**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ**

**Московский государственный технический университет**

**им. Н.Э. Баумана**

**(МГТУ им. Н.Э. Баумана)**

**Кафедра «Информационная системы и технологии» (ИУ3)**

Лабораторная работа № 2

По дисциплине: «Цифровая обработка изображений»

# Тема: «Разработка программы распознавания блюд бауманской столовой»

Выполнил: Быковский М.К.

студент группы ИУ3-22М

Проверил: Алфимцев А.Н.

Профессор каф. ИУ3

Москва, 2018 г.

## Задание

1. Подготовить выборку 10 цветных цифровых изображений блюд бауманской столовой (по аналогии с примером).
2. В среде Spyder (сборка Anaconda) на языке Python (3.5 и старше) создать проект и подключить библиотеку *scikit-image.*
3. Из л.р. 1 взять модуль загрузки цветного цифрового изображения и модуль обработки пикселей.
4. Запрограммировать формулу перевода цветного цифрового изображения в цветовое пространство в соответствии с вариантом 3 - CMYK
5. Создать классификацию блюд из цифровых изображений.
6. Для каждого блюда определить цветовые характеристики. Задать распределение значений каждого цвета.
7. Провести эксперимент по распознаванию блюд с визуализацией результатов.
8. Дать комментарий каждой строчке кода!
9. Продемонстрировать работу программы преподавателю.
10. Построить график ошибок первого и второго рода по результатам распознавания.
11. Подготовить и защитить отчет (титульный лист, задание, теоретическая часть, диаграмма структуры программы, принтскрины интерфейса и основных шагов работы программы, листинг программы с комментариями, список использованной литературы).

## Теоретическая часть

Четырёхцветная автотипия (CMYK: Cyan, Magenta, Yellow, Key color) — [субтрактивная схема](https://ru.wikipedia.org/wiki/%D0%A1%D1%83%D0%B1%D1%82%D1%80%D0%B0%D0%BA%D1%82%D0%B8%D0%B2%D0%BD%D1%8B%D0%B9_%D1%81%D0%B8%D0%BD%D1%82%D0%B5%D0%B7) формирования цвета, используемая прежде всего в [полиграфии](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%B8%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F) для стандартной триадной печати. Схема CMYK обладает сравнительно с [RGB](https://ru.wikipedia.org/wiki/RGB) меньшим [цветовым охватом](https://ru.wikipedia.org/wiki/%D0%A6%D0%B2%D0%B5%D1%82%D0%BE%D0%B2%D0%BE%D0%B9_%D0%BE%D1%85%D0%B2%D0%B0%D1%82).

По-русски эти [цвета](https://ru.wikipedia.org/wiki/%D0%A6%D0%B2%D0%B5%D1%82) часто называют [голубым](https://ru.wikipedia.org/wiki/%D0%93%D0%BE%D0%BB%D1%83%D0%B1%D0%BE%D0%B9_%D1%86%D0%B2%D0%B5%D1%82), [пурпурным](https://ru.wikipedia.org/wiki/%D0%9F%D1%83%D1%80%D0%BF%D1%83%D1%80%D0%BD%D1%8B%D0%B9_%D1%86%D0%B2%D0%B5%D1%82) и [жёлтым](https://ru.wikipedia.org/wiki/%D0%96%D1%91%D0%BB%D1%82%D1%8B%D0%B9_%D1%86%D0%B2%D0%B5%D1%82), хотя первый точнее называть [сине-зелёным](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D0%B5-%D0%B7%D0%B5%D0%BB%D1%91%D0%BD%D1%8B%D0%B9_%D1%86%D0%B2%D0%B5%D1%82), а [маджента](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D0%B4%D0%B6%D0%B5%D0%BD%D1%82%D0%B0_(%D1%86%D0%B2%D0%B5%D1%82)" \o "Маджента (цвет)) — лишь часть пурпурного спектра. Печать четырьмя красками, соответствующими CMYK, также называют печатью триадными красками.

Цвет в CMYK зависит не только от спектральных характеристик красителей и от способа их нанесения, но и их количества, характеристик бумаги и других факторов. Фактически, цифры CMYK являются лишь набором аппаратных данных для [фотонаборного автомата](https://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%82%D0%BE%D0%BD%D0%B0%D0%B1%D0%BE%D1%80%D0%BD%D1%8B%D0%B9_%D0%B0%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82) или [CTP](https://ru.wikipedia.org/wiki/CTP) и не определяют [цвет](https://ru.wikipedia.org/wiki/%D0%A6%D0%B2%D0%B5%D1%82) однозначно.

