierwsza) jest inna
Opis	Sprawdzenie czy dla danych macierzy(testowane na trzech macierzach) funkcja zwróci dobrą pierwszą lokalizację, do której nie została przypisana liczba	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 1.21			
Tytuł	gridToStringTrue	Oczekiwany rezultat	Funkcja gridToString zwraca macierz w postaci stringa, według przyjętej przez programistę konwencji

Opis	Sprawdzenie czy funkcja poprawnie przekształca macierz na ciąg znaków	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

2. BrutalSolverTest

```
package pl.sudokusolver.solver;
```

ID 2.1			
Tytuł	solveSudokuTrueAndEqualsTest	Oczekiwany rezultat	Zwrócenie macierzy (rozwiązanych sudoku) identycznych jak dane przez testera, oraz zwrócenie true
Opis	Sprawdzenie czy algorytm działa zgodnie z oczekiwaniem, czyli rozwiązuje sudoku według algorytmu brutalnego oraz czy zwraca true, jeżeli udało się rozwiązać sudoku (test na dwóch sudoku, jedno same zera)	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 2.2			
Tytuł	solveSudokuTrueAndNotEqualsTest	Oczekiwany rezultat	Zwrócenie innych macierzy niż danych przez testera, zwrócenie false bo sudoku udało się rozwiązać

Opis	Sprawdzenie czy algorytm działa zgodnie z oczekiwaniem, czyli rozwiązuje sudoku w inny sposób niż tester oraz zwrócenie true, jeżeli sudoku jest możliwe do rozwiązania	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 2.3			
Tytuł	solveSudokuFalseTest	Oczekiwany rezultat	Funkcja solve zwraca false, ponieważ dostarczone przez testera macierze są niemożliwe do rozwiązania
Opis	Sprawdzenie czy funkcja zwraca false dla sudoku, którego nie da się rozwiązać(sprawdzone na 4 sudoku, 2 skrajne przypadki)	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

3. SmartSolverTest

```
package pl.sudokusolver.solver;
```

ID 3.1

Tytuł	solveSudokuTest	Oczekiwany rezultat	Zwrócenie macierzy (rozwiązanych sudoku) identycznych jak dane przez testera, oraz zwrócenie true
Opis	Sprawdzenie czy algorytm rozwiązuje sudoku według algorytmu SmartSolver	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 3.2			
Tytuł	solveSudokuTrueAndEqualsTest	Oczekiwany rezultat	Zwrócenie takiej samej macierzy, jak dana przez testera
Opis	Sprawdzenie czy algorytm działa zgodnie z oczekiwaniem, dla zerowej macierzy	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 3.3			
Tytuł	solveSudokuTrueAndNotEquals	Oczekiwany rezultat	Funkcja solve powinna zwrocic solve, ale sudoku powinno zostać rozwiązane w inny sposób niż dane przez testera
Opis	Sprawdzenie algorytmu	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 3.3			
Tytuł	solveSudokuFalse	Oczekiwany rezultat	Funkcja solve zwraca false, ponieważ dostarczone przez testera macierze są niemożliwe do rozwiązania
Opis	Sprawdzenie czy funkcja zwraca false dla sudoku, którego nie da się rozwiązać - dwa skrajne przypadki	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

4. SmartSolvePerformanceTest

```
package pl.sudokusolver.solver;  
@ignore
```

ID 4.1			
Tytuł	SmartSolvePerformanceTest	Opis i oczekiwany rezultat	Klasa stworzona po to, żeby przetestować czas rozwiązywania Sudoku algorytmem SmartSolve dla dużej liczby danych. Dostarczamy 159 sudoku do rozwiązania i średni czas dla jednego sudoku wynosi 31.232704402515722 ms co jest dla nas czasem w pełni zadowalającym, ponieważ w wymaganiach projektu mamy, że czas rozwiązywania dla jednego sudoku powinien wynosić poniżej 7 sekund

```
package pl.sudokusolver.recognizerlib;
```

5. CenterLinesComparatorTest

```
package pl.sudokusolver.recognizerlib.utility.comparators;
```

ID 5.1			
Tytuł	compareTest	Oczekiwany rezultat	Funkcja powinna zwrócić jeden jeśli punkt1.x > punkt2.y, zero jeśli punkt1.x==punkt2.y, minus jeden jeśli punkt1.x<punkt2.y
Opis	Sprawdzenie działania funkcji compare z klasy CenterLinesComparator, sprawdzam kilka razy wszystkie trzy możliwe wartości, które może zwracać compare	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

6. PointComparatorTest

```
package pl.sudokusolver.recognizerlib.utility.comparators;
```

ID 6.1			
Tytuł	compareTestOnlyY	Oczekiwany rezultat	Funkcja powinna zwrócić jeden jeśli $(\text{punkt1.y} - \text{punkt2.y}) \geq 10$, zero jeśli $ \text{punkt1.y} - \text{punkt2.y} < 10$, minus jeden jeśli $\text{punkt1.y} - \text{punkt2.y} \leq -10$
Opis	Sprawdzenie działania funkcji compare z klasy PointComparator, w tym przypadku testowym porównujemy punkty(x,y), które różnią się tylko wartościami y	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 6.2			
Tytuł	compareTestOnlyX	Oczekiwany rezultat	Funkcja powinna zwrócić jeden jeśli $\text{punkt1.x} > \text{punkt2.x}$, zero jeśli $\text{punkt1.x} == \text{punkt2.x}$, minus jeden jeśli $\text{punkt1.x} < \text{punkt2.x}$

Opis	Sprawdzenie działania funkcji compare z klasy PointComparator, w tym przypadku testowym porównujemy punkty(x,y), które różnią się tylko wartościami x, a różnice między wartościami y obu punktów są mniejsze niż 10	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 6.3			
Tytuł	compareTest	Oczekiwany rezultat	Dla zadanych przez testera punktów, funkcja powinna zwracać najpierw trzy 0, potem trzy -1, potem trzy 1
Opis	Sprawdzenie działania funkcji compare z klasy PointComparator, porównujemy punkty testując wszystkie trzy wartości, które może zwracać funkcja compare, funkcja compare porównuje najpierw po y według warunków, z poprzednich przypadków testowych w tej klasie, potem po x	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

7. ImageProcessingTest

```
package pl.sudokusolver.recognizerlib.utility.staticmethods;
```

ID 7.1			
Tytuł	procSimpleTest	Oczekiwany rezultat	Macierz o jednym wierszu, size*size kolumnach oraz wszystkich komórkach o wartości 1.0
Opis	Sprawdzenie czy funkcja procSimple działa prawidłowo, czyli czy zwraca macierz o jednym wierszu oraz size*size kolumnach oraz sprawdzenie w pętli czy zwraca oczekiwane przez nas wartości dla poszczególnych komórek	Poprawność wykonania	Ok

Warunek wstępny	- macierz musi być typu CV_32FC1 -załadowanie OpenCV	Kroki	---
------------------------	---	--------------	-----

ID 7.2			
Tytuł	centerTest	Oczekiwany rezultat	Szerokość i wysokość macierzy po działaniu funkcji, powinna wynosić size - czyli parametr funkcji
Opis	Sprawdzenie czy funkcja center przekształca zdjęcie w zdjęcie o określonym rozmiarze równym size*size	Poprawność wykonania	Ok
Warunek wstępny	-macierz musi być typu CV_32FC1 -załadowanie OpenCV	Kroki	---

ID 7.3			
Tytuł	deskewTest	Oczekiwany rezultat	Macierz wejściowa o rozmiarze 10x10 ma wartość 255 w komórkach gdzie indeks kolumny równy indeksowi wiersza, po użyciu funkcji deskew wartości te powinny być w 5 kolumnie

Opis	Funkcja deskew obraca macierz, więc aby sprawdzić czy robi to prawidłowo przypisuje do macierzy początkowej wartości do ustalonych kolumn, wykonuje funkcje deskew i sprawdzam czy w odpowiednich kolumnach dostaje przypisane wartosci	Poprawność wykonania	Ok
Warunek wstępny	- macierz musi być typu -załadowanie OpenCV	Kroki	---

ID 7.4			
Tytuł	applyMaskTest	Oczekiwany rezultat	Wpisuje do macierzy wejściowej wartosci 250, nakładam maskę i liczę ile razy wystąpiła wartość 250 w macierzy po aplikacji maski
Opis	Sprawdzam czy maska jest prawidłowo nakładana, dlatego tworzę macierz, przypisuję do niej wartości, nakładam maskę i sprawdzam wartosci w macierzy po nałożeniu maski	Poprawność wykonania	Ok
Warunek wstępny	- macierz musi być typu -załadowanie OpenCV	Kroki	---

8. UtilityTest

```
package pl.sudokusolver.recognizerlib.utility.staticmethods;
```

ID 8.1

Tytuł	orderPointTest1	Oczekiwany rezultat	Posortowana macierz punktów (x,y) po sumie x+y
Opis	Sprawdzenie czy funkcja orderPoints prawidłowo sortuje punkty według sumy x+y i zwraca poprawną macierz	Poprawność wykonania	Ok
Warunek wstępny	-tylko liczby naturalne -załadowanie OpenCV	Kroki	---

ID 8.2			
Tytuł	orderPointTest2	Oczekiwany rezultat	Posortowana macierz punktów (x,y) po sumie x+y
Opis	Sprawdzenie czy funkcja orderPoints prawidłowo sortuje punkty według sumy x+y i zwraca poprawną macierz dla innego zestawu danych	Poprawność wykonania	Ok
Warunek wstępny	-tylko liczby naturalne -załadowanie OpenCV	Kroki	---

ID 8.3			
Tytuł	distanceTest1	Oczekiwany rezultat	Funkcja zwraca całkowitą część euklidesowej odległości w wektorze dla zadanych punktów
Opis	Sprawdzenie czy funkcja distance prawidłowo oblicza odległość między punktami	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 8.4			
Tytuł	distanceTest2	Oczekiwany rezultat	Funkcja zwraca całkowitą część euklidesowej odległości w wektorze dla innych zadanych punktów
Opis	Sprawdzenie czy funkcja distance prawidłowo oblicza odległość między punktami	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

ID 8.5			
Tytuł	ApplyFiltersTest	Oczekiwany rezultat	Macierz po wykonaniu funkcji powinna mieć postać jak po nałożeniu filtru NOT
Opis	Sprawdzenie poprawności funkcji nakładającej filtry na macierz (sprawdzone filtry NOT oraz brak filtra null)	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 8.6

Tytuł	applyResizeAndToGrayFiltersTest	Oczekiwany rezultat	Macierz po wykonaniu funkcji applyFilters powinna mieć postać jak po wykonaniu filtrów Resize oraz ToGrayFilter dla danych zadanych przez testera
Opis	Sprawdzenie poprawności funkcji nakładającej filtry na macierz (sprawdzone filtry Resize i ToGrayFilter)	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 8.7

Tytuł	applyFilterCounterTest	Oczekiwany rezultat	Ilość nałożonych filtrów powinna się zgadzać z ilością elementów(którymi są obiekty klas z package filters) listy
Opis	Sprawdzenie czy funkcja applyFilters nakłada dokładnie tyle filtrów ile jest w liście, którą przekazujemy do funkcji	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 8.8

Tytuł	getFileFromResourcesTest	Oczekiwany rezultat	Trzy testy wykonane w tym przypadku powinny zwrócić true, przy sprawdzeniu czy pliki istnieją
Opis	Sprawdzenie czy funkcje getANNDump, getSVMDump, getTessData zwracają pliki, które istnieją	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 8.9			
Tytuł	matToBufferedImageTest	Oczekiwany rezultat	Zwrócenie odpowiednich typów dla macierzy jednokanałowej i trzykanałowej
Opis	Sprawdzanie czy dla macierzy jednokanałowych oraz trzykanałowych po użyciu na macierzy funkcji matToBufferedImage, obiekt typu BufferedImage zwraca dobry typ	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV - dopuszczalne macierze zwracające channels 1 lub channels 3	Kroki	---

