Labor 2

A gyakorlat célja: ismerkedjünk meg a színes kép anatómiájával. Egy színes képnek van három színcsatornája, a vörös (R), a zöld (G) meg a kék (K). Ezek a színcsatornák egymástól teljesen függetlenek, ezért őket szét lehet választani három darab egy színcsatornás képre. Fordított irányba haladva pedig három darab azonos méretű egy színcsatornás képből elvileg össze lehet rakni egy színes képet, csak azt kell megmondanunk, melyik legyen a vörös, melyik a zöld és melyik a kék.

Az OpenCV lehetőséget ad a Mat struktúra által leírt több színcsatornás képet szétszedni színcsatornáira:

```
split(im, bgr);
```

Hivatkozás a színcsatornákra

```
Mat bgr [3], &b = bgr [0], &g = bgr [1], &r = bgr [2];
```

Hasonlóan, össze is lehet rakni színcsatornákból egy színes képet:

```
merge(vector<Mat>{b, g, z}, result);
```

Az új kép létrehozása

```
Mat imBig = Mat(im.rows * 3, im.cols * 6, im.type());
Mat result = im.clone();
Mat z(im.rows, im.cols, CV_8UC1, Scalar(0));
```

Színes háttér

```
imBig.setTo(Scalar(128, 128, 255, 0));
```

Feladat: Töltsünk be egy színes képet. (eper.jpg)

1. Állítsunk elő olyan színes képeket, amelyekben egy-egy színcsatornát konstans nullával helyettesítünk. Elvileg három ilyen kép létezik.

2. Állítsunk elő olyan színes képeket, amelyekben két-két színcsatornát konstans nullával helyettesítünk. Elvileg három ilyen kép létezik.

3. Permutáljuk meg a színes kép színcsatornáit. Elvileg hat ilyen kép létezik, amiből az egyik az eredeti. A BGR komponensekből rakjunk össze olyan képet, amelyben pl. az eredeti vörös adatait adjuk meg kéknek, a zöldet alkalmazzuk a vörös helyett és a kéket a zöld helyett.

4. Állítsunk elő olyan színes képeket, amelyekben az egyik színkomponenst helyettesítjük a saját negatívjával. Elvileg három ilyen kép létezik. (Használhatjuk a ~ operátort pl. Mat ib = ~b;)

5. Állítsunk elő azt a színes képet, amelyikben az Y komponenset helyettesítjük a saját negatívjával. YCrCb kódolást kell ehhez használni, mint a hisztogram kiegyenlítésnél.

```
cvtColor(im, result, COLOR_BGR2YCrCb);
```

```
cvtColor => BGR2YCrCb
split => Y Cr Cb
merge => ~Y Cr Cb
cvtColor => YCrCb2BGR
```

6. Tegyük fel az eredeti kép negatívját is a nagy képre.

Az így előállított képeket helyezzük el mozaikszerűen egy nagyobb képen.
ROI képre való ráhelyezése (C++ -ban egyszerűen Rect -et kell létrehozni)

```
void showMyImage(Mat &imBig, Mat &im, int& index) {
    im.copyTo(imBig(Rect((index % 6) * (im.cols), (index / 6) * (im.rows), im.cols, im.rows)));
    imshow("Ablak", imBig);
    index = (index + 1) % 18;
    waitKey();
}

showMyImage(imBig, result, index);
```