Так, исторически в разных странах сложилось несколько стандартизованных процессов [офсетной печати](https://ru.wikipedia.org/wiki/%D0%9E%D1%84%D1%81%D0%B5%D1%82%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B5%D1%87%D0%B0%D1%82%D1%8C). Сегодня это американский, европейский и японский стандарты для мелованной и немелованной бумаг. Именно для этих процессов разработаны стандартизованные бумаги и краски (например, стандарты ECI). Для них же созданы соответствующие [цветовые модели](https://ru.wikipedia.org/wiki/%D0%A6%D0%B2%D0%B5%D1%82%D0%BE%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C) CMYK, которые используются в процессах цветоделения. Однако многие типографии, в которых работают специалисты с достаточной квалификацией (или способные на время пригласить такого специалиста), нередко создают профиль, описывающий печатный процесс конкретной печатной машины с конкретной бумагой. Этот профиль они предоставляют своим заказчикам.

Особо выделяется цветовая модель CMYK 255. Суть модели: каждый из цветов описывается градацией не от 0 до 100 (как в классической CMYK модели), а от 0 до 255.

В CMYK используются четыре цвета, первые три в аббревиатуре названы по первой букве цвета, а в качестве четвёртого используется [чёрный](https://ru.wikipedia.org/wiki/%D0%A7%D1%91%D1%80%D0%BD%D1%8B%D0%B9_%D1%86%D0%B2%D0%B5%D1%82). Одна из версий утверждает, что K — сокращение от [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) blacK. Согласно этой версии, при [выводе полиграфических плёнок](https://ru.wikipedia.org/w/index.php?title=%D0%92%D1%8B%D0%B2%D0%BE%D0%B4_%D0%BF%D0%BE%D0%BB%D0%B8%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D1%85_%D0%BF%D0%BB%D1%91%D0%BD%D0%BE%D0%BA&action=edit&redlink=1) на них одной буквой указывался цвет, которому они принадлежат. Чёрный не стали обозначать B, чтобы не путать с B ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) blue) из модели [RGB](https://ru.wikipedia.org/wiki/RGB), а стали обозначать K (по последней букве). Профессиональные цветокорректоры работают с десятью каналами RGBCMYKLab, используя доступные цветовые пространства. Поэтому при обозначении CMYK как CMYB фраза «манипуляция с каналом B» требовала бы уточнения «манипуляция с каналом B из CMYB», что было бы неудобно.