9. BlurFilterTest

```
package pl.sudokusolver.recognizerlib.filters;
```

ID 9.1			
Tytuł	blurChangeParametersTest	Oczekiwany rezultat	Macierz po użyciu funkcji apply, powinna mieć takie same wartości w komórkach, jak wpisane przez testera
Opis	Sprawdzenie czy funkcja apply klasy blurFilter prawidłowo przekształca macierz zgodnie z https://docs.opencv.org/4.0.1/d4/d86/group_imgproc_filter.html#gaabe8c836e97159a9193fb0b11ac52cf1	Poprawność wykonania	Ok
Warunek wstępny	- macierz powinna być typu CV_8UC1 lub innego typu z jednym kanałem - parametry funkcji muszą spełniać założenia (size >= 0 && blockSize >= 2 && c >= 0 && size % 2 == 1) -załadowanie OpenCV	Kroki	---

ID 9.2			
Tytuł	blurDefParametersTest	Oczekiwany rezultat	Oczekiwana macierz powinna posiadać same zera
Opis	Sprawdzenie działania funkcji apply z klasy blurFilter	Poprawność wykonania	Ok
Warunek wstępny	<ul style="list-style-type: none"> - macierz powinna być typu CV_8UC1 lub innego typu z jednym kanałem - parametry funkcji muszą spełniać założenia (size >= 0 && blockSize >= 2 && c >= 0 && size % 2 == 1) -załadowanie OpenCV 	Kroki	---

ID 9.3			
Tytuł	wrongTypeOfMatrixTest	Oczekiwany rezultat	Rzucenie wyjątku CvException() przez funkcję apply
Opis	Sprawdzenie jak zareaguje funkcja apply na podanie nieprawidłowej macierzy	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 9.4			
Tytuł	wrongCreationTest	Oczekiwany rezultat	Konstruktor BlurFilter() powinien dla (złych) danych podanych przez testera rzucić wyjątek IllegalArgumentException

	Opis	Sprawdzenie czy podczas tworzenia obiektu klasy BlurFilter zostanie rzucony wyjątek, jeśli podamy złe dane czyli size < 0 lub blockSize < 2 lub c < 0 lub size%2!=1	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---	

10.FixedWidthResizeFilterTest

```
package pl.sudokusolver.recognizerlib.filters;
```

ID 10.1			
Tytuł	applyTest	Oczekiwany rezultat	Macierz po użyciu funkcji apply powinien mieć zmienioną wysokość według wzoru height=width*(input.size().height/ input.size().width);

Opis	Sprawdzenie czy funkcja prawidłowo przekształca macierz, czyli proporcjonalnie zmienia wysokość do zmiany szerokości	Poprawność wykonania	Ok
Warunek wstępny	- macierz powinna być jednokanałowa	Kroki	---
ID 10.2			
Tytuł	applyCvExceptionTest	Oczekiwany rezultat	Rzucenie wyjątku CvException przez funkcję apply
Opis	Sprawdzenie, czy funkcja apply rzuci wyjątek dla zmiany szerokości na ujemną lub złego typu macierz	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

11. MaxResizeFilterTest

```
package pl.sudokusolver.recognizerlib.filters;
```

ID 11.1

Tytuł	applyTest	Oczekiwany rezultat	Zwrócenie macierzy o takich rozmiarach, jak te które podał tester (sprawdzanie czy funkcja skaluje zgodnie z oczekiwaniami)
Opis	Jest to też test funkcji <code>resizeToMaxFilter</code> z klasy <code>ImageProcessing</code> . Funkcja zmienia rozmiar macierzy, skaluje macierz. Test bada wszystkie 4 ścieżki działania funkcji	Poprawność wykonania	Ok
Warunek wstępny	- parametry funkcji muszą być >0 -załadowanie OpenCV	Kroki	---

ID 11.2			
Tytuł	invalidCreationTest	Oczekiwany rezultat	Rzucenie <code>IllegalArgumentException()</code> podczas tworzenia obiektu klasy <code>MaxResizeFilter</code> dla podanych złych parametrów
Opis	Sprawdzenie, czy podczas tworzenia obiektu klasy zostanie rzucony wyjątek, jak podamy złe parametry(muszą być > 0)	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

12. MedianBlurTest

```
package pl.sudokusolver.recognizerlib.filters;
```

ID 12.1			
Tytuł	blurChangeParametersTest	Oczekiwany rezultat	Macierz zwraca wartości wpisane przez testera, więcej można znaleźć w dokumentacji openCv https://docs.opencv.org/4.0.1/d4/d86/group_imgproc_filter.html#g_aabe8c836e97159a9193fb0b11ac52cf1
Opis	Sprawdzenie działania funkcji apply, do której przekazujemy macierz	Poprawność wykonania	Ok
Warunek wstępny	- macierz powinna być typu CV_8U -załadowanie OpenCV	Kroki	---

ID 12.2			
Tytuł	blurDefParametersTest	Oczekiwany rezultat	Oczekiwany rezultat jak w poprzednim przypadku, czyli trzy wartości w pierwszym wierszu 255.0 oraz w dwóch kolejnych wierszach po trzy 0.0
Opis	Sprawdzenie działania funkcji apply, do której przekazujemy macierz, tworzymy obiekt klasy MedianBlur za pomocą konstruktora bez parametrow	Poprawność wykonania	Ok
Warunek wstępny	- oczekiwany typ macierzy CV_8U -załadowanie OpenCV	Kroki	---

ID 12.3			
Tytuł	wrongTypeOfMatrixTest	Oczekiwany rezultat	Rzucenie CvException przez konstruktor obiektu klasy MedianBlur

Opis	Jeśli podamy zły typ macierzy to powinien zostać rzucony wyjątek CvException	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 12.4			
Tytuł	wrongCreationTest	Oczekiwany rezultat	Rzucenie wyjątku IllegalArgumentException
Opis	Jeśli przekazujemy niewłaściwe parametry do funkcji powinien zostać rzucony wyjątek IllegalArgumentException, w tym teście właśnie przekazujemy argumenty, po których ten wyjątek ma wystąpić	Poprawność wykonania	Ok
Warunek wstępny	- macierz powinna być typu CV_8U -załadowanie OpenCV	Kroki	---

13. NotFilterTest

```
package pl.sudokusolver.recognizerlib.filters;
```

ID 13.1			
Tytuł	applyTest	Oczekiwany rezultat	Pierwszy test - uzyskanie macierzy z wartościami 255, drugi test - uzyskanie macierzy z samymi zerami
Opis	NotFilter wykonuje tylko bramkę not na macierzy składających się z samych zer, oczekuje otrzymania macierzy składających się z bitów 255, potem otrzymaną macierz znowu przepuszczam przez filtr żeby uzyskać same zera	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV - macierz powinna być typu CV_8U	Kroki	---

14. ResizeFilterTest

ID 14.1			
Tytuł	applyTest	Oczekiwany rezultat	Po wykonaniu funkcji apply rozmiar macierzy powinien być taki, jak wpisaliśmy w parametrach
Opis	Sprawdzam czy filtr poprawnie zmienia rozmiar macierzy	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 14.2			
Tytuł	invalidCreationsTest	Oczekiwany rezultat	Rzucenie wyjątku IllegalArgumentException przy każdej próbie stworzenia obiektu ResizeFilter dla size <0
Opis	Nie można ustawić rozmiaru macierzy na < 0, wtedy powinien zostać rzucony wyjątek IllegalArgumentException	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

15. ToGrayFilterTest

```
package pl.sudokusolver.recognizerlib.filters;
```

ID 15.1			
Tytuł	applyTest	Oczekiwany rezultat	Po użyciu apply na macierzach, macierze te powinny zwracać funkcja channels() powinna zwracać 1
Opis	ToGrayFilter tak przekształca macierz, żeby ta była jednokanałowa. Sprawdzamy na macierzach wielokanałowych, czy uda się je zmienić na jednokanałowe.	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 15.2			
Tytuł	applyTestIfChannel1	Oczekiwany rezultat	Funkcja channel() przed i po wykonaniu apply powinna zwracać to samo

Opis	Test sprawdza czy do funkcji da się podać też macierz jednokanałową, wtedy po wykonaniu apply powinna być ona dalej jednokanałowa	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

16.SudokuTest

```
package pl.sudokusolver.recognizerlib.sudoku;
```

ID 16.1			
Tytuł	sudokuOnCreateHaveEmptyGridTest	Oczekiwany rezultat	Konstruktor zwróci puste sudoku o wymiarach 9x9, funkcja empty() zwróci true

Opis	Test sprawdza, czy sudoku stworzone przez konstruktor bez żadnych parametrów jest macierzą 9x9 wypełnioną zerami, testujemy też w tym teście funkcję empty()	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 16.2			
Tytuł	readFromDatTest	Oczekiwany rezultat	Dla dwóch poprawnych ścieżek do dwóch plików funkcja readFromDat dla każdej z nich zwraca poprawne Sudoku, takie jak w pliku dat o danym numerze
Opis	Test sprawdza czy funkcja readFromDat prawidłowo odczytuje i tworzy sudoku z pliku .dat (są to pliki tekstowe, do których przepisywaliśmy sudoku ze zdjęcia dla łatwiejszego testowania aplikacji)	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV -pliki .dat z zapisanymi według schematu sudoku	Kroki	---

ID 16.3			
Tytuł	readFromDatExceptionTest	Oczekiwany rezultat	Funkcja readFromDat powinna rzucić IOException w dwóch pierwszych testach i w trzecim NullPointerException

Opis	Test sprawdza czy dla niepoprawnej ścieżki zostanie rzucony IOException oraz dla wstawionego null w miejsce ścieżki zostanie rzucony NullPointerException	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 16.4			
Tytuł	scoreTest	Oczekiwany rezultat	Funkcja score powinna zwracać dla każdego przypadku liczbę double taką samą jak obliczona ręcznie dla danego przypadku przez programistę, dopuszczalny błąd double wynosi 1E-5
Opis	Test sprawdza działanie funkcji score, która porównuje jedno sudoku z drugim oraz zwraca procent ich podobieństwa, przeprowadzam tutaj 4 testy, dla identycznych sudoku, dla całkowicie różnych, oraz dwa testy z wysokim i niskim stopniem podobieństwa	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 16.5			
Tytuł	toStringTest	Oczekiwany rezultat	Zwrócenie przez funkcję toString takiego samego stringa jak podany przez testera
Opis	Funkcja powinna przekształcać sudoku na stringa według założonego schematu	Poprawność wykonania	Ok

Warunek wstępny	-załadowanie OpenCV	Kroki	---
------------------------	---------------------	--------------	-----

17.SimpleRowDataTest

```
package pl.sudokusolver.recognizerlib.data;
```

ID 17.1			
Tytuł	getDataFromImgTest	Oczekiwany rezultat	<p>Obiekt klasy SimpleRowData powinien zwracać:</p> <ul style="list-style-type: none"> -DataType.Simple -size równy podanemu w konstruktorze - ilość wierszy dla dwóch macierzy ma być identyczna - poprawną ilość kolumn, dla obu macierzy
Opis	<p>W klasie SimpleRowData kluczowa jest funkcja loadFromSheet, która jest wykorzystywana w jednym z konstruktorów SimpleRowData, sprawdzam typ, rozmiar, liczbę kolumn i labels zwracanych przez obiekt tej klasy, stworzony za pomocą konstruktora z prywatną metodą loadFromSheets</p>	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV - zdjęcie do załadowania	Kroki	---

ID 17.2

Tytuł	simpleCreationTest	Oczekiwany rezultat	<p>Obiekt klasy SimpleRowData powinien zwracać:</p> <ul style="list-style-type: none"> -DataType.Simple -size równy podanemu w konstruktorze - ilość wierszy dla dwóch macierzy ma być identyczna - ilość kolumn dla obu macierzy ma być identyczna - identyczne wartości w obu macierzach
Opis	<p>W tym przypadku tworzę obiekt SimpleRowData w inny sposób, za pomocą podania konstruktora dwóch macierzy i rozmiaru, metoda testowania podobna do testu 17.1 + przed przekazaniem do funkcji nadałem macierzom identyczne wartości, na końcu sprawdzam, czy nic nie zostało przekształcone</p>	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

18.MnistReaderTest

```
package pl.sudokusolver.recognizerlib.data;
```

ID 18.1

Tytuł	mnistFilesDontExistTest	Oczekiwany rezultat	Dla dwóch poprawnych ścieżek do dwóch plików funkcja readFromDat dla każdej z nich zwraca poprawne Sudoku, takie jak w pliku dat o danym numerze
Opis	Sprawdzamy czy funkcja read rzuci IOException dla nieprawidłowych ścieżek	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 18.2			
Tytuł	versionMismatchedExceptionTest	Oczekiwany rezultat	Funkcja dla podanych przez testera danych plików powinna rzucić wyjątek VersionMismatchException
Opis	Przekazywane do funkcji pliki powinny mieć tą samą wersję	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV -pliki .dat z zapisanymi według schematu sudoku	Kroki	---

