DIGITALNA OBRADA SLIKE

ČAS 1 – OSNOVE MATLABA U OBRADI SLIKE

Opšte informacije

asistent: Dragomir El Mezeni mail: elmezeni@el.etf.rs

Časovi u blokovima od po 2 časa, svake druge nedelje

Termin održavanja vežbi **utorak 8-10** u sali 70 (dok ne stignu računari biće u računskom centru)

Opšte informacije

Način polaganja ispita: pismeni ispit 40% + domaći zadaci 60%

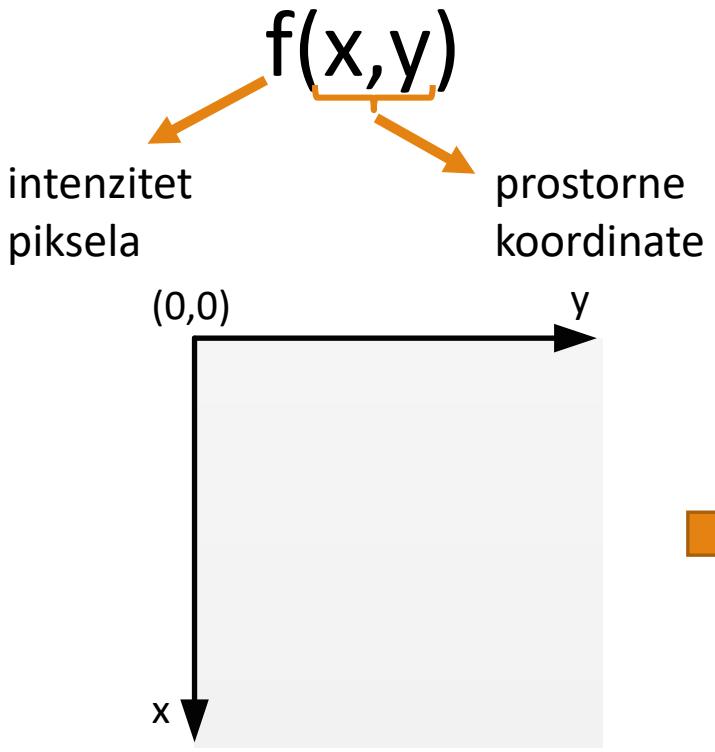
Ukupno 4 domaća zadataka u toku semestra

Domaći se uglavnom rade korišćenjem Matlab-a (+ određeni delovi u C-u)

Rešenja domaćih zadataka je potrebno predati do naznačenog datuma, svaki dan kašnjenja povlači -10% poena

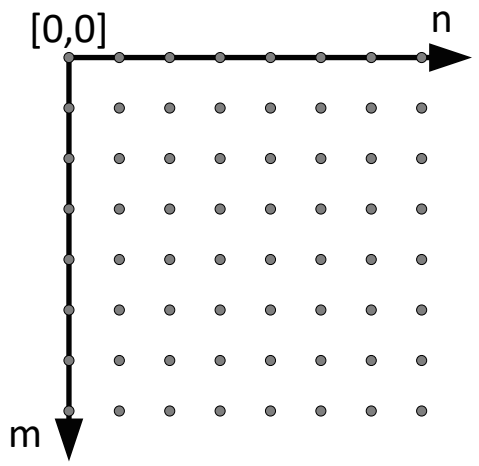
Na kraju semestra biće održana odbrana domaćih zadataka

Predstavljanje digitalnih slika



MxN matrica piksela

C: 0, ..., M-1 0, ..., N-1
MATLAB: 1, ..., M 1, ..., N



$$f = \begin{bmatrix} f(1,1) & f(1,2) & \cdots & f(1,N) \\ f(2,1) & f(2,2) & \cdots & f(2,N) \\ \vdots & \vdots & \ddots & \vdots \\ f(M,1) & f(M,2) & \cdots & f(M,N) \end{bmatrix}$$

Učitavanje sirovih slika

Slike mogu biti upisane u tekstualne ili binarne fajlove bez zaglavlja. Slike koje sadrže samo podatke obično se nazivaju sirove slike. Postupak za čitanje podataka iz ovih fajlova je vrlo sličan kao u C-u. Podaci se učitavaju u jedan niz kolona. Nakon toga je potrebno promeniti dimenzije ovog niza u odgovarajuću matricu kako bi učitana slika dobila smisao. Kako se pikseli slike upisuju red po red u fajl a po očitavanju smeštaju u posebne kolone potrebno je transponovati očitanu sliku kako bi se dobio korektan prikaz.

