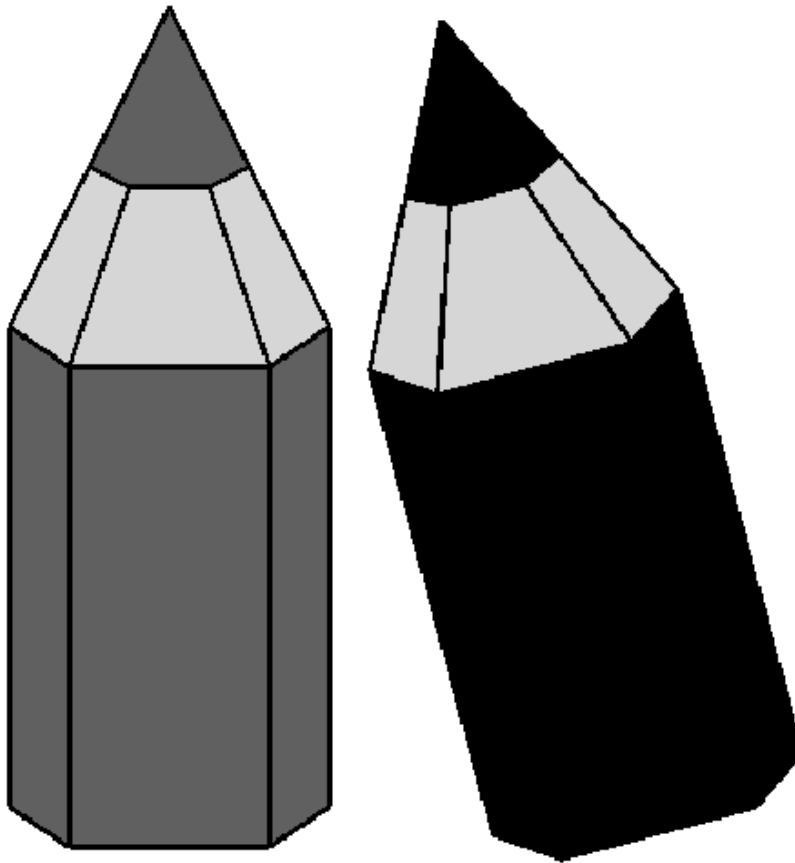


## Seminář 7

### Redundance - kódování

```
I1 = rgb2gray(imread('red_kodovani.png'));  
figure, imshow(I1);
```



```
[M,N] = size(I1);
```

### Počet barev v obraze

```
barvy = unique(I1)
```

```
barvy = 4x1 uint8 column vector  
    0  
   96  
  214  
  255
```

### Pravděpodobnost každé barvy (histogram)

```
n = imhist(I1);
p = n/(M*N);
pravdepodobnost = p(double(barvy)+1)
```

```
pravdepodobnost = 4×1
    0.2555
    0.2029
    0.1098
    0.4317
```

## Kódování - 8 kód

počet bitů použitých ke kódování

```
l1 = 8*ones(256,1);

l1_avg = sum(l1.*p);
display(l1_avg);
```

```
l1_avg = 8
```

## Kódování nestejně dlouhými kódy

- 0 : 01
- 96 : 000
- 214 : 010
- 255 : 1

```
l2 = zeros(256,1);
l2(1)=2;
l2(97) = 3;
l2(215) = 3;
l2(256) = 1;

l2_avg = sum(l2.*p);
display(l2_avg);
```

```
l2_avg = 1.8810
```

## Komprese a relativní redundance

```
b1 = M*N*l1_avg;
b2 = M*N*l2_avg;

% komprese
C = b1/b2;
display(C);
```

```
C = 4.2530
```

```
% relativní redundance
```

```
R = 1-(1/C);  
display(R);
```

```
R = 0.7649
```

## Úkol 1

Spočítejte kompresi a relativní redundanci, pokud každou barvu v obrázku 'red\_kodovani.png' (I1) kódujeme 2 bity.