ID 18.3			
Tytuł	readCorrectFilesTest	Oczekiwany rezultat	Cztery gettery dla dwóch stworzonych macierzy powinny zwrócić wartości oczekiwane przez testera
Opis	W tym teście używać poprawnie funkcji read z klasy MNISTReader, potem w testach sprawdzam liczby kolumn i wierszy stworzonych macierzy	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

19. DigitBoxContoursTest

```
package pl.sudokusolver.recognizerlib.digitbox;
```

ID 19.1			
Tytuł	getDigitBoxTest	Oczekiwany rezultat	Oczekuje, że obiekt został poprawnie stworzony - dla obiektu opcjonalnego funkcja isPresent powinna zwrócić true, 4 gettery powinny zwrócić oczekiwane wartości
Opis	W tym przypadku tworzę macierz samych zer 30x30, potem nadaje wartości dla indeksów i, j ≥ 10 and $i, j \leq 20$. Sprawdzam czy funkcja getDigitBox poprawnie tworzy obiekt, zwraca pierwszy x,y, szerokosc i wysokosc	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 19.2			
Tytuł	digitNotFoundTest	Oczekiwany rezultat	Funkcja isPresent dla obiektu opcjonalnego powinna zwrócić false
Opis	Obiekt nie powinien zostać stworzony, bo podana macierz zawiera same zera	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 19.3			
Tytuł	digitCoverAllImageTest	Oczekiwany rezultat	Oczekuje, że gettery poprawnie stworzonego obiektu zwróca wartości równe oczekiwany

Opis	Sprawdzam przypadek, w którym cyfry wypełniają całą macierz. Następnie jak w przypadku ID 19.1 sprawdzam gettery czy zwracają poprawne wartości	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

20. SizeCellExtractStrategyTest

```
package pl.sudokusolver.recognizerlib.extractors.cells;
```

ID 20.1			
Tytuł	cellExtractionShouldFailed Test	Oczekiwany rezultat	Rzucenie wyjątku przez funkcję extract - wyjątek: CellExtractionsFailedException
Opis	W tym teście podaje zbyt małe macierze do rozdzielania, więc powinien zostać rzucony wyjątek, który znajduje się w package pl.sudokusolver.recognizerlib.exceptions;	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 20.2			
Tytuł	cellExtractionsTest	Oczekiwany rezultat	Oczekujemy, że została stworzona poprawna ilość mniejszych macierzy to jest 81 dla naszego przypadku oraz każda macierz będzie równa macierzy jedynek 10x10

Opis	Funkcja extract zwraca listę macierzy, które są kawałkami o równym rozmiarze większej macierzy - wszystkie możliwe podziały macierzy na macierze 10x10. W tym przypadku dzielimy macierz samych jedynek 90x90 na kawałki,	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV - wejściowa macierz musi być prostokątem	Kroki	---

21. FastDigitExtractStrategyTest

```
package pl.sudokusolver.recognizerlib.extractors.digits;
```

ID 21.1			
Tytuł	digitNotFound	Oczekiwany rezultat	Funkcja isPresent dla obiektu opcjonalnego powinna zwrócić false
Opis	Obiekt nie powinien zostać stworzony, bo podana macierz zawiera same zera	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 21.2			
Tytuł	digitCoverAllImage	Oczekiwany rezultat	Oczekuje, że gettery zwróca wartości równe oczekiwanym oraz, że funkcja isPresent dla obiektu opcjonalnego zwróci true
Opis	Sprawdzam czy dla poprawnie stworzonego obiektu przez funkcję getDigitBox zostaną zwrócone poprawne gettery	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

22. DefaultGridExtractStrategyTest

```
package pl.sudokusolver.recognizerlib.extractors.grid;
```

ID 22.1			
Tytuł	extractFromImg	Oczekiwany rezultat	Rozmary dwóch macierzy powinny być identyczne, wartości dwóch macierzy w każdej komórce też powinny być identyczne

Opis	Test polega na sprawdzeniu czy macierz jest prawidłowo tworzona/wycinana ze zdjęcia. W celu sprawdzenia porównuje macierz po użyciu funkcji extract z macierzą oczekiwaną. Test przeprowadzam dwa razy dla dwóch zdjęć wejściowych i dwóch oczekiwanych, żeby mieć pewność poprawności działania	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV - zdjęcia wejściowe i wyjściowe, znajdują się w resources	Kroki	---

23. ANNTest

```
package pl.sudokusolver.recognizerlib.ocr.ml;
```

ID 23.1

Tytuł	rec	Oczekiwany rezultat	Dla każdego obrazka prawidłowo rozpoznana cyfra
Opis	Test sprawdza czy udało się stworzyć obiekt ANN oraz co najważniejsze sprawdza rozpoznawanie cyferek dla pięciu zdjęć, każde zdjęcie zawiera jedną cyfrę	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV - zdjęcia wejściowe i wyjściowe, znajdują się w resources/rectest	Kroki	---

24. SVMTest

```
package pl.sudokusolver.recognizerlib.ocr.ml;
```

ID 24.1			
Tytuł	rec	Oczekiwany rezultat	Dla każdego obrazka prawidłowo rozpoznana cyfra
Opis	Test sprawdza czy udało się stworzyć obiekt SVM oraz sprawdza rozpoznawanie cyferek dla pięciu zdjęć, każde zdjęcie zawiera jedną cyfrę	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV - zdjęcia wejściowe i wyjściowe, znajdują się w resources/rectest	Kroki	---

25. TesseractSimpleTest

```
package pl.sudokusolver.recognizerlib.ocr.tesseract;
```

ID 25.1			
Tytuł	rec	Oczekiwany rezultat	Dla każdego obrazka prawidłowo rozpoznana cyfra
Opis	Test sprawdza czy udało się stworzyć obiekt TesseractSimple oraz sprawdza rozpoznawanie cyferek dla pięciu zdjęć, każde zdjęcie zawiera jedną cyfrę	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV - zdjęcia wejściowe i wyjściowe, znajdują się w resources/rectest	Kroki	---

26. TesseractSingletonWrapperTest

```
package pl.sudokusolver.recognizerlib.ocr.tesseract;
```

ID 26.1			
Tytuł	goodLoadTesseract	Oczekiwany rezultat	Udało się stworzyć obiekt za pomocą singletonu, sprawdzamy czy stworzony obiekt nie jest nullem
Opis	Test sprawdza czy prawidłowo został utworzony obiekt Tesseract, został opakowany przez klasę TesseractSingletonWrapper	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

27.InitTest

ID 27.1			
Tytuł	checkMainResource	Oczekiwany rezultat	Oczekujemy, że 4 pliki istnieją, a funkcja Init.class.getResource zawsze zwróci coś innego niż null
Opis	Test sprawdza czy w resources znajdują się potrzebne pliki	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV - 4 pliki resource	Kroki	---

```
pl.sudokusolver.server.
```

28. DigitRecoginzerTest

```
package pl.sudokusolver.server.bean;
```

ID 28.1			
Tytuł	checkIfDigitRecognizerLoadCorrectly	Oczekiwany rezultat	Oczekujemy zwrócenia true przy odpowiednim trybie
Opis	Sprawdzamy czy getRecoginzer ładuje odpowiedni tryb	Poprawność wykonania	Ok

Warunek wstępny	ExtendWith(SpringExtension.class) @WebAppConfiguration() @ContextConfiguration(classes = {WebConfig.class})	Kroki	---
------------------------	--	--------------	-----

29.ApiControllerTest

```
package pl.sudokusolver.server.controllers;
```

ID 29.1			
Tytuł	loggerIsSet	Oczekiwany rezultat	Oczekujemy zwrócenia true przy odpowiednim trybie

Opis	Sprawdzamy czy logger jest ustawiony	Poprawność wykonania	Ok
Warunek wstępny	ExtendWith(SpringExtension.class) @WebAppConfiguration() @ContextConfiguration(classes = {WebConfig.class}) @Autowired	Kroki	---

ID 29.2			
Tytuł	solveExceptions	Oczekiwany rezultat	Oczekujemy zwrócenia true przy odpowiednim trybie
Opis	Uruchamiam sudoku i je rozwiązuje	Poprawność wykonania	Ok
Warunek wstępny	ExtendWith(SpringExtension.class) @WebAppConfiguration() @ContextConfiguration(classes = {WebConfig.class}) @Autowired	Kroki	---

ID 29.3			
Tytuł	solveOk	Oczekiwany rezultat	Po rozwiązaniu sudoku nie powinny w nim wystąpić zera, a responseStatus powinien być równy 1
Opis	Sprawdzamy czy sudoku zostało poprawnie rozwiązane	Poprawność wykonania	Ok
Warunek wstępny	ExtendWith(SpringExtension.class) @WebAppConfiguration() @ContextConfiguration(classes = {WebConfig.class}) @Autowired	Kroki	---

ID 29.4			
Tytuł	photoNoOk	Oczekiwany rezultat	Status po wykonaniu zadań 400 oraz rzucenie wyjątku
Opis	Sprawdzamy za pomocą mockMVC .perform czy status po wykonaniu funkcji jest równy 400 oraz sprawdzamy rzucenie wyjątku	Poprawność wykonania	Ok
Warunek wstępny	ExtendWith(SpringExtension.class) @WebAppConfiguration() @ContextConfiguration(classes = {WebConfig.class}) @Autowired	Kroki	---

ID 29.5			
Tytuł	photoOk	Oczekiwany rezultat	Status po wykonaniu zadań 200 OkResponse.status = 1
Opis	Sprawdzamy funkcje i ich działanie, jeżeli wyślemy poprawne zdjęcie sudoku na server	Poprawność wykonania	Ok
Warunek wstępny	ExtendWith(SpringExtension.class) @WebAppConfiguration() @ContextConfiguration(classes = {WebConfig.class}) @Autowired	Kroki	---

30. MainControllerTest

```
package pl.sudokusolver.server.bean;
```

ID 30.1			
Tytuł	homePage	Oczekiwany rezultat	Po wykona wykonaniu zadań oczekiwany status ==status().isOk()
Opis	Oczekuje poprawnie wczytanej strony głównej	Poprawność wykonania	Ok
Warunek wstępny	ExtendWith(SpringExtension.class) @WebAppConfiguration() @ContextConfiguration(classes = {WebConfig.class}) @InjectMocks MainController controller; MockMvc mockMvc;	Kroki	---

31. ErrorResponseTest

```
package pl.sudokusolver.server.data;
```

ID 31.1			
Tytuł	getErrorMessage	Oczekiwany rezultat	Zwrócenie errorMessage użytego podczas tworzenia obiektu klasy ErrorResponse
Opis	Sprawdza czy funkcja getErrorMessage poprawnie zwraca message	Poprawność wykonania	Ok
Warunek wstępny	brak	Kroki	---

ID 31.2			
Tytuł	setErrorMessage	Oczekiwany rezultat	Zwrócenie poprawnego errorMessage
Opis	Sprawdzanie czy funkcja setErrorMessage poprawnie ustawia errorMessage	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

32. GridModelTest

```
package pl.sudokusolver.server.models;
```

ID 32.1			
Tytuł	constructorTest	Oczekiwany rezultat	Pola po wywołaniu konstruktora klasy GridModel będą miały oczekiwane wartości
Opis	Sprawdza czy GridModel jest poprawnie tworzony dla dwóch macierzy	Poprawność wykonania	Ok
Warunek wstępny	Brak	Kroki	---

TESTY METOD PRYWATNYCH

33. MNISTReaderPrivateTest

```
package pl.sudokusolver.recognizerlib.data;
```

ID 33.1

Tytuł	checkIfSizelsCorrectlySet	Oczekiwany rezultat	getDeclaredField("size") powinno zwrócić 28 dla klasy MNISTReader
Opis	Sprawdzamy czy pole size jest równe 28 - tak powinno być ustawione. Pole to jest prywatne dlatego używamy funkcji setAccesible	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 33.2			
Tytuł	createLabel	Oczekiwany rezultat	Liczba kolumn powinna zawsze wynosić 1, liczba wierszy powinna być równa size oraz typ też powinien zostać zwracany taki jak oczekiwany przez testera
Opis	W tym teście ustawiamy dostęp do metody createLabel na true przez setAccesible. Używamy createLabel dla trzech przypadków i sprawdzamy kolumny, wiersze i typ dla każdego	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 33.3