Primer tekstualna datoteka:

```
fid = fopen('im1.dat', 'r');
I = fscanf(fid, '%d');
fclose(fid);

figure
J = reshape(I, [11, 38])'; imshow(J);
J = reshape(I, [38, 11])'; imshow(J);
J = reshape(I, [19, 22])'; imshow(J);
```

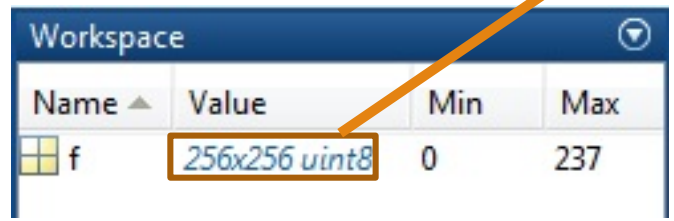
Primer binarna datoteka:

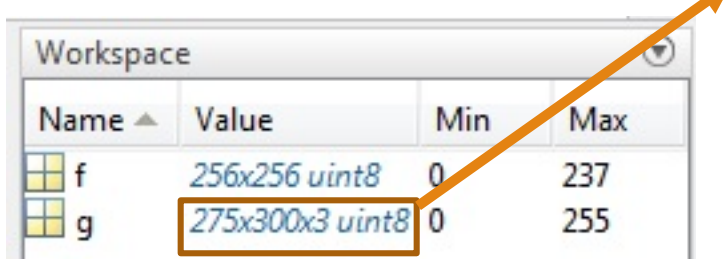
```
fid = fopen('im2.bin', 'rb');
I = uint8(fread(fid, 'uint8'));
fclose(fid);

figure
J = reshape(I, [480, 160])'; imshow(J);
J = reshape(I, [240, 320])'; imshow(J);
J = reshape(I, [320, 240])'; imshow(J);
```

Učitavanje slika

`imread('naziv_slike')`

```
Primer:

f = imread('einstein.tif');

[height, width] = size(f);

g = imread('cartman.png');

[height, width, num_planes] = size(g)
```





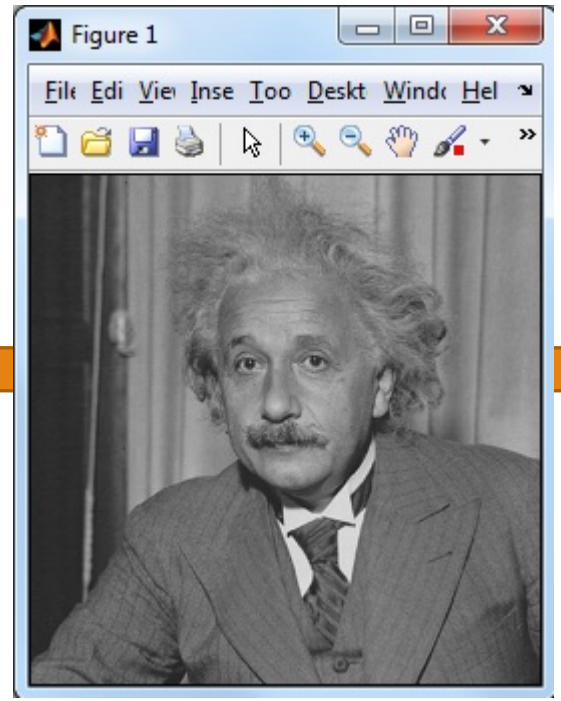
monohromatska slika
256 redova, 256 kolona
pikseli celi brojevi veličine 8 bita
opseg vrednosti [0, 255]

slika u boji (3 komponente boje)
275 redova, 300 kolona
pikseli celi brojevi veličine 8 bita
opseg vrednosti [0, 255]