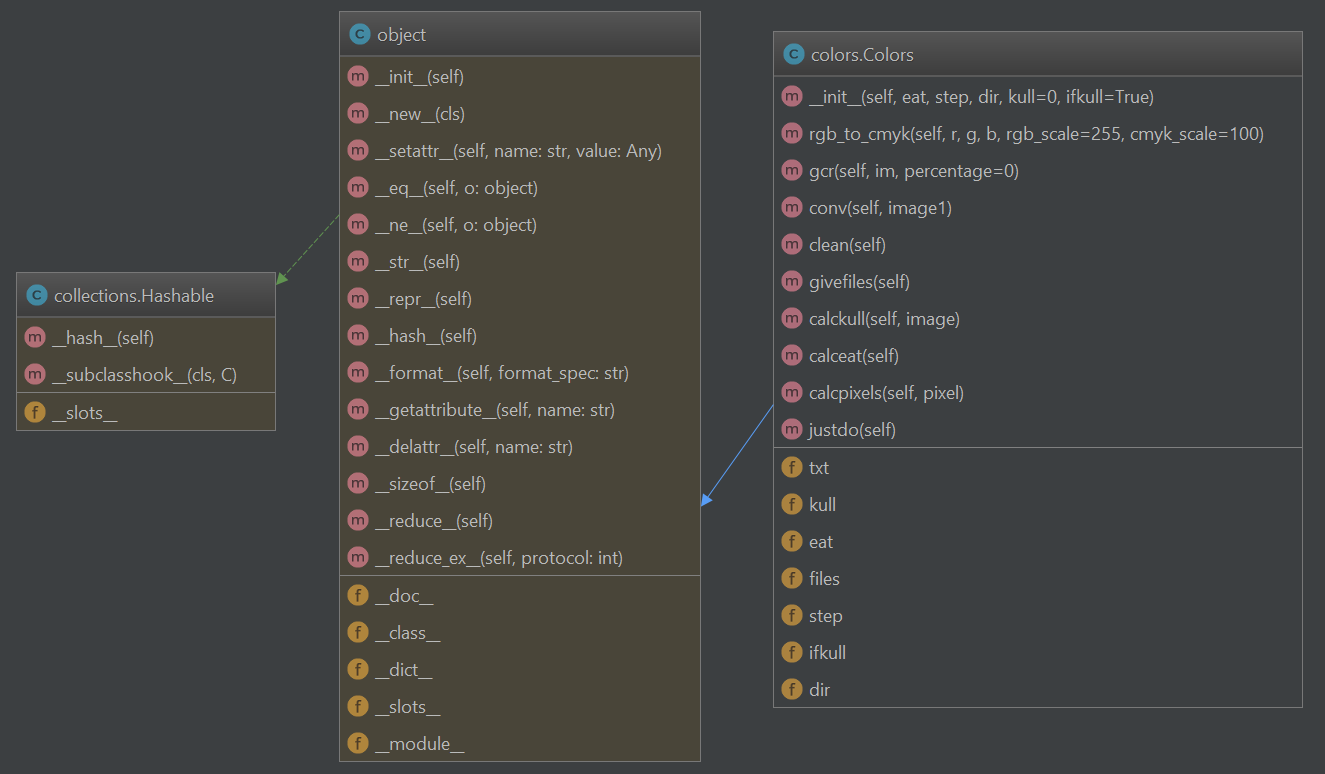
Согласно другому варианту, K является сокращением от слова ключевой: в англоязычных странах термином key plate обозначается печатная форма для чёрной краски, печатаемая последней поверх заранее напечатанных трех предыдущих красок.

Третий вариант говорит о немецком происхождении К — [нем.](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BC%D0%B5%D1%86%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) Kontur. Эта версия подтверждается ещё и тем, что многие старые монтажники так и называют соответствующую плёнку — контур, контурная. Тем более, что в технологии печати чёрный и вправду как бы окантовывает изображение.

Несмотря на то, что [чёрный цвет](https://ru.wikipedia.org/wiki/%D0%A7%D1%91%D1%80%D0%BD%D1%8B%D0%B9_%D1%86%D0%B2%D0%B5%D1%82) можно получать смешением в равной пропорции пурпурного, голубого и жёлтого [красителей](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B0%D1%81%D0%B8%D1%82%D0%B5%D0%BB%D0%B8), по ряду причин ([чистота цвета](https://ru.wikipedia.org/w/index.php?title=%D0%A7%D0%B8%D1%81%D1%82%D0%BE%D1%82%D0%B0_%D1%86%D0%B2%D0%B5%D1%82%D0%B0&action=edit&redlink=1), переувлажнение бумаги и др.) такой подход обычно неудовлетворителен. Основные причины использования дополнительного чёрного пигмента таковы:

* На практике в силу неидеальности красителей и погрешностей в пропорциях компонентов смешение реальных пурпурного, голубого и жёлтого цветов даёт скорее грязно-коричневый или грязно-серый цвет; триадные краски не дают той глубины и насыщенности, которая достигается использованием настоящего чёрного. Так как чистота и насыщенность чёрного цвета, а также стабильность оттенка нейтральных (серых) областей чрезвычайно важны в печатном процессе, был введён ещё один цвет.
* При выводе мелких чёрных деталей изображения или текста без использования чёрного пигмента возрастает риск неприводки (недостаточно точное совпадение точек нанесения) пурпурного, голубого и жёлтого цветов. Увеличение же точности печатающего аппарата требует неадекватных затрат.
* Смешение 100% пурпурного, голубого и жёлтого пигментов в одной точке в случае [струйной](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%B9%D0%BD%D1%8B%D0%B9_%D0%BF%D1%80%D0%B8%D0%BD%D1%82%D0%B5%D1%80) печати существенно смачивает бумагу, деформирует её и увеличивает время просушки. Аналогичные проблемы с так называемой суммой красок возникают и в офсетной печати. В зависимости от типа материала и технологии печати ограничение по сумме красок может быть ниже 300 % (например, в газетной печати типичное ограничение 260—280 %), что делает технически невозможным синтез насыщенного чёрного из трёх стопроцентных компонентов триады.
* Чёрный пигмент (в качестве которого, как правило, используется [сажа](https://ru.wikipedia.org/wiki/%D0%A1%D0%B0%D0%B6%D0%B0)) существенно дешевле остальных трёх.