### Huffmanovo kódování

```
vstup = 'barbaraabarboraubaru'
```

```
vstup =  
'barbaraabarboraubaru'
```

```
delka_vstupu = numel(vstup)
```

```
delka_vstupu = 20
```

```
A = unique(vstup)
```

```
A =  
'aboru'
```

```
Acell = cellstr(A)';  
p_A = arrayfun(@(x)sum(vstup==x), A)/numel(vstup);  
n = size(A,2)
```

```
n = 5
```

```
% kodova abeceda
```

```
B = {'0' '1'}
```

```
B = 1x2 cell  
'0'      '1'
```

```
m = size(B,2)
```

```
m = 2
```

```
kody = Huffman(Acell, B, p_A, true);  
table(A',kody', 'VariableNames',{'Znak', 'Kod'})
```

```
ans = 5x2 table
```

	Znak	Kod
1	a	'10'
2	b	'11'
3	o	'011'

	Znak	Kod
4	r	'00'
5	u	'010'

## Kódování

```
vystup = Huffman_enc(vstup, A, kody)
```

```
vystup =  
'1110001110001010111000110110010010111000010'
```

## Huffmanovo kódování na obrázku

- 0 : a
- 96 : b
- 214 : c
- 255 : d

```
I = imread('red_kodovani.png');  
[m, n] = size(I);  
vstup_pom = I(:);  
vstup(vstup_pom==0) = 'a';  
vstup(vstup_pom==96) = 'b';  
vstup(vstup_pom==214) = 'c';  
vstup(vstup_pom==255) = 'd';  
  
A = 'abcd';  
Acell = cellstr(A)';  
p_A = arrayfun(@(x)sum(vstup==x), A)/numel(vstup);  
n = size(A,2)
```

```
n = 4
```

```
% kodova abeceda  
B = {'0' '1'}
```

```
B = 1x2 cell  
'0'      '1'
```

```
m = size(B,2)
```

```
m = 2
```

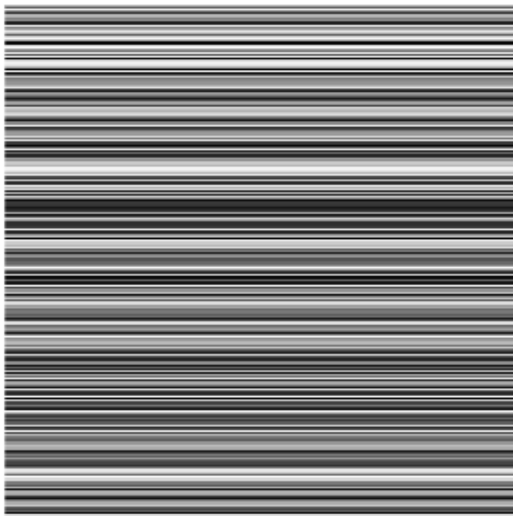
```
kody = Huffman(Acell, B, p_A, true);  
table(A',kody', 'VariableNames',{'Znak', 'Kod'})
```

```
ans = 4x2 table
```

	Znak	Kod
1	a	'11'
2	b	'100'
3	c	'101'
4	d	1×1 cell

## Prostorová redundance

```
I2 = imread('red_prostor.png');
figure, imshow(I2);
```



```
[M,N] = size(I2);
```

## Histogram

```
figure, imhist(I2);
```



```
vstup = 'aaaabbcccaabb'
```

```
vstup =  
'aaaabbcccaabb'
```

```
% kodovani  
enc = RLEenc(vstup)
```

```
enc =  
'4a2b3c2a2b'
```

```
dec = RLEdec(enc)
```

```
dec =  
'aaaabbcccaabb'
```

Je možné, aby bylo toto kódování neefektivní?

### RLE aplikované na obrázek

```
I = imread('prikladLena.png');  
[m, n] = size(I);  
vstup_pom = I(:);  
vstup(vstup_pom) = 'w';  
vstup(vstup_pom==0) = 'b';
```

Délka vstupu. Jedná se o binární obrázek, každý pixel je možné kódovat 1 bitem.

```
delka_vstup = m*n
```

```
delka_vstup = 262144
```

Délka kódu. Každé číslo je kódováno jedním bytem a znak jedním bitem.

```
[enc delka_kod] = RLEenc(vstup);  
delka_kod
```

```
delka_kod = 80370
```

## Úkol 3

*Jaká je komprese a relativní redundance výše zmíněného kódu (získaného zjednodušenou RLE kompresí) pro obrázek 'prikladLena.png'.*