Tytuł	createLabelsFromString	Oczekiwany rezultat	Metoda labels.invoke powinna zwrócić wartości oczekiwane przez testera
Opis	Sprawdzamy czy label jest poprawnie tworzony z Stringa, ustawiamy dostęp do funkcji prywatnej getStringFromLabels na true	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 33.4			
Tytuł	putLabel	Oczekiwany rezultat	Dwa gettery powinny zwracać wartości, które ustawiliśmy za pomocą putLabel dla danych kolumn i wierszy
Opis	Ustawiamy dostęp do metody prywatnej putLabel na true, sprawdzamy jej działanie dla macierzy	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 33.5			
Tytuł	putImg	Oczekiwany rezultat	Po użyciu metody put.invoke() dla macierzy utworzonej przez testera i dla danych rozmiarów i innych danych, wartości wszystkich kolumn otrzymanej macierzy powinny być równe wartościom oczekiwany
Opis	Ustawiamy dostęp do prywatnej metody putImg na true, sprawdzam jej działanie	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

34. DefaultGridExtractStrategyTest

```
package pl.sudokusolver.recognizerlib.extractors.grid;
```

ID 34.1			
Tytuł	getBiggestBlob	Oczekiwany rezultat	Po użyciu funkcji blob.invoke(defaultGridExtractStrategy, contours), getBiggestBlob powinien zwrócić oczekiwane wartości przez testera. Testujemy na dwa przypadki, gdzie oczekiwana wartość jest równa 1, w drugim -1
Opis	Ustawiamy dostęp do metody getBiggestBlob na true, tworzę listę obiektów MatOfPoint, nazwałem ją contours	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

ID 34.2			
Tytuł	calcApprox	Oczekiwany rezultat	Dla niepoprawnych danych powinien zostać złapany wyjątek, dla danych poprawnych powinny zostać zwrócone dane oczekiwane przez testera - sprawdzane na dwóch testach
Opis	Ustawiamy dostęp do calcApprox na true, sprawdzamy czy dla złych danych łapany jest wyjątek, sprawdzam czy funkcja calcApprox	Poprawność wykonania	Ok

Warunek wstępny	-załadowanie OpenCV	Kroki	---
------------------------	---------------------	--------------	-----

ID 34.3			
Tytuł	prospectiveWrap	Oczekiwany rezultat	Robimy dwa testy, w drugim oczekujemy rzucenia wyjątku dla niepoprawnych danych, w pierwszym sprawdzamy czy macierz została odpowiednio przekształcona według działania transformacji geometrycznej dla danych punktów i macierzy. W tym celu sprawdzamy uzyskaną macierz z macierzą oczekiwaną napisaną ręcznie.
Opis	https://docs.opencv.org/3.1.0/da/d6e/tutorial_py_geometric_transformations.html w linku powyżej opisane działanie funkcji, sprawdzamy czy metoda prospectiveWrap rzuca wyjątek, oraz sprawdzamy czy daje oczekiwany wynik	Poprawność wykonania	Ok
Warunek wstępny	-załadowanie OpenCV	Kroki	---

PRZYPADKI TESTOWE DLA APP – TESTY JEDNOSTKOWE

1. RecognitionTest

```
package pl.sudokusolver.app.CustomViews.RadioButtons
```

ID 1.1			
Tytuł	getValue	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok

Warunek wstępny	---	Kroki	---
ID 1.2			
Tytuł	setValue	Oczekiwany rezultat	Udało się ustawić zadaną wartość
Opis	Test sprawdza poprawność settera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

2. ScalingTest

```
package pl.sudokusolver.app.CustomViews.RadioButtons
```

ID 2.1			
Tytuł	getValue	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 2.2			

Tytuł	setValue	Oczekiwany rezultat	Udało się ustawić zadaną wartość
Opis	Test sprawdza poprawność settera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

3. StrictModeTest

```
package pl.sudokusolver.app.CustomViews.RadioButtons
```

ID 3.1			
Tytuł	getValue	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 3.2			

Tytuł	setValue	Oczekiwany rezultat	Udało się ustawić zadaną wartość
Opis	Test sprawdza poprawność settera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

4. BlurBlockSizeTest

```
package pl.sudokusolver.app.CustomViews.Sliders
```

ID 4.1			
Tytuł	getValue	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 4.2			

Tytuł	setValue	Oczekiwany rezultat	Udało się ustawić zadaną wartość
Opis	Test sprawdza poprawność settera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

5. BlurCTest

```
package pl.sudokusolver.app.CustomViews.Sliders
```

ID 5.1			
Tytuł	getValue	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 5.2			

Tytuł	setValue	Oczekiwany rezultat	Udało się ustawić zadaną wartość
Opis	Test sprawdza poprawność settera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

6. BlurSizeTest

```
package pl.sudokusolver.app.CustomViews.Sliders
```

ID 6.1			
Tytuł	getValue	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 6.2			

Tytuł	setValue	Oczekiwany rezultat	Udało się ustawić zadaną wartość
Opis	Test sprawdza poprawność settera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

7. DistanceTest

```
package pl.sudokusolver.app.CustomViews.Sliders
```

ID 7.1			
Tytuł	getValue	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 7.2			

Tytuł	setValue	Oczekiwany rezultat	Udało się ustawić zadaną wartość
Opis	Test sprawdza poprawność settera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

8. GaussianBlurTest

```
package pl.sudokusolver.app.CustomViews.Sliders
```

ID 8.1			
Tytuł	getValue	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 8.2			

Tytuł	setValue	Oczekiwany rezultat	Udało się ustawić zadaną wartość
Opis	Test sprawdza poprawność settera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

9. LineGapTest

```
package pl.sudokusolver.app.CustomViews.Sliders
```

ID 9.1			
Tytuł	getValue	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 9.2			

Tytuł	setValue	Oczekiwany rezultat	Udało się ustawić zadaną wartość
Opis	Test sprawdza poprawność settera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

10. LineThicknessTest

```
package pl.sudokusolver.app.CustomViews.Sliders
```

ID 10.1			
Tytuł	getValue	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 10.2			

Tytuł	setValue	Oczekiwany rezultat	Udało się ustawić zadaną wartość
Opis	Test sprawdza poprawność settera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

11. LineTresholdTest

```
package pl.sudokusolver.app.CustomViews.Sliders
```

ID 11.1			
Tytuł	getValue	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 11.2			

Tytuł	setValue	Oczekiwany rezultat	Udało się ustawić zadaną wartość
Opis	Test sprawdza poprawność settera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

12.MinLineSizeTest

```
package pl.sudokusolver.app.CustomViews.Sliders
```

ID 12.1			
Tytuł	getValue	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 12.2			

Tytuł	setValue	Oczekiwany rezultat	Udało się ustawić zadaną wartość
Opis	Test sprawdza poprawność settera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

13. ProgingTest

```
package pl.sudokusolver.app.CustomViews.Sliders
```

ID 13.1			
Tytuł	getValue	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 13.2			

Tytuł	setValue	Oczekiwany rezultat	Udało się ustawić zadaną wartość
Opis	Test sprawdza poprawność settera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

14. CanvasTest

```
package pl.sudokusolver.app
```

ID 14.1			
Tytuł	getInitial	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 14.2			

Tytuł	clear	Oczekiwany rezultat	Elementy tablic zostają ustawione na „0”
Opis	Test sprawdza poprawność clear (czyszczenia tablic z danych)	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 14.3			
Tytuł	getOffsetY	Oczekiwany rezultat	Udało się otrzymać oczekiwaną wartość
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 14.4			
Tytuł	modifySolution	Oczekiwany rezultat	Dane zostały zastąpione
Opis	Test sprawdza zmienianie danych w tablicy (poprawnie nadpisywanie)	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 14.5			
Tytuł	modifyInitial	Oczekiwany rezultat	Dane zostały zastąpione
Opis	Test sprawdza zmienianie danych w tablicy (poprawnie nadpisywanie)	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 14.6			
Tytuł	onValueInserted	Oczekiwany rezultat	Poprawność wprowadzenia prawidłowych danych w przeciwnym wypadku rzucenie wyjątku

Opis	Test sprawdza czy możemy wprowadzić wartość do sudoku	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

15. ControlsTest

```
package pl.sudokusolver.app
```

ID 15.1			
Tytuł	onKeyPressed_Numpad	Oczekiwany rezultat	Odpowiedni klawisz wywołuje odpowiedni warunek

Opis	Test sprawdza poprawność działania funkcji onKeyPressed dla klawiszów wprowadzonych z klawiatury numerycznej	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 15.2			
Tytuł	onKeyPressed_Digit	Oczekiwany rezultat	Odpowiedni klawisz wywołuje odpowiedni warunek
Opis	Test sprawdza poprawność działania funkcji onKeyPressed dla klawiszów wprowadzonych z użyciem standardowej klawiatury	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 15.3			
Tytuł	onKeyPressed_BackSpace	Oczekiwany rezultat	Odpowiedni klawisz wywołuje odpowiedni warunek
Opis	Test sprawdza poprawność działania funkcji onKeyPressed dla BackSpace w tym przypadku wywołania funkcji wymazującej cyfrę	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 15.4			
Tytuł	onKeyPressed_Delete	Oczekiwany rezultat	Odpowiedni klawisz wywołuje odpowiedni warunek
Opis	Test sprawdza poprawność działania funkcji onKeyPressed dla Delete w tym przypadku wywołania funkcji wymazującej cyfrę	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

ID 15.3			
Tytuł	onKeyPressed_Escape	Oczekiwany rezultat	Odpowiedni klawisz wywołuje odpowiedni warunek
Opis	Test sprawdza poprawność działania funkcji onKeyPressed dlaEscape w tym przypadku wywołania funkcji wymazującej cyfrę	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 15.4			
Tytuł	onKeyPressed_Moving	Oczekiwany rezultat	Odpowiedni klawisz wywołuje odpowiedni warunek
Opis	Test sprawdza poprawność działania funkcji onKeyPressed dla strzałek na klawiaturze w tym przypadku wywołania funkcji poruszających się w odpowiednie pola sudoku	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 15.5			
Tytuł	onKeyPressed_Moving_2	Oczekiwany rezultat	Odpowiedni klawisz wywołuje odpowiedni warunek
Opis	Test sprawdza poprawność działania funkcji onKeyPressed dla WSAD(analogicznie jak strzałki) na klawiaturze w tym przypadku wywołania funkcji poruszających się w odpowiednie pola sudoku	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

16. SingletonTest

```
package pl.sudokusolver.app
```

ID 16.1			
Tytuł	isBlocked	Oczekiwany rezultat	Funkcja zwraca oczekiwaną wartość
Opis	Test sprawdza poprawność działania gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 16.2			
Tytuł	block	Oczekiwany rezultat	Funkcja ustawia oczekiwaną wartość (w tym przypadku true)
Opis	Test sprawdza działanie block który zmienia wartość pola	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 16.3			
Tytuł	unblock	Oczekiwany rezultat	Funkcja ustawia oczekiwaną wartość (w tym przypadku false)
Opis	Test sprawdza działanie unblock który zmienia wartość pola	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 16.3			
Tytuł	unblock_block	Oczekiwany rezultat	Funkcja ustawia oczekiwaną wartość (w tym przypadku true a następnie false)
Opis	Test sprawdza działanie zarówno unblock jak i block w postaci kombinacji zmieniając dwa razy wartość pola	Poprawność wykonania	Ok

Warunek wstępny	---	Kroki	---
------------------------	-----	--------------	-----

17. UtilitiesTest

```
package pl.sudokusolver.app
```

ID 17.1			
Tytuł	isJavaValid	Oczekiwany rezultat	Poprawne sprawdzenie wersji javy
Opis	Test sprawdza poprawność zwracania wartości dla wersji javy	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 17.2			
Tytuł	getError	Oczekiwany rezultat	Poprawne errorry dla danej liczby
Opis	Test sprawdza poprawność zwracania errorów dla odpowiednich numerów	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 17.3			
Tytuł	getFileExtension	Oczekiwany rezultat	Poprawne odczytanie rozszerzenia dla pliku. W przeciwnym wypadku rzucenie wyjątku
Opis	Test sprawdza poprawność odczytywania rozszerzenia pliku. Gdy nie jest to możliwe rzuca wyjątek	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