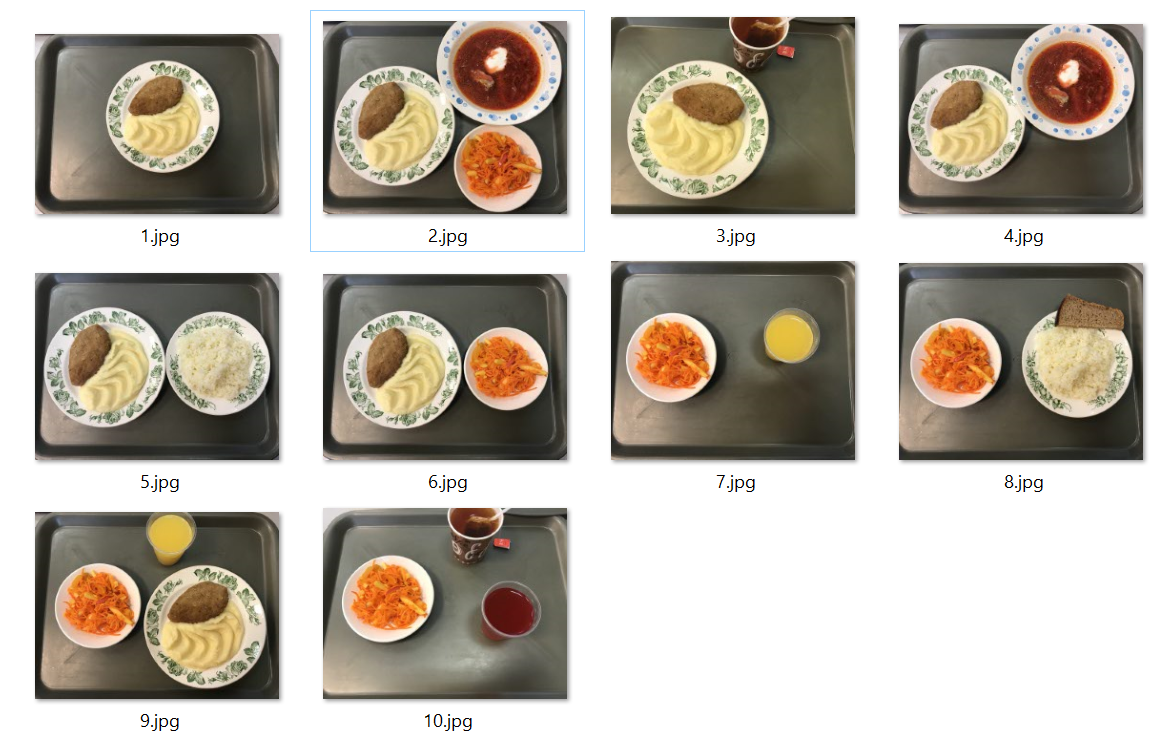
## Диаграмма структуры программы



## Классификация блюд

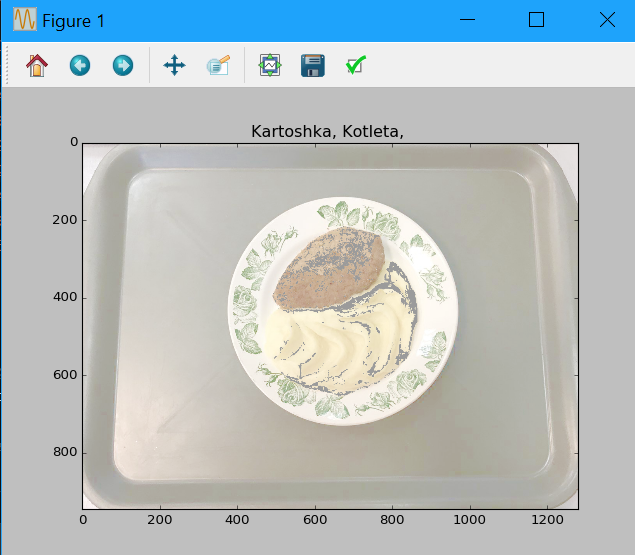
|  |  |  |
| --- | --- | --- |
| Изображение | Название | Значение цвета CMYK |
|  | Картошка | (0, 10, 39, 16) |
|  | Котлета | (0, 28, 58, 43) |
|  | Чай | (0, 53, 71, 58) |
|  | Морковь | (0, 55, 90, 19) |
|  | Борщ | (0, 62, 75, 64) |
|  | Рис | (0, 8, 32, 12) |
|  | Сок | (0, 18, 71, 9) |
|  | Компот | (0, 80, 82, 65) |
|  | Хлеб | (0, 21, 46, 40) |