Prikaz slika

`imshow(matrica_slike)`

```
Primer:

imshow(f);

figure; imshow(g);
```

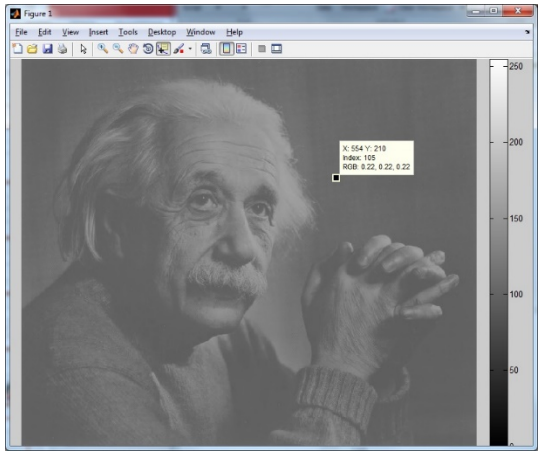


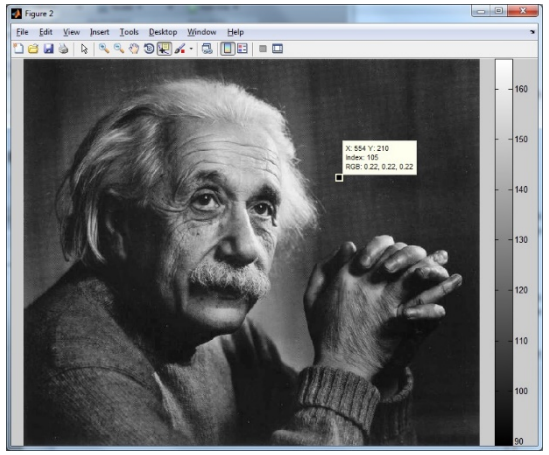

```
[g, ~, alpha] = imread('cartman.png');
g_fig = imshow(g);
set(g_fig, 'AlphaData', alpha);
```

Prikaz slika

imshow(matrica_slike, [low, high])
imshow(matrica_slike, [])

```
f = imread('einstein_lowcontrast.tif');
imshow(f); disp(min(f(:))); disp(max(f(:)));
figure; imshow(f, []);
```





Svi pikseli čiji je intenzitet manji od ***low*** prikazuju se crnom bojom.

Svi pikseli čiji je intenzitet veći od ***high*** prikazuju se belom bojom.

Prikaz piksela čiji intenzitet je između ove dve vrednosti se skalira

Ako se ne navedu granice onda se prikaz skalira između minimalne i maksimalne vrednosti piksela na slici

Vrednosti intenziteta samih piksela se ne menjaju! Proveriti ovo sa Data Cursor alatom.

Upisivanje slike u fajl

imwrite(matrica_slike, 'naziv_fajla')

imwrite(matrica_slike, 'naziv_fajla', 'quality', q)

```
Primer:

f = imread('einstein.tif');

imwrite(f, 'einstein.png');

imwrite(f, 'einstein_q10.jpg', 'quality', 10);
imwrite(f, 'einstein_q50.jpg', 'quality', 50);
imwrite(f, 'einstein_q90.jpg', 'quality', 90);
```

Klase u MATLAB-u

Naziv	Opis
`double`	Realni brojevi u prikazu sa pokretnom tačkom dvostruke preciznosti. Opseg $\pm 10^{308}$. Veličina 64 bita.
`single`	Realni brojevi u prikazu sa pokretnom tačkom jednostruke preciznosti. Opseg $\pm 10^{38}$. Veličina 32 bita.
`uint8`	Neoznačeni celi brojevi veličine 8 bita. Opseg [0, 255].
`uint16`	Neoznačeni celi brojevi veličine 16 bita. Opseg [0, 65535].
`uint32`	Neoznačeni celi brojevi veličine 32 bita. Opseg [0, 4294967295].
`int8`	Označeni celi brojevi veličine 8 bita. Opseg [-128, 127].
`int16`	Označeni celi brojevi veličine 16 bita. Opseg [-32768, 32767].
`int32`	Označeni celi brojevi veličine 32 bita. Opseg [-2147483648, 2147483647].
`char`	Karakteri veličine 16 bita.
`logical`	Logičke vrednosti iz skupa 0 i 1. Veličina 8 bita.