18. GameBoardTest

```
package pl.sudokusolver.app
```

ID 18.1			
Tytuł	getInitial	Oczekiwany rezultat	Zwrócenie oczekiwanej wartości
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.2			
Tytuł	getSolution	Oczekiwany rezultat	Zwrócenie oczekiwanej wartości
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.3			
Tytuł	clear	Oczekiwany rezultat	Tablice mają wszystkie elementy zerowe
Opis	Test sprawdza poprawność clear (ustawienia elementów tablic na wartości 0).	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.4			

Tytuł	modifySolution	Oczekiwany rezultat	Elementy zostają nadpisane w poprawny sposób
Opis	Test sprawdza poprawność nadpisywania elementów poprzez zadaną tablicę elementów	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.5			
Tytuł	modifyInitial	Oczekiwany rezultat	Elementy zostają nadpisane w poprawny sposób
Opis	Test sprawdza poprawność nadpisywania elementów poprzez zadaną tablicę elementów	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.6			
Tytuł	modifyInitial1	Oczekiwany rezultat	Elementy zostają nadpisane w poprawny sposób
Opis	Test sprawdza poprawność nadpisywania elementów poprzez zadaną kolumnę, wiersz i wartość	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

19. ParametersTest

```
package pl.sudokusolver.app
```

ID 18.1			
Tytuł	set	Oczekiwany rezultat	Ustawienie odpowiedniej wartości dla odpowiednich pól
Opis	Test sprawdza poprawność setera (przez kopiowanie obiektu)	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.2			
Tytuł	set	Oczekiwany rezultat	Ustawienie odpowiedniej wartości dla odpowiednich pól
Opis	Test sprawdza poprawność setera (przez podanie kolejno wartości dla każdego pola)	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.3			
Tytuł	getLineTreshsold	Oczekiwany rezultat	Zwrócenie oczekiwanej wartości

Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.4			
Tytuł	getLineGap	Oczekiwany rezultat	Zwrócenie oczekiwanej wartości
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.5			
Tytuł	getMinLineSize	Oczekiwany rezultat	Zwrócenie oczekiwanej wartości
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.6			
Tytuł	getBlurSize	Oczekiwany rezultat	Zwrócenie oczekiwanej wartości
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.7			
Tytuł	getBlurBlockSize	Oczekiwany rezultat	Zwrócenie oczekiwanej wartości
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.8			
Tytuł	getBlurC	Oczekiwany rezultat	Zwrócenie oczekiwanej wartości

Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.9			
Tytuł	getScaling	Oczekiwany rezultat	Zwrócenie oczekiwanej wartości
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.10			
Tytuł	getRecognition	Oczekiwany rezultat	Zwrócenie oczekiwanej wartości
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.11			
Tytuł	getBlurC	Oczekiwany rezultat	Zwrócenie oczekiwanej wartości
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---
ID 18.12			
Tytuł	getStrictMode	Oczekiwany rezultat	Zwrócenie oczekiwanej wartości
Opis	Test sprawdza poprawność gettera	Poprawność wykonania	Ok
Warunek wstępny	---	Kroki	---

1.2 Raporty o błędach.

Wszystkie zgłoszone błędy zostały naprawione.

Raport o błędzie

ID	I_01	Nr commitu	c9d7a0a
Tytuł	Błędna konfiguracja loggera	Oczekiwany rezultat	System logów oparty o log4j
Priorytet	Mały	Rzeczywisty rezultat	Framework nie przechwytyuje logów
Warunki wstępne	---	Kroki	---

Raport o błędzie

ID	I_02	Nr commitu	9fd22eb
Tytuł	Brak wsparcia dla debiana	Oczekiwany rezultat	Poprawne wczytanie openCV
Priorytet	Bloker	Rzeczywisty rezultat	Błąd podczas wczytywania
Warunki wstępne	Wymagany system debian 9	Kroki	---

Raport o błędzie

ID	I_03	Nr commitu	97fc474
Tytuł	Błędna implementacja buildera dla SudokuExtractora	Oczekiwany rezultat	Stworzenie obiektu klasy BaseSudokuExtractor za pomocą buildera
Priorytet	Średni	Rzeczywisty rezultat	Obiekt nie tworzy się poprawnie
Warunki wstępne	---	Kroki	---

Raport o błędzie

ID	I_04	Nr commitu	dc2ec0a
Tytuł	Niepoprawne działanie MaxResizeFilter	Oczekiwany rezultat	Filter ma za zadanie przeskalować obraz zachowując proporcje
Priorytet	Średni	Rzeczywisty rezultat	Brak widocznego rezultatu
Warunki wstępne	---	Kroki	---

Raport o błędzie

ID	I_05	Nr commitu	c3f2a5f
Tytuł	SmartSolver nie czyści się automatycznie	Oczekiwany rezultat	Obiekt potrafi rozwiązać więcej niż jedno sudoku
Priorytet	Średni	Rzeczywisty rezultat	Występuje NullPointerException
Warunki wstępne	---	Kroki	1. Rozwiązać sudoku 2. Ponowna próba rozwiązania sudoku kończy się niepowodzeniem

Raport o błędzie

ID	I_07	Nr commitu	7e52d13
Tytuł	Błędny nazwa wyświetlanego błędu	Oczekiwany rezultat	Wyświetlenie błędu mówiącego, że nie można tak zrobić
Priorytet	Bloker	Rzeczywisty rezultat	Wyświetlenie błędu z nieporównanym opisem
Warunki wstępne	Wpisanie wartości w danym rzędzie/ kolumnie/ kwadracie sudoku i próba wpisania kolejnego takiego samego	Kroki	---

Raport o błędzie

ID	I_08	Nr commitu	e4799f8
Tytuł	Brak wyświetlenia błędu kiedy serwer nie zwracał poprawnej odpowiedzi	Oczekiwany rezultat	Wyświetlenie okienka z błędem
Priorytet	Bloker	Rzeczywisty rezultat	Brak jakiegokolwiek informacji o błędzie
Warunki wstępne	Niedziałający serwer i wysłanie do niego requesta	Kroki	---

Raport o błędzie

ID	I_11	Nr commitu	29ee653
Tytuł	Niepoprawne działanie LOAD	Oczekiwany rezultat	Po anulowaniu próby załadowania zdjęcia, a następnie ponowieniu czynności, przycisk LOAD powinien reagować i umożliwić próbę załadowania zdjęcia
Priorytet	Wysoki	Rzeczywisty rezultat	Po anulowaniu próby załadowania zdjęcia, a następnie ponowieniu czynności przycisk LOAD nie reaguje
Warunki wstępne	---	Kroki	<ol style="list-style-type: none"> 1. Anulować próbę załadowania zdjęcia 2. Ponownie próbować załadować zdjęcie

Raport o błędzie

ID	I_12	Nr commitu	29ee653
Tytuł	Niepoprawne działanie SOLVE podczas braku połączenia z serwerem	Oczekiwany rezultat	Komunikat o braku połączenia
Priorytet	Średni	Rzeczywisty rezultat	Brak jakiejkolwiek reakcji
Warunki wstępne	---	Kroki	

Raport o błędzie

ID	I_13	Nr commitu	7e52d13
Tytuł	Brak możliwości przywrócenia 'pustej komórki'	Oczekiwany rezultat	Po wpisaniu wartości do danej komórki próba usunięcia klawiszem „backspace” powinna zakończyć się powodzeniem
Priorytet	Wysoki	Rzeczywisty rezultat	Error w postaci powtórzenia wartości w kolumnie/wierszu/ kwadracie 3x3
Warunki wstępne	---	Kroki	1.Wpisanie wartości do komórki 2. Próba usunięcia ich za pomocą „backspace”

Raport o błędzie

ID	I_14	Nr commitu	6e7f784
Tytuł	Brak wykrywania powtarzających się wartości w wierszach/ kolumnach	Oczekiwany rezultat	Po wpisaniu tej wartości do danego wiersza / kolumny otrzymujemy error
Priorytet	Wysoki	Rzeczywisty rezultat	Możliwość wpisania tych samych wartości do kolumny/wiersza
Warunki wstępne	---	Kroki	1. Wpisanie wartości w rogach sudoku odpowiednio: Lewy dolny, Prawy dolny, Lewy górny, Prawy górny.

Raport o błędzie

ID	I_15	Nr commitu	926dc73
Tytuł	Brak komunikatu podczas próby rozwiązania niepoprawnie wczytanego sudoku	Oczekiwany rezultat	Podczas wciśnięcia SOLVE komunikat z errorem o niepoprawnych danych
Priorytet	Średni	Rzeczywisty rezultat	Próba rozwiązania niepoprawnego sudoku.
Warunki wstępne	---	Kroki	

Raport o błędzie

ID	I_16	Nr commitu	029e124
Tytuł	Błędny komunikat podczas próby rozwiązywania sudoku ze złymi danymi wejściowymi	Oczekiwany rezultat	Podczas kliknięcia SOLVE przy złych danych wejściowych oczekujemy na error błędnych danych
Priorytet	Niski	Rzeczywisty rezultat	Podczas kliknięcia SOLVE przy złych danych wejściowych otrzymujemy error w postaci „Couldn't open file”
Warunki wstępne	---	Kroki	1.Wczytanie zdjęcia ze złymi danymi 2.Kliknięcie przycisku SOLVE

Raport o błędzie

ID	I_17	Nr commitu	029e124
Tytuł	Błędny komunikat podczas nie znalezienia sudoku na zdjęciu	Oczekiwany rezultat	Po załadowaniu zdjęcia w słabej jakości lub bez sudoku przyciskiem LOAD otrzymujemy error informujący o braku sudoku na zdjęciu
Priorytet	Niski	Rzeczywisty rezultat	Podczas załadowania zdjęcia w słabej jakości lub bez sudoku otrzymujemy error: „Couldn't file open”
Warunki wstępne	---	Kroki	1.Wczytanie zdjęcia ze w słabej jakości lub bez sudoku

Raport o błędzie

ID	I_18	Nr commitu	0fce865
Tytuł	Brak komunikatu, że został załadowany do sekcji Image uszkodzony plik.	Oczekiwany rezultat	Po załadowaniu uszkodzonego pliku do sekcji Image za pomocą przycisku LOAD otrzymujemy błąd związany z uszkodzonym plikiem
Priorytet	Niski	Rzeczywisty rezultat	Możemy załadować uszkodzony plik do programu
Warunki wstępne	---	Kroki	1.Wczytanie uszkodzonego pliku .jpg lub .png

Raport o błędzie

ID	I_19	Nr commitu	34226b3
Tytuł	Ogromny memory leak	Oczekiwany rezultat	Pamięć po przetworzeniu zdjęcia pozostaje zwolniona.
Priorytet	Krytyczny	Rzeczywisty rezultat	Po przetworzeniu przez serwer dużej ilość zdjęć pamięć ram kończy się.
Warunki wstępne	---	Kroki	---

Raport o błędzie

ID	I_20	Nr commitu	34226b3
Tytuł	Błędne wczytywanie danych MNIST	Oczekiwany rezultat	Etykiety 0 zostaną pominięte.
Priorytet	Ważne	Rzeczywisty rezultat	Etykiety 0 znajdują się w danych wyjściowych.
Warunki wstępne	---	Kroki	---

Raport o błędzie

ID	I_21	Nr commitu	634a8ad
Tytuł	ToGrayFilter rzuca wyjątek gdy zdjęcie wejściowe ma już 1 kanał.	Oczekiwany rezultat	ToGrayFilter radzi sobie z zdjęciami z jednym kanałem.
Priorytet	Niski	Rzeczywisty rezultat	Zostaje rzucony wyjątek z openCV.
Warunki wstępne	---	Kroki	---

Raport o błędzie

ID	I_21	Nr commitu	c4d70a1
Tytuł	Ann błędna interpretacja wyników.	Oczekiwany rezultat	Wyniki zwrócone przez ann są zgodne z tym co zwraca model z openCV.
Priorytet	Wysoki	Rzeczywisty rezultat	Wyniki zawsze różnią się o 1.
Warunki wstępne	---	Kroki	---

2 Testy systemowe.

Scenariusze i przypadki testowe zostały opisane według wzoru:

Scenariusz Testowy	
ID	Typ
Opis	
Czynności przygotowawcze	
Czynności końcowe	
Przypadki testowe	
Lp.	Oczekiwany rezultat
Tytuł	
Priorytet	Kroki
Warunki wstępne	