### Nerelevantní informace

```
I3 = imread('red_info.png');
```

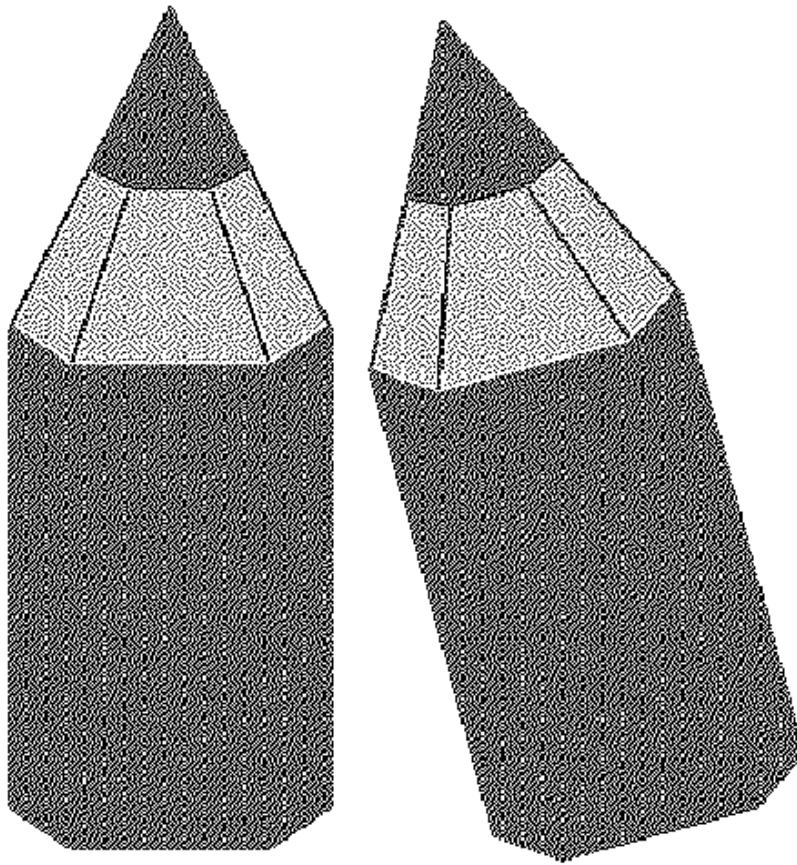
```
figure, imshow(I3);
```



### **Obrázek s roztaženým kontrastem**

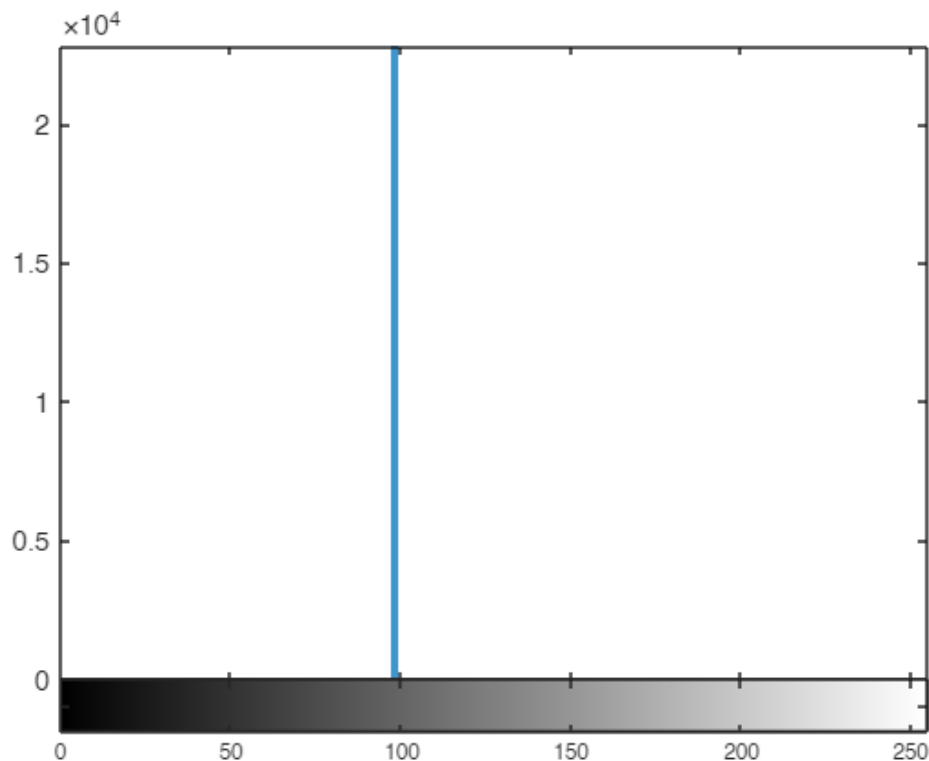
```
figure, imshow(I3,[]);
```





### Histogram

```
figure, imhist(I3,256);
```



## Úkol 4

Spočítejte kompresi a relativní redundanci při kódování obrázku 'red\_info.png' jednou hodnotou (každý pixel má hodnotu 125).