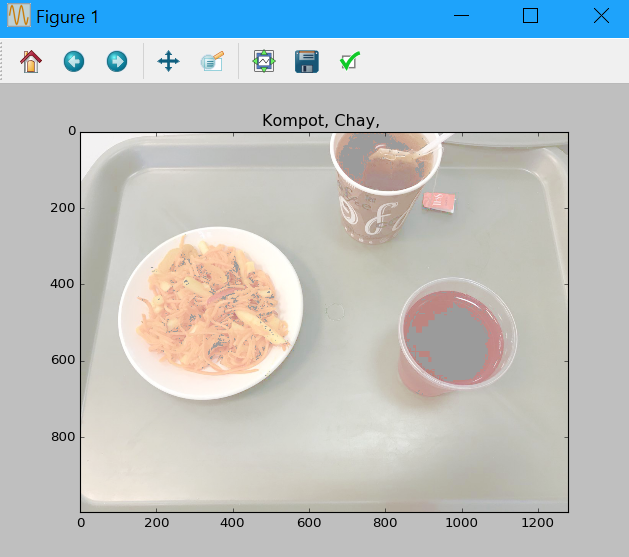
## Выборка блюд

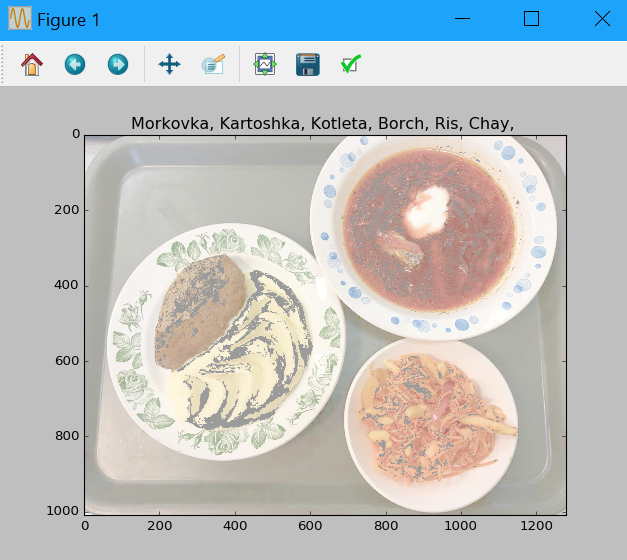


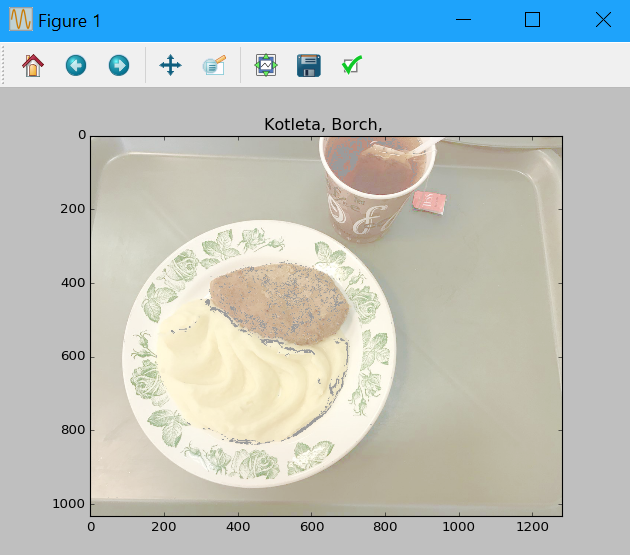
## Интерфейс и основные шаги работы программы