2.1 Testy poszczególnych scenariuszy przypadków użycia.

Scenariusz Testowy

ID	PU1	Typ	testy funkcyjne
Opis	Wczytanie obrazu		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny 2.Użytkownik rozpoczyna proces wczytania zdjęcia poprzez kliknięcie przycisku „Wczytaj” 3.Aplikacja wyświetla formularz wyboru (domyślny systemowy). 4.Użytkownik wybiera plik (.jpg lub .png) 5.Aplikacja wyświetla formularz ze zdjęcie wybranym przez użytkownika(podgląd) 6.Użytkownik zatwierdza wybór zdjęcia poprzez kliknięcie przycisku „Zatwierdź”.		
Czynności końcowe	Użytkownik wyświetla na ekranie formularz główny aplikacji - podgląd wczytanego zdjęcia, odpowiednio wczytanego sudoku.		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Załadowanie poprawnego zdjęcia (sudoku bez błędnych danych)	Sudoku zostanie poprawnie przepisane, a na podglądzie pojawi się wczytane zdjęcie	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Połączenie z serwerem	---	OK
Lp.	2	Oczekiwany rezultat	
Tytuł	Załadowanie niepoprawnego zdjęcia (sudoku z błędnymi danymi np. te same liczby w wierszu)	Sudoku zostanie poprawnie przepisane, powtórzenia zaznaczone na kolor czerwony, a na podglądzie pojawi się wczytane zdjęcie	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Połączenie z serwerem	---	OK
Lp.	3	Oczekiwany rezultat	

Tytuł	Załadowanie niepoprawnego zdjęcia (bez sudoku)	Aplikacja wyświetla komunikat: „Nie można znaleźć sudoku”, zdjęcie zostaje na podglądzie	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Połączenie z serwerem	---	OK
Lp.	4	Oczekiwany rezultat	
Tytuł	Załadowanie poprawnego zdjęcia (sudoku bez błędnych danych)	Aplikacja wyświetla komunikat: „Nie udało się połączyć z serwerem”	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Brak połączenia z serwerem	---	OK
Lp.	5	Oczekiwany rezultat	
Tytuł	Załadowanie niepoprawnego zdjęcia (sudoku z błędnymi danymi np. te same liczby w wierszu)	Aplikacja wyświetla komunikat: „Nie udało się połączyć z serwerem”	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Brak połączenia z serwerem	---	OK
Lp.	6	Oczekiwany rezultat	
Tytuł	Załadowanie niepoprawnego zdjęcia (bez sudoku)	Aplikacja wyświetla komunikat: „Nie udało się połączyć z serwerem”	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Brak połączenia z serwerem	---	OK

Scenariusz Testowy

ID	PU2	Typ	testy akceptacyjne
Opis	Wczytanie nieobsługiwanego formatu pliku lub ścieżki do pliku nieistniejącego.		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny 2.Rozpoczęcie procesu wczytania zdjęcia przez kliknięcie przycisku „Wczytaj” 3.Aplikacja wyświetla formularz wyboru(domyślny systemowy) 4.Wybór pliku lub ścieżki do pliku		
Czynności końcowe	Użytkownik wyświetla na ekranie formularz główny aplikacji		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Wczytanie nieobsługiwanego formatu pliku	Aplikacja wyświetla błąd E002	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Brak wcześniej wczytanego zdjęcia w formularzu głównym	1.Kliknięcie przycisku „Wczytaj” 2.Wybór ścieżki do nieobsługiwanego formatu pliku	OK
Lp.	2	Oczekiwany rezultat	
Tytuł	Wczytanie nieobsługiwanego formatu pliku	Aplikacja wyświetla błąd E002	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Wcześniej zostało wczytane zdjęcie do formularza głównego	1.Kliknięcie przycisku „Wczytaj” 2.Wybór ścieżki do nieobsługiwanego formatu pliku	OK
Lp.	3	Oczekiwany rezultat	
Tytuł	Wczytanie pliku nieistniejącego	Aplikacja wyświetla błąd E003	

Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Brak wcześniej wczytanego zdjęcia w formularzu głównym	1.Kliknięcie przycisku „Wczytaj” 2.Wybór ścieżki do pliku nieistniejącego	OK
Lp.	4	Oczekiwany rezultat	
Tytuł	Wczytanie pliku nieistniejącego	Aplikacja wyświetla błąd E003	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Wcześniej zostało wczytane zdjęcie do formularza głównego	1.Kliknięcie przycisku „Wczytaj” 2.Wybór ścieżki do pliku nieistniejącego	OK

Scenariusz Testowy

ID	PU10	Typ	testy funkcyjne
Opis	Edycja zdjęcia		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny 2.Użytkownik rozpoczyna proces wczytywanie zdjęcia przez kliknięcie „Wczytaj” 3.Aplikacja wyświetla formularz wyboru (domyślny systemowy) 4.Użytkownik wybiera plik .jpg lub .png 5.Użytkownik naciska przycisk „Zaawansowane” w sekcji Image		
Czynności końcowe	Powrót do formularza głównego aplikacji		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Zmiana wartości suwaków	Zmiany zostały zachowane	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Wcześniej suwaki były ustawione na wartości domyślne	1.Użytkownik wybiera dowolne suwaki i modyfikuje ich wartości 2.Zatwierdza zmiany klikając „OK”	OK
Lp.	2	Oczekiwany rezultat	
Tytuł	Zmiana wartości suwaków	Zmiany zostały zachowane	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Wcześniej suwaki były ustawione na losowe wartości	1.Użytkownik wybiera dowolne suwaki i modyfikuje ich wartości 2.Zatwierdza zmiany klikając „OK”	OK

Scenariusz Testowy

ID	PU4	Typ	testy akceptacyjne
Opis	Przywrócenie ustawień domyślnych		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny 2.Użytkownik rozpoczyna proces wczytania zdjęcia poprzez kliknięcie przycisku „Wczytaj” 3.Aplikacja wyświetla formularz wyboru (domyślny systemowy). 4.Użytkownik wybiera plik (.jpg lub .png) i przechodzi do sekcji Image		
Czynności końcowe	Zamknięcie sekcji Image ikoną zamykania (krzyżyk) i powrót do formularza głównego aplikacji.		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Przywracanie ustawień domyślnych	Wszelkie modyfikacje ustawień wprowadzonych przez użytkownika zostają wycofane	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	1.Użytkownik rozpoczyna wprowadzanie ustawień klikając przycisk „Zaawansowane” 2.Użytkownik akceptuje wybrane przez siebie ustawienia klikając przycisk „OK”	1.Użytkownik klika ponownie przycisk „Zaawansowane” 2.Użytkownik klika przycisk „Przywróć domyślne ustawienia”	OK
Lp.	1	Oczekiwany rezultat	
Tytuł	Przywracanie ustawień domyślnych na ustawieniach domyślnych	Ustawienia pozostają bez zmian	
Priorytet	Wysoki	Kroki	Poprawność wykonania

Warunki wstępne	1.Użytkownik nie zmodyfikował ustawień domyślnych	1.Użytkownik klika ponownie przycisk „Zaawansowane” 2.Użytkownik klika przycisk „Przywróć domyślne ustawienia”	OK
-----------------	---	---	----

cenariusz Testowy

ID	PU5	Typ	testy akceptacyjne
Opis	Zmiana wybranego zdjęcia		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny 2.Użytkownik rozpoczyna proces wczytywania zdjęcia poprzez kliknięcie przycisku „Wczytaj” 3.Aplikacja wyświetla formularz wyboru 4.Użytkownik wybiera plik (.jpg lub .png) 5.Aplikacja wyświetla formularz ze zdjęciem wybranym przez użytkownika (podgląd)		
Czynności końcowe	Użytkownik zamyka wyświetlone formularze i na ekranie zostaje wyświetlony formularz główny aplikacji.		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Kliknięcie poza wczytanym zdjęciem	Brak reakcji	
Priorytet	Średni	Kroki	Poprawność wykonania
Warunki wstępne	---	1.Kliknięcie poza obszarem wczytanego zdjęcia	OK
Lp.	2	Oczekiwany rezultat	
Tytuł	Kliknięcie na wczytane uprzednio zdjęcie	Powrót do formularza wyboru	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	---	1.Kliknięcie w obszar wczytanego zdjęcia	OK
Lp.	3	Oczekiwany rezultat	
Tytuł	Kliknięcie na wczytane uprzednio zdjęcie i anulowanie wyboru	Powrót do formularza głównego aplikacji	
Priorytet	Średni	Kroki	Poprawność wykonania

Warunki wstępne	---	1.Kliknięcie w obszar wczytanego zdjęcia 2.Kliknięcie „Anuluj” w formularzu wyboru	OK
-----------------	-----	---	----

Scenariusz Testowy

ID	PU6	Typ	testy akceptacyjne
Opis	Anulowanie wczytanego zdjęcia		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny 2.Użytkownik rozpoczyna proces wczytywania zdjęcia poprzez kliknięcie przycisku „Wczytaj” 3.Aplikacja wyświetla formularz wyboru		
Czynności końcowe	Użytkownik zamyka wyświetlany komunikat ikoną zamykania (krzyżyk). Na ekranie użytkownika aplikacja wyświetla formularz główny.		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Użytkownik zamyka systemowy formularz wyboru pliku poprzez przycisk „Anuluj”	Przejsie do formularza głównego aplikacji	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	---	1.Kliknięcie przycisku „Anuluj” w formularzu wyboru	OK
Lp.	2	Oczekiwany rezultat	
Tytuł	Użytkownik zamyka systemowy formularz wyboru pliku poprzez kliknięcie ikony zamykania (krzyżyk)	Przejsie do formularza głównego aplikacji	
Priorytet	Średni	Kroki	Poprawność wykonania
Warunki wstępne	---	1.Kliknięcie ikony zamykania (krzyżyk) na formularzu wyboru	OK
Lp.	3	Oczekiwany rezultat	
Tytuł	Użytkownik zamyka systemowy formularz wyboru pliku poprzez wciśnięcie kombinacji klawiszy „Alt + F4”	Przejsie do formularza głównego aplikacji	

Priorytet	Niski	Kroki	Poprawność wykonania
Warunki wstępne	---	1.Wciśnięcie kombinacji klawiszy „ALT + F4” po wyświetleniu się formularza wyboru	OK

Scenariusz Testowy

ID	PU7	Typ	testy funkcyjne
Opis	Przycięcie zdjęcia w sekcji Image		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny 2.Użytkownik rozpoczyna proces wczytania zdjęcia poprzez kliknięcie przycisku „Wczytaj” 3.Aplikacja wyświetla formularz wyboru (domyślny systemowy). 4.Użytkownik wybiera plik (.jpg lub .png) i przechodzi do sekcji Image 5.W okienku Image użytkownik kadruje odpowiednio zdjęcie.		
Czynności końcowe	Użytkownik klika ikonę zamknięcia formularza (krzyżyk). Na ekranie wyświetla się formularz główny aplikacji		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Próba kadrowania bez zaznaczenia obszaru	Na ekranie wyświetla się informacja o błędzie E009	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	---	1.Użytkownik klika przycisk „Przytnij”	OK
Lp.	2	Oczekiwany rezultat	
Tytuł	Kadrowanie zaznaczonego obszaru	Podgląd zdjęcia zmienia się na kadrowany	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	---	1.Użytkownik zaznacza kadr 2. Użytkownik klika przycisk „Przytnij”	OK

Scenariusz Testowy

ID	PU8	Typ	testy funkcyjne
Opis	Wpisanie sudoku przez użytkownika		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny		
Czynności końcowe	Użytkownik czyści sudoku klikając na pasku narzędzi „Sudoku”, kolejno klikając „Wyczyść”		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Kliknięcie w pole (na siatce sudoku)	Czerwona obramówka wokół klikniętego pola	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	---	1.Użytkownik klika w pole (na siatce sudoku)	OK
Lp.	2	Oczekiwany rezultat	
Tytuł	Wprowadzanie cyfr dozwolonych (takich które mogą się znajdować w odpowiednim miejscu)	W wybranym miejscu pojawia się wprowadzona cyfra	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Wybrane pole jest puste	1.Użytkownik wybiera dowolną komórkę i wprowadza cyfrę	OK
Lp.	3	Oczekiwany rezultat	
Tytuł	Wprowadzanie cyfr dozwolonych(takich które mogą się znajdować w odpowiednim miejscu)	W wybranym miejscu pojawia się wprowadzona cyfra nadpisując poprzednią	
Priorytet	Wysoki	Kroki	Poprawność wykonania

