1. Napište funkci discnorm(f, L), která lineárně transformuje hodnoty jasu v obrázku f na celočíselné hodnoty od 1 do L:

```
function g = discnorm(f, L)
```

— g je transformovaný obrázek (datový typ double)

— implicitní hodnota L = 256

— nápověda: nargin

2. Napište funkci histogram(f, L), která pro obrázek f vypočte histogram h (tj. vektor četností úrovní jasu) pro hodnoty {1, ..., L}:

```
function h = histogram(f, L)
```

— h je sloupcový vektor délky L

— implicitní hodnota L = 256

— nápověda: find() nebo histc()

3. Napište funkci smoothhist(h, R), která histogram h vyhladí konvolucí s obdélníkem šířky 2*R+1:

```
function h2 = smoothhist(h, R)
```

Funkci vyzkoušejte mj. na obrázku temple.png. Vykreslete si původní a vyhlazený histogram obrázku a porovnejte je.

— pozor při normalizaci masky na okrajích

— nápověda: bar(), title()

4. Napište funkci bandthreshold(f, t1, t2), která oprahuje obrázek f tak, že pixely s hodnotami jasu v intervalu [t1, t2] označí jako objekt, ostatní jako pozadí:

```
function g = bandthreshold(f, t1, t2)
```

Funkci vyzkoušejte mj. na obrázku saturn.png. Vyzkoušejte nejdříve použít pouze jeden práh. Vyzkoušejte různé hodnoty prahů.

— g je oprahovaný (binární) obrázek

— implicitní hodnota t2 = maximální jas (jednoduché prahování)

— nápověda: hist(), zobr()

5. Napište funkci ptile(f, p), která oprahuje obrázek f tak, aby práh t rozděloval plochu histogramu v poměru p : (1-p):

```
function [g, t] = ptile(f, p)
```

Funkci vyzkoušejte mj. na obrázku paragraph.png. Zkuste najít optimální hodnotu p.

— g je oprahovaný obrázek, t vypočtená hodnota prahu

— p je z intervalu [0, 1]

— nechte funkci vykreslovat kumulativní histogram

— nápověda: cumsum(), find()

6. Napište funkci otsu(f), která

i. lineárně transformuje hodnoty jasu v obrázku f na celočíselné hodnoty od 1 do L

ii. pomocí Otsu algoritmu nalezne práh t, kterým obrázek oprahuje

iii. vykreslí graf hodnot between-class variance

```
function [g, t] = otsu(f, L)
```

Funkci vyzkoušejte na několika obrázcích, např. kruhy.png, coins.png nebo rice.png. Vykreslete si také histogram obrázku f a hodnotu spočítaného prahu t.

— g je oprahovaný obrázek, t vypočtená hodnota prahu

— implicitní hodnota L = 256

— nápověda: Otsu-paper.pdf

7. Napište funkci localthreshold(f, n1, n2), která obrázek f rovnoměrně rozdělí na n1-krát-n2 obdélníků, na kterých ho lokálně oprahuje pomocí Otsu metody:

```
function [g, t] = localthreshold(f, n1, n2)
```

Funkci vyzkoušejte mj. na obrázku rice.png. Zkuste najít optimální počet obdélníků pro tento obrázek a porovnejte výsledek s globálním prahováním.

— g je lokálně oprahovaný obrázek

— t je matice vypočtených lokálních prahů

— nápověda: floor(), close()

8. Napište funkci semithreshold(f, t), která pixely s hodnotou jasu menší než práh t označí jako pozadí, zatímco hodnoty ostatních pixelů nezmění:

```
function [g, t, m] = semithreshold(f, t)
```

Funkci vyzkoušejte např. na obrázku moon.png.

— g je polooprahovaný obrázek, t hodnota prahu, m prahovací maska

— není-li t zadáno, vypočte se Otsu algoritmem

9. V barevném obrázku sloup.png přebarvěte pixely oblohy na bílo.

— nápověda: repmat()

10. Z obrázku textura.png vysegmentujte oblast textury v půlkruhu dole. Jako příznak zkuste použít lokální rozptyl.

— nápověda: conv2()