## JPEG komprese

```
I = rgb2gray(imread('pastelky.png'));

imwrite(I,'p100.jpg','Quality', 100);
imwrite(I,'p80.jpg','Quality', 80);
imwrite(I,'p50.jpg','Quality', 50);
imwrite(I,'p10.jpg','Quality', 10);
imwrite(I,'p5.jpg','Quality', 5);

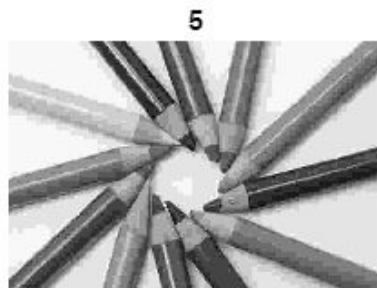
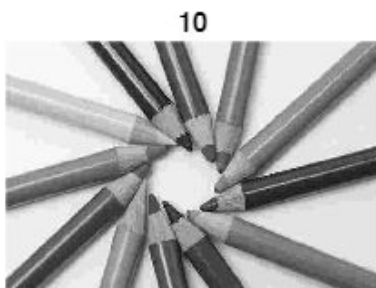
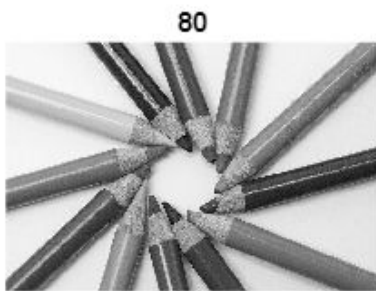
I1 = imread('p100.jpg');
I2 = imread('p80.jpg');
I3 = imread('p50.jpg');
I4 = imread('p10.jpg');
I5 = imread('p5.jpg');

figure,
subplot(2,2,1), imshow(I2);
title("80");
subplot(2,2,2), imshow(I3);
```

```

title("50");
subplot(2,2,3), imshow(I4);
title("10");
subplot(2,2,4), imshow(I5);
title("5");

```



```

p100 = imfinfo('p100.jpg');
fprintf("Obrazek p100 ma velikost %i byte", p100.FileSize);

```

Obrazek p100 ma velikost 136317 byte

```

p5 = imfinfo('p5.jpg');
fprintf("Obrazek p5 ma velikost %i byte", p5.FileSize);

```

Obrazek p5 ma velikost 6123 byte

Příklad JPEG komprese ze slidů

```

puvodni = uint8([139 144 149 153 155 155 155 155;
144 151 153 156 159 156 156 156;
150 155 160 163 158 156 156 156;
159 161 162 160 160 159 159 159;
159 160 161 162 162 155 155 155;
161 161 161 161 160 157 157 157;
162 162 161 163 162 157 157 157;

```

```

162 162 161 161 163 158 158 158]);

puvodni = imresize(puvodni,[160,160],"nearest");

novy = uint8([142 144 147 150 152 155 155 155;
149 150 153 155 156 157 156 156;
157 158 159 161 161 160 159 158;
162 162 163 163 162 160 158 157;
162 162 162 162 161 158 156 155;
160 161 161 161 160 158 156 154;
160 160 161 162 161 160 158 157;
160 161 163 164 164 163 161 160]);

novy = imresize(novy,[160,160],"nearest");

figure, imshow(puvodni);

```



```
figure, imshow(novy);
```



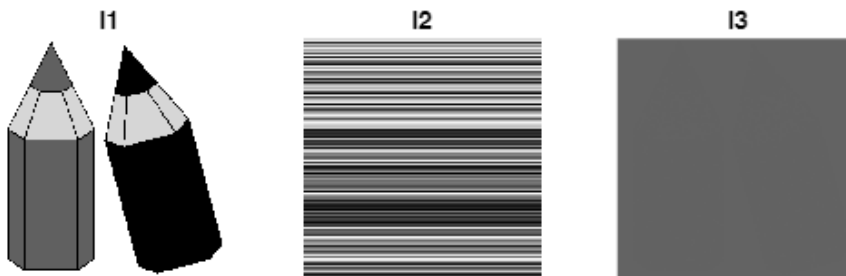
```

% imwrite(puvodni,'pr_puvodni.png');
% imwrite(novy,'pr_novy.png');

```

## Entropie

```
I1 = imread('red_kodovani.png');  
figure,  
subplot(1,3,1), imshow(I1);  
title("I1");  
I2 = imread('red_prostor.png');  
subplot(1,3,2), imshow(I2);  
title("I2");  
I3 = imread('red_info.png');  
subplot(1,3,3), imshow(I3);  
title("I3");
```



```
J1 = entropy(I1);  
J2 = entropy(I2);  
J3 = entropy(I3);
```

```
display(J1);
```

```
J1 = 1.8431
```

```
display(J2);
```

```
J2 = 8
```

```
display(J3);
```