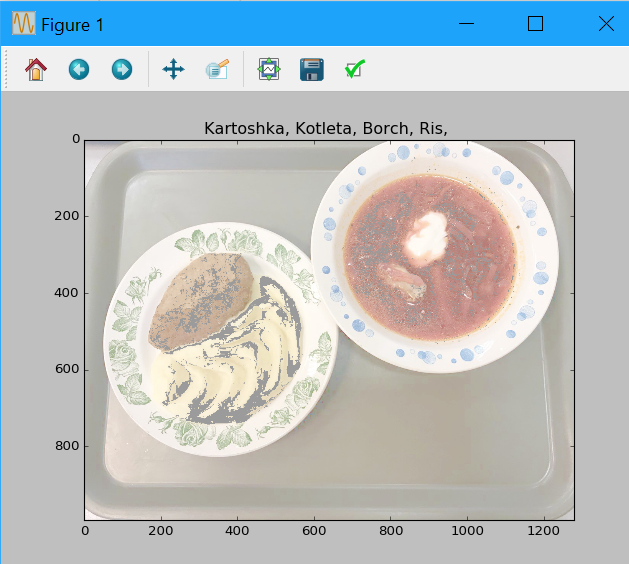
В результате работы программы мы получаем следующий результат:

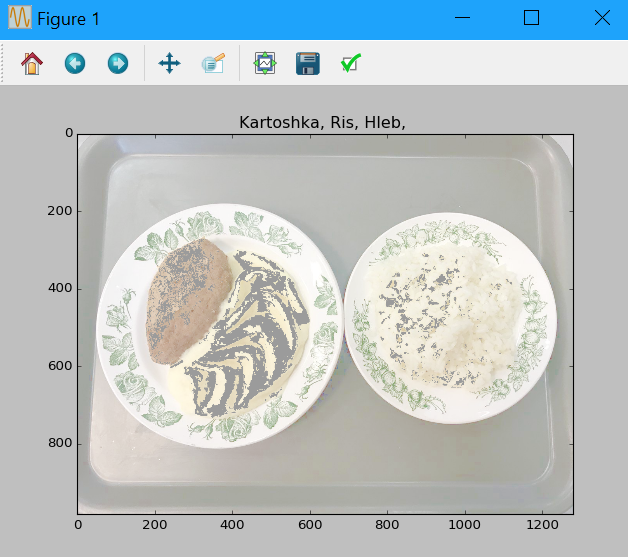


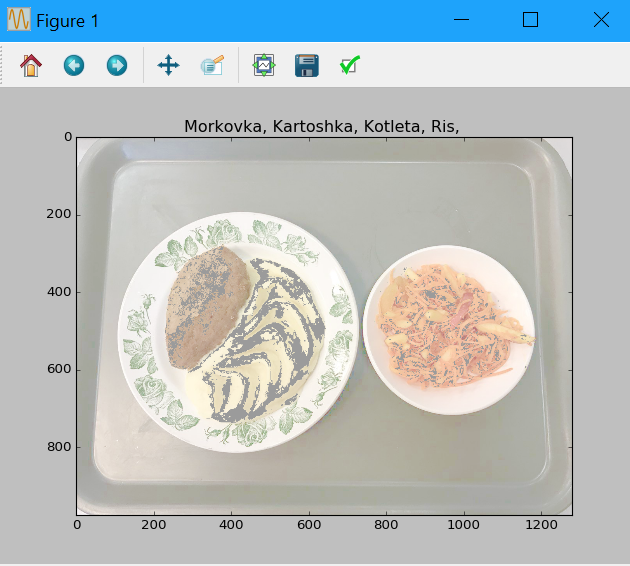


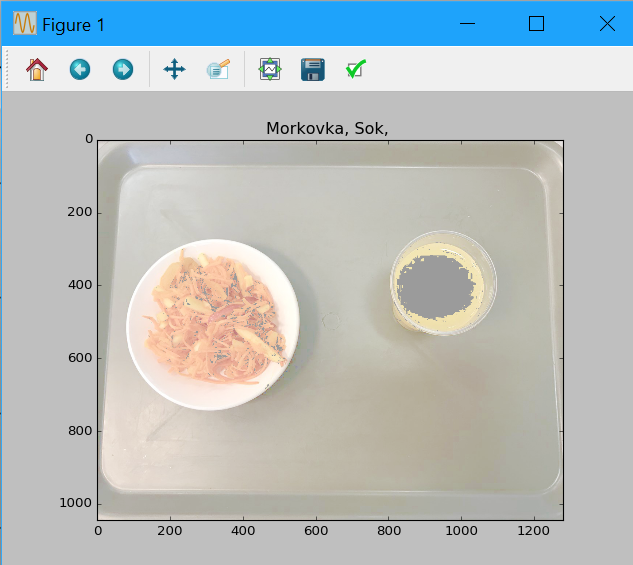


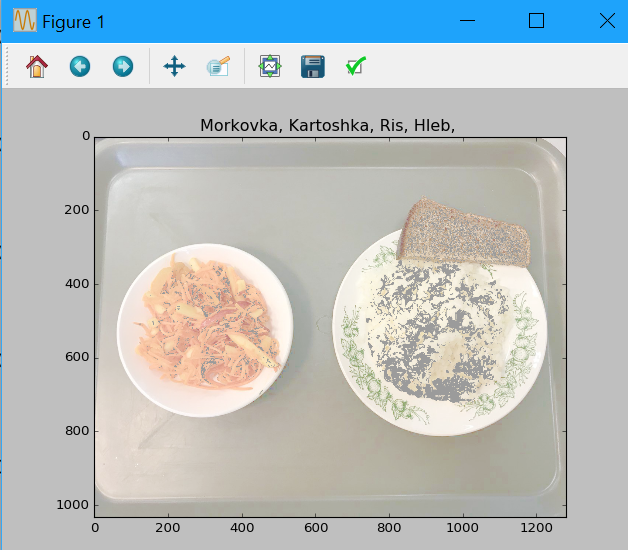


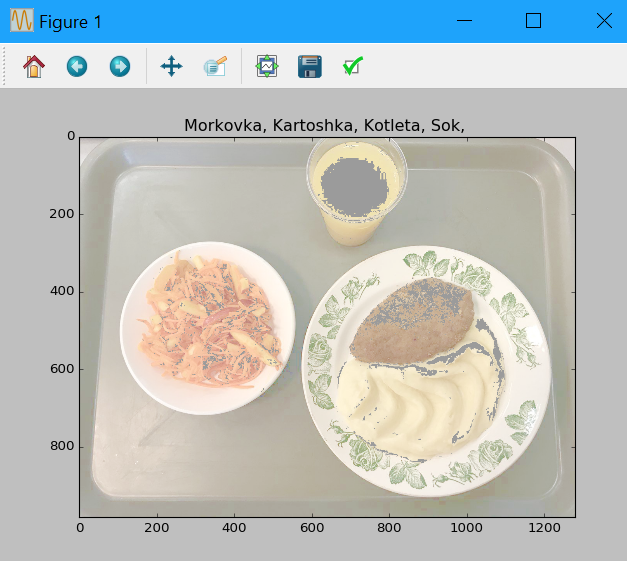












## Ошибки первого и второго рода

|  |  |  |
| --- | --- | --- |
|  | Принять H0 | Отвергнуть H0 |
| Верна H0 | 26 | 7 |
| Ложна H0 | 4 | 53 |

Всего: 90 измерений

Верно: 88%

Ошибка первого рода: 4%

Ошибка второго рода: 8%

## Листинг программы

Colors.py:

import os  
from PIL import Image  
from matplotlib import pyplot as plt  
  
class Colors:  
  
 def \_\_init\_\_(self, eat, step, dir, kull=0, ifkull=True):  
 *# задаем начальные данные* self.eat = eat  
 self.step = step  
 self.dir = dir  
 self.kull = kull  
 self.ifkull = ifkull  
 self.txt = ''  
 self.files = []  
 self.givefiles()  
  
 def rgb\_to\_cmyk(self, r, g, b, rgb\_scale=255, cmyk\_scale=100):  
  
 if (r == 0) and (g == 0) and (b == 0):  
 *# black* return 0, 0, 0, cmyk\_scale  
  
 *# rgb [0,255] -> r'g'b'[0,1]* R = r / float(rgb\_scale)  
 G = g / float(rgb\_scale)  
 B = b / float(rgb\_scale)  
  
 *# extract cmyk [0,1]* m = max(R, G, B)  
 k = 1 - m  
 c = (1 - R - k) / (1 - k)  
 m = (1 - G - k) / (1 - k)  
 y = (1 - B - k) / (1 - k)  
  