Konvertovanje između klasa

B = naziv_klase(A)

```
Primer:

f = [-1.2, 0.3; 2.5, -0.4]
uint8(f)
int8(f)
im2uint8(f)
mat2gray(f)
k = im2bw(f)
K = im2bw(f, 0.1)

g = uint8([120, 10; 280, -5])
double(g)
im2double(g)
```

Komanda	Izlazni tip	Ulazni tip
im2uint8	uint8	logical, uint8, uint16, int16, single, double
im2uint16	uint16	logical, uint8, uint16, int16, single, double
im2double	double	logical, uint8, uint16, int16, single, double
im2single	single	logical, uint8, uint16, int16, single, double
mat2gray	double$\in$[0,1]	logical, uint8, uint16, int16, uint32, int32, single, double
im2bw	logical	uint8, uint16, int16, single, double

Indeksiranje

v(indeks)
v(start:korak:stop)
v([niz indeksa])

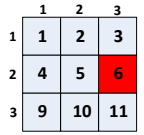
Naredba:	Rezultat
`v = [1 3 5 7 9 11]`	`1 3 5 7 9 11`
`v(3)`	`5`
`v(2:5)`	`3 5 7 9`
`v(4:end)`	`7 9 11`
`v(1:2:5)`	`1 5 9`
`v(end:-2:2)`	`11 7 3`
`v([1 2 6])`	`1 3 11`

m(red, kolona)
m(linearni indeks)

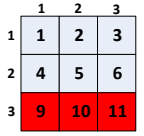
Naredba i rezultat:

```
m = [1 2 3; 4 5 6; 9 10 11]
```

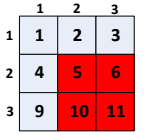
m(2,3)



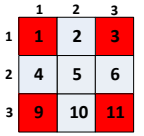
m(3,:)

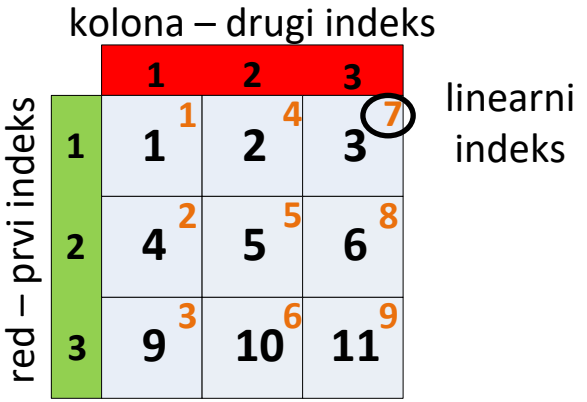


m(2:end, 2:3)



m([1 3], [1 3])

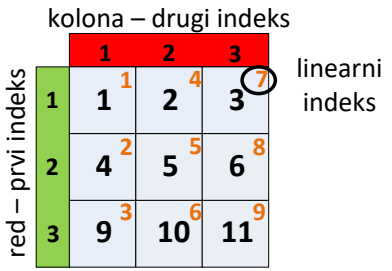




Naredba i rezultat:

```
m(7)    3

m(1:5)  1 4 9 2 5

m(end-1:-2:6)  6 10
```

Indeksiranje – konverzija koordinata



linearni_indeks = sub2ind(size(m), red, kolona)
[red, kolona] = ind2sub(size(m), linearni_indeks)

Naredba i rezultat:

[red, kolona] = ind2sub(size(m), 7) → red = 1 kolona = 3

[red, kolona] = ind2sub(size(m), 1:5) → red = 1 2 3 1 2 kolona = 1 1 1 2 2

[red, kolona] = ind2sub(size(m), end-1:-2:6) → error!!!