J3 = 0.8989

## Měření kvality komprese

### Mean-squared error

```
I = rgb2gray(imread('pastelky.png'));  
I_noise = imnoise(I, 'salt & pepper', 0.02);  
  
figure,  
subplot(1,2,1), imshow(I);  
subplot(1,2,2), imshow(I_noise);
```



```
% immse = Mean-Squared Error.  
mse = immse(I, I_noise);  
display(mse);
```

mse = 449.3124

```
% root mean-squared error  
  
rmse = sqrt(immse(I, I_noise));  
display(rmse);
```

rmse = 21.1970

## Úkol 5

Spočítejte chybu pro obrázek 'red\_info.png' a jeho úpravu, kdy jsou všechny pixely rovny hodnotě 99.

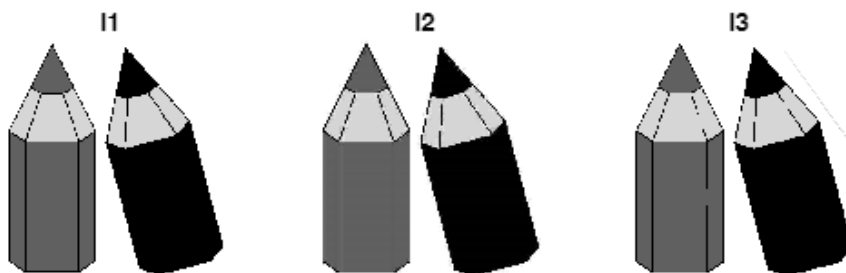
### Subjektivní hodnocení

```
I1 = rgb2gray(imread('kriterium1.png'));  
I2 = rgb2gray(imread('kriterium2.png'));
```

Warning: PNG library warning: iCCP: cHRM chunk does not match sRGB.

```
I3 = rgb2gray(imread('kriterium3.png'));
```

```
figure,  
subplot(1,3,1), imshow(I1);  
title('I1');  
subplot(1,3,2), imshow(I2);  
title('I2');  
subplot(1,3,3), imshow(I3);  
title('I3');
```



```
mse1 = immse(I1, I2);  
fprintf("Obrázek I2 má mse rovnu %i", mse1);
```

Obrázek I2 má mse rovnu 3.180263e+01

```
mse2 = immse(I1, I3);
fprintf("Obrazek I3 ma mse rovnu %i", mse2);
```

Obrazek I3 ma mse rovnu 2.721793e+01

## Pomocné funkce

```
function [enc, delka] = RLEenc(vstup)
    % kodovani
    enc = '';
    r = 0;
    delka = 0;
    for i = 1 : size(vstup,2)
        a = vstup(i);
        if r == 0
            x = a;
            r = 1;
        else
            if a == x
                r = r + 1;
            else
                enc = [enc num2str(r) x];
                x = a;
                r = 1;
                delka = delka + 9; %pocet je 1 byte, znak 1 bite
            end
        end
    end
    enc = [enc num2str(r) a];
    delka = delka + 9;
end

function dec = RLEdec(enc)
    % dekodovani
    r = 0;
    dec = '';
    for i = 1 : 2 : size(enc,2)
        n = str2num(enc(i));
        znak = enc(i+1);
        dec = [dec repmat(znak,[1 n])];
    end
end

function kody = Huffman(A, B, p_A, prvni)
    n = size(A,2);
    m = size(B,2);
    kody = cell(1,n);
    if n <= m
        for i = 1 : n
```



```

        kody{i} = B(i);
    end
else
    if prvni
        m2 = mod(n-2, (m-1)) +2;
    else
        m2 = m;
    end
    [p_A_sort,indexy] = sort(p_A, 'descend');
    A2 = A(indexy);
    A3 = A2(1 : n - m2);
    p_A3 = p_A_sort(1 : n - m2);
    p_A3 = [p_A3 sum(p_A_sort(n - m2+1:end))];
    A3 = [A3 [A2{n - m2+1:end}]];
    kody2 = Huffman(A3, B, p_A3, false);

    for i = 1 : n - m2
        kody{indexy(i)} = kody2{i};
    end
    for i = 1 : m2
        kody{indexy(n - m2 + i)} = [char(kody2{end}) B{i}];
    end
end
end

function vystup = Huffman_enc(vstup, znaky,kody)
    vystup = [];
    for i = 1 : size(vstup,2)
        index = find(znaky==vstup(i),1,'first');
        vystup = [vystup,kody{index}];
    end
end
end

```