 *# rescale to the range [0,cmyk\_scale]* return int(c\*cmyk\_scale), int(m\*cmyk\_scale), int(y\*cmyk\_scale), int(k\*cmyk\_scale)  
  
 def gcr(self, im, percentage=0):  
 *'''basic "Gray Component Replacement" function. Returns a CMYK image with  
 percentage gray component removed from the CMY channels and put in the  
 K channel, ie. for percentage=100, (41, 100, 255, 0) >> (0, 59, 214, 41)'''* cmyk\_im = im.convert('CMYK')  
 if not percentage:  
 return cmyk\_im  
 cmyk\_im = cmyk\_im.split()  
 cmyk = []  
 for i in range(4):  
 cmyk.append(cmyk\_im[i].load())  
 for x in range(im.size[0]):  
 for y in range(im.size[1]):  
 gray = min(cmyk[0][x,y], cmyk[1][x,y], cmyk[2][x,y]) \* percentage / 100  
 for i in range(3):  
 cmyk[i][x,y] = cmyk[i][x,y] - gray  
 cmyk[3][x,y] = gray  
 return Image.merge('CMYK', cmyk\_im)  
  
 def conv(self, image1):  
 mass = [self.rgb\_to\_cmyk(pixel[0], pixel[1], pixel[2]) for pixel in iter(image1.getdata())]  
 cmyk = self.gcr(image1)  
 cmyk.putdata(mass)  
 return cmyk  
  
 def clean(self):  
 i = 0  
 for k in self.eat:  
 self.eat[i][4] = 0  
 i += 1  
  
 def givefiles(self):  
 self.files = os.listdir(self.dir)  
  
 def calckull(self, image):  
 width, height = image.size  
 if (width >= height):  
 self.kull = width \* 10  
 else:  
 self.kull = height \* 10  
  
 def calceat(self):  
 self.txt = ''  
 for e in self.eat:  
 if (e[4] > self.kull):  
 self.txt += e[5] + ', '  
  
 def calcpixels(self, pixel):  
 i = 0  
 for e in self.eat:  
 if (e[0] - self.step < pixel[0] and pixel[0] < e[0] + self.step and  
 e[1] - self.step < pixel[1] and pixel[1] < e[1] + self.step and  
 e[2] - self.step < pixel[2] and pixel[2] < e[2] + self.step and  
 e[3] - self.step < pixel[3] and pixel[3] < e[3] + self.step):  
 self.eat[i][4] += 1  
 m = [0, 0, 0, 100]  
 return m  
 i += 1  
 return False  
  
 def justdo(self):  
  
 for im in self.files:  
  
 image = Image.open(self.dir + im)  
  
 if(self.ifkull):  
 self.calckull(image)  
  
 image = self.conv(image)  
 *#image.show()* self.clean()  
  
 i = 0  
  
 imag = image.getdata()  
 pixels = [[pixel[0], pixel[1], pixel[2], pixel[3]] for pixel in iter(imag)]  
  
 for pixel in iter(imag):  
 p = self.calcpixels(pixel)  
 if(p):  
 pixels[i] = p  
 i += 1  
  
 print(self.eat)  
  
 pix = [(pixel[0], pixel[1], pixel[2], pixel[3]) for pixel in iter(pixels)]  
  
 self.calceat()  
  
 image.putdata(pix)  
 plt.title(self.txt)  
 plt.imshow(image)  
 plt.show()

lab2.py:

import colors *#подключаем написанную библиотеку  
  
# исходные данные  
# eat = [[C, M, Y, K, COUNT, TITLE],[]...]*eat = [  
 [0, 55, 90, 19, 0, 'Morkovka'],  
 [0, 10, 39, 16, 0, 'Kartoshka'],  
 [0, 28, 58, 43, 0, 'Kotleta'],  
 [0, 62, 75, 64, 0, 'Borch'],  
 [0, 18, 71, 9, 0, 'Sok'],  
 [0, 8, 32, 12, 0, 'Ris'],  
 [0, 21, 46, 40, 0, 'Hleb'],  
 [0, 80, 82, 65, 0, 'Kompot'],  
 [0, 53, 71, 58, 0, 'Chay']  
]  
step = 5  
dir = 'lab2/'  
  
*# новый экзампляр класса*color = colors.Colors(eat, step, dir, 5000, False)  
  
*# функция распознавания*color.justdo()

## Список использованной литературы

1. <https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80_%D0%9A%D1%8D%D0%BD%D0%BD%D0%B8> - Электронный ресурс – дата обращения 15.02.2018 г.
2. https://habrahabr.ru/post/114589/ – Электронный ресурс – дата обращения 15.02.2018 г.