Warunki wstępne	Wybrane pole jest już uzupełnione	1.Użytkownik wybiera dowolną komórkę i wprowadza cyfrę	OK
-----------------	-----------------------------------	--	----

Scenariusz Testowy

ID	PU9	Typ	testy akceptacyjne
Opis	Wprowadzenie do sudoku błędnych danych		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny 2.Użytkownik klika w pole (na siatce sudoku), w którym chce dokonać zmian 3.System podświetla kliknięte pole (czerwone obramowanie) 4.Użytkownik podaje niepoprawną cyfrę lub literę		
Czynności końcowe	Użytkownik czyści wprowadzone dane poprzez kliknięcie w Menu „Sudoku” a następne kliknięcie z rozwijanych opcji funkcji „Wyczyść”. Użytkownik wyświetla na ekranie formularz główny aplikacji		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Próba wpisania litery do komórki sudoku	Wskazane miejsce pozostaje bez zmian	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Wybrana komórka jest pusta	1.Wybranie dowolnej komórki poprzez kliknięcie w nią 2.Wciśnięcie klawisza z literą	OK
Lp.	2	Oczekiwany rezultat	
Tytuł	Próba wpisania litery do komórki sudoku	Wskazane miejsce pozostaje bez zmian	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Wybrana komórka nie jest pusta	1.Wybranie dowolnej komórki poprzez kliknięcie w nią 2.Wciśnięcie klawisza z literą	OK
Lp.	3	Oczekiwany rezultat	
Tytuł	Próba wpisania cyfry '0' do komórki sudoku	Wskazane miejsce pozostaje bez zmian	
Priorytet	Wysoki	Kroki	Poprawność wykonania

Warunki wstępne	Wybrana komórka nie jest pusta	1.Wybranie dowolnej komórki poprzez kliknięcie w nią 2.Wciśnięcie klawisza z literą	OK
Lp.	4	Oczekiwany rezultat	
Tytuł	Próba wpisania cyfry '0' do komórki sudoku	Wskazane miejsce pozostaje bez zmian	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Wybrana komórka jest pusta	1.Wybranie dowolnej komórki poprzez kliknięcie w nią 2.Wciśnięcie klawisza z literą	OK

Scenariusz Testowy

ID	PU10	Typ	testy funkcyjne
Opis	Rozwiązanie poprawnego Sudoku przez program		
Czynności przygotowawcze	1.Zawiera PU1 lub PU9 (wczytanie poprawnie sudoku ze zdjęcia lub z klawiatury) 2.Użytkownik klika przycisk Rozwiąż 3.Aplikacja wysyła zapytanie POST do serwera tablicą cyfr 4.Sudoku zostaje rozwiązane po stronie serwera 5.Aplikacja otrzymuje odpowiedź od serwera 6.Aplikacja wyświetla uzupełnioną siatkę w formularzu głównym, gdzie na zielono zaznaczone są uzupełnione przez algorytm cyfry		
Czynności końcowe	Powrót do formularza głównego aplikacji		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Sudoku z pustymi polami	Losowo uzupełnione sudoku	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Połączenie z serwerem	1.Użytkownik klika przycisk rozwiąż	OK
Lp.	2	Oczekiwany rezultat	
Tytuł	Sudoku z jedną wartością	Poprawne rozwiązanie sudoku z uwzględnieniem wprowadzonych wartości	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Połączenie z serwerem	1.Użytkownik wybiera dowolną komórkę i uzupełnia ją wartością lub wczytuje zdjęcie sudoku z jedną wartością 2.Użytkownik klika przycisk rozwiąż	OK
Lp.	3	Oczekiwany rezultat	

Tytuł	Sudoku z dwoma wartościami	Poprawne rozwiązanie sudoku z uwzględnieniem wprowadzonych wartości	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Połączenie z serwerem	1.Użytkownik wybiera dowolne komórki i uzupełnia je wartościami lub wczytuje poprawne zdjęcie sudoku z dwoma wartościami 2.Użytkownik klika przycisk rozwiąż	OK

Scenariusz Testowy

ID	PU11	Typ	testy akceptacyjne
Opis	Sudoku zostało odczytane błędnie		
Czynności przygotowawcze	1.Apliakacja wyświetla formularz główny 2.Użytkownik rozpoczyna proces wczytywania zdjęcia poprzez kliknięcie „Wczytaj” 3.Aplikacja wyświetla formularz wyboru 4.Użytkownik wybiera plik (.jpg lub .png)		
Czynności końcowe	Użytkownik wyświetla na ekranie formularz główny aplikacji		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Zdjęcie zawierające błędne sudoku	Na ekranie pojawia się komunikat w postaci błędu E004	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Połączenie z serwerem	1.Użytkownik klika przycisk „Zatwierdź” w sekcji IMAGE	OK
Lp.	2	Oczekiwany rezultat	
Tytuł	Zdjęcie poprawnego sudoku w zbyt słabej jakości	Na ekranie pojawia się komunikat w postaci błędu E004	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Połączenie z serwerem	1.Użytkownik klika przycisk „Zatwierdź” w sekcji IMAGE	OK
Lp.	3	Oczekiwany rezultat	
Tytuł	Zdjęcie zrobione pod nieodpowiednim kątem zawierające sudoku	Na ekranie pojawia się komunikat w postaci błędu E004	
Priorytet	Wysoki	Kroki	Poprawność wykonania

Warunki wstępne	Połączenie z serwerem	1.Użytkownik klika przycisk „Zatwierdź” w sekcji IMAGE	OK
-----------------	-----------------------	--	----

Scenariusz Testowy

ID	PU12	Typ	testy akceptacyjne
Opis	Nie rozpoznano sudoku na zdjęciu		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny 2.Użytkownik rozpoczyna proces wczytywania zdjęcia poprzez kliknięcie „Wczytaj” 3.Aplikacja wyświetla formularz wyboru 4.Użytkownik wybiera plik (.jpg lub .png)		
Czynności końcowe	Użytkownik wyświetla na ekranie formularz główny aplikacji		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Zdjęcie nie zawierające sudoku	Na ekranie pojawia się komunikat w postaci błędu E003	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	---	1.Użytkownik klika przycisk „Zatwierdź”	OK
Lp.	2	Oczekiwany rezultat	
Tytuł	Zdjęcie zawierające sudoku	Na ekranie pojawia się komunikat w postaci błędu E003	
Priorytet	Średni	Kroki	Poprawność wykonania
Warunki wstępne	Przycięcie zdjęcia w sposób nieodpowiedni(tak, że na zdjęciu brakuje ramek sudoku)	1.Użytkownik klika przycisk „Zatwierdź”	OK
Lp.	3	Oczekiwany rezultat	
Tytuł	Zdjęcie zawierające sudoku	Na ekranie pojawia się komunikat w postaci błędu E003	
Priorytet	Średni	Kroki	Poprawność wykonania

Warunki wstępne	Źle dobrane parametry w sekcji Image	1.Użytkownik klika przycisk „Zatwierdź”	OK
-----------------	--------------------------------------	---	----

Scenariusz Testowy

ID	PU13	Typ	testy funkcyjne
Opis	Serwis nieaktywny		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny 2.Użytkownik rozpoczyna proces wczytania zdjęcia poprzez kliknięcie przycisku „Wczytaj” 3.Aplikacja wyświetla formularz wyboru (domyślny systemowy). 4.Użytkownik wybiera plik (.jpg lub .png) 5.Aplikacja wyświetla formularz ze zdjęcie wybranym przez użytkownika(podgląd) 6.Użytkownik zatwierdza wybór zdjęcia poprzez kliknięcie przycisku „Zatwierdź” 7.Aplikacja wysyła zdjęcie metodą POST na serwer w celu przeczytania sudoku.		
Czynności końcowe	Użytkownik wyświetla na ekranie formularz główny aplikacji.		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Załadowanie zdjęcia bez połączenia z serwerem	Aplikacja wyświetla błąd E006	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Brak połączenia z serwerem - serwer nieaktywny	---	OK

Scenariusz Testowy

ID	PU14	Typ	testy funkcyjne
Opis	Brak połączenia z siecią		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny 2.Użytkownik rozpoczyna proces wczytania zdjęcia poprzez kliknięcie przycisku „Wczytaj” 3.Aplikacja wyświetla formularz wyboru (domyślny systemowy). 4.Użytkownik wybiera plik (.jpg lub .png) 5.Aplikacja wyświetla formularz ze zdjęcie wybranym przez użytkownika(podgląd) 6.Użytkownik zatwierdza wybór zdjęcia poprzez kliknięcie przycisku „Zatwierdź” 7.Aplikacja wysyła zdjęcie metodą POST na serwer w celu przeczytania sudoku.		
Czynności końcowe	Użytkownik wyświetla na ekranie formularz główny aplikacji.		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Załadowanie zdjęcia bez połączenia z serwerem	Aplikacja wyświetla błąd E005	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Brak połączenia z serwerem - brak połączenia z siecią	---	OK

Scenariusz Testowy

ID	PU15	Typ	testy akceptacyjne
Opis	Zmiana motywu		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny		
Czynności końcowe	Użytkownik na ekranie wyświetla formularz główny aplikacji ze zmienioną szatą graficzną		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Brak wybrania z rozwijanej listy motywu	Szata graficzna formularza głównego aplikacji pozostaje bez zmian	
Priorytet	Średni	Kroki	Poprawność wykonania
Warunki wstępne	---	1.Kliknięcie przycisku „MOTYW”	OK
Lp.	2	Oczekiwany rezultat	
Tytuł	Wybranie motywu „Jasny”	Zmiana szaty graficznej formularza głównego aplikacji na jasne odcienie	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	---	1.Kliknięcie przycisku „MOTYW” 2. Wybór z rozwijanej listy motywu „Jasny”	OK
Lp.	3	Oczekiwany rezultat	
Tytuł	Wybranie motywu „Ciemny”	Zmiana szaty graficznej formularza głównego aplikacji na ciemne odcienie	
Priorytet	Wysoki	Kroki	Poprawność wykonania

Warunki wstępne	---	1.Kliknięcie przycisku „MOTYW” 2. Wybór z rozwijanej listy motywu „Ciemny”	OK
-----------------	-----	---	----

Scenariusz Testowy

ID	PU16	Typ	testy akceptacyjne
Opis	Wyczyszczenie sudoku		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny OPCJONALNIE: 2.Użytkownik klika myszką przycisk „WCZYTAJ” 3.Użytkownik wybiera z formularzu wyboru plik (.jpg, .png) LUB: 2.Użytkownik klika w odpowiednie pola siatki uzupełniając je danymi		
Czynności końcowe	Powrót do formularza głównego aplikacji		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Czyszczenie siatki	Pola siatki pozostają bez zmian	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Wcześniej zdjęcie nie zostało wczytane, siatka sudoku jest pusta	1.Kliknięcie przycisku „Sudoku” na pasku narzędzi 2.Kliknięcie przycisku „Czyść” z rozwijanej listy	OK
Lp.	3	Oczekiwany rezultat	
Tytuł	Czyszczenie siatki	Pola siatki stają się puste	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Wcześniej zdjęcie zostało wczytane, siatka sudoku nie jest pusta(dane zostały wprowadzone ręcznie)	1.Kliknięcie przycisku „Sudoku” na pasku narzędzi 2.Kliknięcie przycisku „Czyść” z rozwijanej listy	OK
Lp.	4	Oczekiwany rezultat	

Tytuł	Czyszczenie siatki	Pola siatki stają się puste	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Wcześniej zdjęcie zostało wczytane, nie wprowadzano danych ręcznie	1.Kliknięcie przycisku „Sudoku” na pasku narzędzi 2.Kliknięcie przycisku „Czyść” z rozwijanej listy	OK

Scenariusz Testowy

ID	PU17	Typ	testy akceptacyjne
Opis	Wystąpił problem podczas otwierania pliku		
Czynności przygotowawcze	1.Aplikacja wyświetla formularz główny 2.Użytkownik klika myszką przycisk „WCZYTAJ”		
Czynności końcowe	Użytkownik wyświetla na ekranie formularz główny aplikacji		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Próba wczytania uszkodzonego pliku	Aplikacja wyświetla komunikat: "Plik nie jest poprawny"	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	---	1.Użytkownik wybiera uszkodzony plik .jpg lub .png	OK

Scenariusz Testowy

ID	PU18	Typ	testy akceptacyjne
Opis	Uruchomienie aplikacji		
Czynności przygotowawcze	1.Użytkownik ma zainstalowaną aplikację		
Czynności końcowe	Użytkownik wyświetla na ekranie formularz główny aplikacji		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Uruchomienie aplikacji przy odpowiedniej wersji Javy	Wyświetlenie formularza głównego aplikacji.	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Użytkownik posiada odpowiednią wersję Javy do uruchomienia aplikacji	1.Użytkownik klika ikonę aplikacji	OK

Scenariusz Testowy

ID	PU19	Typ	testy akceptacyjne
Opis	Uruchomienie aplikacji		
Czynności przygotowawcze	1.Użytkownik ma zainstalowaną aplikację		
Czynności końcowe	Użytkownik zamyka wyświetlony błąd ikoną zamykania (krzyżyk)		

Przypadki testowe

Lp.	1	Oczekiwany rezultat	
Tytuł	Uruchomienie aplikacji przy nieodpowiedniej wersji Javy	Na ekranie pojawia się komunikat w postaci błędu E001	
Priorytet	Wysoki	Kroki	Poprawność wykonania
Warunki wstępne	Użytkownik nie posiada na swoim komputerze zainstalowanej odpowiedniej wersji Javy	1.Użytkownik klika ikonę aplikacji	OK

3 Dodatkowe testy dotyczące obróbki i rozpoznania obrazu.

3.1 Testy odczytu Sudoku ze zdjęcia.

Obsługiwane formaty: jpg, png

Prerekwizyt: zdjęcie musi zawierać sudoku (nie musi ono być czarno-białe, ale przy innym kolorze zdjęcia nie gwarantujemy skuteczność)

Wstęp

Wymaganie zostało sprawdzone na podstawie porównania wyników otrzymanych z użytkowania naszego systemu z przygotowanymi wcześniej oczekiwanymi wartościami. Faza testowania została podzielona na dwie części. Pierwsza z nich dotyczy skuteczności samego modelu. Druga natomiast bardziej przypomina faktyczne użycie aplikacji i polegała na symulacji całego procesu ekstrakcji.

W trakcie procesu testów manualnych sprawdzających poprawność kolejnych kroków milowych:

- **Poprawność wycięcia sudoku:** 129/130
Nasze podejście daje bardzo dobre rezultaty. Jedynym przypadkiem jest zdjęcie nr 50 w którym nie byliśmy w stanie wyciąć poprawnie sudoku. Problem opisany poniżej..
- **Poprawność usunięcia linii**
Do usuwania linii używamy HoughLinesP który jest algorytmem probabilistycznym do znajdowania linii na obrazie o wskazanych parametrach. Algorytm zostawia nieznaczące defekty które są rozpoznawane na poziomie „wyliminowania szumu”.
- **Poprawność wyliminowania szumu** (obiektów niebędących cyframi)
Ze względu na różne jakości zdjęć usuwanie szumu usuwa większość zniekształceń. Przy słabych zdjęciach jakościowo może dojść do wyliminowania cyfr. Ze względu na rygorystyczne warunki akceptacji obiektu jako cyfry redukujemy szanse rozpoznania obiektów niepożądanych.
- **Poprawność rozpoznania szumu**
Ze względu na brak możliwości całkowitego pozbycia się zniekształceń szum jest rozpoznawany przez algorytm. W przypadku rozpoznaniu szumu algorytm zwraca 0 (puste pole)

3.1.1 Faza 1

Kod testujący znajduje się w

Web\RecognizerLib\src\test\java\pl\sudokusolver\recognizerlib\ocr\RecognizersTest.java

W tej fazie testujemy poprawność modelu do rozpoznawania cyfr.

W naszej aplikacji możemy skorzystać z kilku modeli:

- SVM
- ANN
- Tesseract

Testowaniu w tej fazie podlegają tylko SVM i ANN, ponieważ Tesseract jest oprogramowaniem zewnętrznym co wiąże się z tym, że testowanie jego skuteczności nie jest wymagane. Do treningu oraz testowania pozostałych modeli skorzystaliśmy z dane z MNIST. Są to co prawda dane związane z pismem ręcznym jednak dają one poprawne wyniki dla pisma drukowanego.

Wyniki fazy 1: Dla SVM poprawność rozpoznania plasuje się na około 96.7%. Natomiast dla ANN wynik to 97.8%.

3.1.2 Faza 2

Kod testujący znajduje się w

Web\RecognizerLib\src\test\java\pl\sudokusolver\recognizerlib\sudoku\BaseSudokuExtractor.java

W tej fazie testowaniu podlegają wszystkie 3 modele dostępne w naszym systemie. Jednak tesseract został przetestowany z użyciem dwóch trybów. Jeden z nich (simple) ignoruje niektóre błędy podczas rozpoznawanie cyfr, drugi (strict) natomiast przerywa dalsze rozpoznawanie w przypadku wystąpienia takiego błędu (co ułatwia wykrywanie zdjęć bardzo słabej jakości). Celem tego testu jest sprawdzenie z jaką dokładnością jesteśmy w stanie rozpoznać oraz przetworzyć zawartość zdjęcia. Więc test ten symuluje działanie całego systemu (części leżącej po stronie serwera). Dane, z których korzystamy znajdują się w Data\TestImgs i są złożone z zdjęcia oraz z oczekiwanego rezultatu. Ostatecznej ocenie podlega procent zgodności wyników z oczekiwanym rezultatem. Algorytmy ekstrakcji sudoku działają na domyślnych parametrach, które zostały dobrane w taki sposób, aby wynik średni był jak najlepszy. Jedyną różnicą pomiędzy testowaniem poszczególnych modeli są one sam.

Wyniki fazy 2:

	SVM	ANN	Tesseract Simple	Tesseract Strict
Średnia skuteczność (wraz z błędami) [%]	97.5	95.0	96.0	72.2
Sudoku rozpoznane w 100% (na 156)	110	22	86	84
Liczba błędów	0	0	0	42
Średnia skuteczność (bez błędów) [%]	97.5	95.0	96.0	98.7
Średni czas przetwarzania [ms]	202	149	4131	3545
Minimalny czas przetwarzania [ms]	84	72	702	240

(Wszystkie pomiary czasowe zostały przeprowadzone na komputerze średniej klasy)

Najgorsze przypadki:

Podczas fazy 2 sprawdziliśmy również czy rozpoznawanie jakiegoś zdjęcia nie jest na poziomie mniejszym niż 70. Znaleźliśmy dwa takie zdjęcia (nr. 50 oraz 64).



Komentarz do zdjęcia nr. 50: Rozpoznawalność tego zdjęcia jest na poziomie 28% i wynika to z tego, że posiada ono grubą „obramówkę”. Powoduje to, że algorytm ekstrakcji traktuje całe zdjęcie jako sudoku. Ten przypadek uznaliśmy za nieznaczący, ponieważ problem ten można rozwiązać przy pomocy aplikacji klienta. Wystarczy przyciąć zdjęcie w taki sposób żeby nie zawierało „obramówki”. Po tym zabiegu nasz system nie ma już problemu z poprawnym rozpoznaniem sudoku.



Komentarz do zdjęcia nr. 64:

Na tym zdjęciu nasz program osiąga skuteczność rzędu 64.1%. W tym przypadku uznajemy, że wynik ten jest nieznaczący z uwagi na niską jakość zdjęcia.

3.1.3 Podsumowanie

Uważamy, że dane do testowania, z których skorzystaliśmy w wiarygodny sposób odzwierciedlają codzienne użycie naszego systemu, ponieważ zawierają zdjęcia o różnej rozdzielczości, zrobione pod różnymi kątami, o różnej jakości, różnie wykadrowane. Jednocześnie nasza aplikacja umożliwia ustawienie parametrów, które są brane pod uwagę w procesie obróbki zdjęcia. Zmiana tych parametrów z wartości domyślnych może wpłynąć pozytywnie na wyniki w niektórych skrajnych wypadkach. Na podstawie tego testu jesteśmy w stanie zagwarantować, że nasz system działa ze

skutecznością większą niż zakładane 70%.

3.2 Klasa DumpManualTest

Klasa służy do zrobienia zrzutu kolejnych etapów obróbki zdjęcia i wyników rozpoznania wraz z oczekiwanym rezultatem.

3.2.1 Hierarchia

- **Grid:** W tym folderze możemy znaleźć wygląd wyciętego sudoku z zdjęcia. Nazwa Pliku informuje o nr. testowanego sudoku. (0-129 w przypadku naszych testów)
- **CleanGrid:** W tym folderze możemy znaleźć wygląd sudoku po nałożonych filtrach czyszczących zdjęcie. Nazwa Pliku informuje o nr. testowanego sudoku. (0-129 w przypadku naszych testów)
- **Cell:** W tym folderze możemy znaleźć wygląd każdej kolejnej komórki naszego sudoku. Nr. podfolderu informuje o nr. testowanego sudoku. (0-129 w przypadku naszych testów) Nr. pliku znajdującego się w folderze informuje o nr. komórki w sudoku.
Zastosowana formuła:
$$\text{kolumna} = \text{nazwaPliku} \% 9$$
$$\text{wiersz} = \text{floor}(\text{nazwaPliku} / 9)$$
- **Digit:** W tym folderze możemy znaleźć wygląd obiektów które rozpozналиśmy jako cyfry. Nr. podfolderu informuje o nr. testowanego sudoku. (0-129 w przypadku naszych testów) Nr. pliku znajdującego się w folderze informuje o nr. komórki w sudoku. W przypadku braku pliku z kolejnym numerem informuje on że w danej komórce nie znaleźliśmy obiektu pasującego do naszych wytycznych. Zastosowana formuła:
$$\text{kolumna} = \text{nazwaPliku} \% 9$$
$$\text{wiersz} = \text{floor}(\text{nazwaPliku} / 9)$$
- **Result:** W tym folderze możemy znaleźć wyniki procentowe rozpoznania każdego sudoku, a także informacje jakie komórki zostały źle rozpoznane. Nazwa podfolderu informuje o algorytmie rozpoznającym który został testowany. Nazwa pliku informuje o Nr. sudoku które jest rozpoznawane a także procentowej skuteczności rozpoznania. Zastosowana formuła:
Nr. Po lewej od symbolu ' _ ' informuje o nr. sudoku
Nr. po prawej od symbolu ' _ ' informuje o procentowym rozpoznaniu

Format Pliku:

- Score: Dokładny wynik procentowy.
- x|y: x oznacza jak rozpoznaliśmy naszą cyfrę, y jaka powinna być (0 w przypadku pustej komórki).
- Separator: | i - określają wygląd sudoku dzieląc plansze na 9 części (standardowe sudoku)

Uwagi: W procesie zapisywania 2 zdjęcia zostały obrócone o 90 stopni. Obrót jest tylko przy zapisywaniu. Nie ma on odzwierciedlenia w działaniu programu. Wyniki mogą się nieznacznie różnić ze względu na błędy zaokrągleń, a także kompresję plików.

3.2.2 Proces Testowania

Ze względu na brak możliwości automatycznego sprawdzania poprawności:

1. Wycięcia sudoku
2. Wyczyszczenia sudoku
3. Poprawnej klasyfikacji obiektu jako cyfry

Proces testowania był stopniowym sprawdzaniem kolejnych faz przeprowadzenia algorytmu.

1. Wycięcie Sudoku.

Głównej ocenie podlegał fakt poprawnego znalezienia i wycięcia sudoku. W przypadku złego znalezienia lub niepoprawnego wycięcia fakt ten był zgłaszany i odnotowany.

2. Wyczyszczenia sudoku.

Głównym kryterium poprawności wyczyszczenia szumu było: usunięcie w większości ramek i linii tworzących sudoku. usunięcie szumu w pobliżu cyfr. Do tego testu wykorzystano głównie folder CleanGrid i Cell Poprawnej klasyfikacji obiektu jako cyfry. W tym kroku było sprawdzenie folderu Digit w których powinny być wyodrębnione tylko cyfry. W przypadku wyodrębniania czegoś poza cyfrą problem był interpretowany indywidualnie w celu analizy “Dlaczego dany obiekt(szum) został rozpoznany jako cyfra”

3. Poprawnej klasyfikacji obiektu jako cyfry. Proces poprawności rozpoznania cyfry został zautomatyzowany. Wykorzystano do tego ręcznie przepisane sudoku i porównywanie naszego wyniku z wynikiem wzorcowym.