[red, kolona] = ind2sub(size(m), size(m,1).*size(m,2)-1:-2:6) → red = 2 3 kolona = 3 2

linearni_indeks = sub2ind(size(m), [1,1,2,3], [2,3,3,2]) → linearni_indeks = 4 7 8 6

Logičko indeksiranje

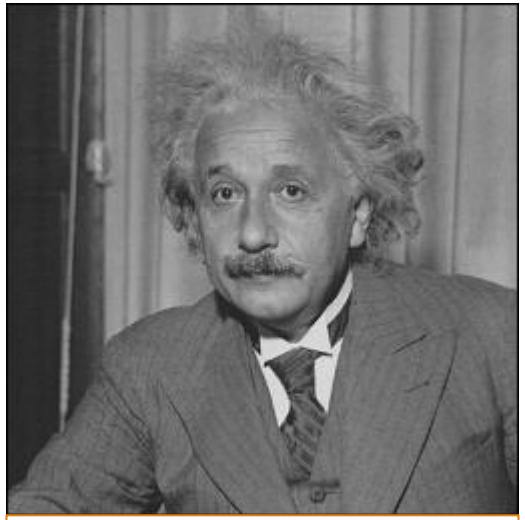
m(mask)

Matrica (ili niz) se može adresirati logičkom matricom istih dimenzija. Selektovana su ona polja originalne matrice na čijim mestima u indeksnoj matrici se nalazi logička jedinica.

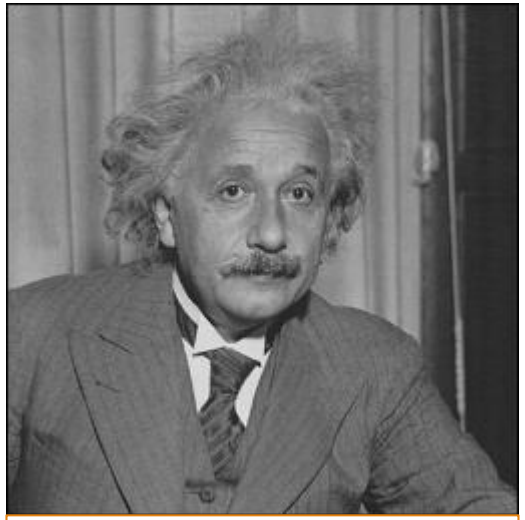
```
Primer:
f = imread('einstein.tif');
load einstein_mask.mat

figure; imshow(f);
figure; imshow(mask);
f(mask) = 128;
figure; imshow(f);
```

Logičko adresiranje se može koristiti za selektovanje svih piksela koji ispunjavaju određeni uslov. Kako je rezultat relacionh i logičkih operatora logička matrica to se na primer selektovanje svih piksela čiji je intenzitet veći od 50 može realizovati kao: **m(m>50)**

Indeksiranje - primeri

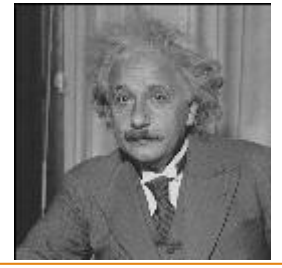


`f`



`f(:, end:-1:1)`

`f(end:-1:1, :)`



`f(1:2:end, 1:2:end)`

`f(76:149, 95:157)`

Važne ugrađene matrice

Komanda	Opis
`zeros(M,N)`	Matrica nula dimenzija MxN
`ones(M,N)`	Matrica jedinica dimenzija MxN
`true(M,N)`	Logička matrica jedinica dimenzija MxN
`false(M,N)`	Logička matrica nula dimenzija MxN
`rand(M,N)`	Matrica slučajnih brojeva uniformno raspoređenih u intervalu [0,1]
`randn(M,N)`	Matrica slučajnih brojeva normalne raspodele, srednje vrednosti 0 i varijanse 1

Operatori u Matlabu

Tip	Operatori
Aritmetički - vektorski	+ - * / \ ^ '
Aritmetički - skalarni	.+ .- .* ./ .\ .^ .'
Relacioni	< <= > >= == ~=
Logički - vektorski	& (and) | (or) ~ (not)
Logički - skalarni	&& ||

Neke osnovne operacije i ugrađene funkcije

```
Primer:

f = [-1.2, 0.3; 2.5, -0.4]
m = [1 2 ; 3 4; 5 6]

k = [f m]  error!
k = [f; m]
k = [f m(2:end, :)]
k = [f' m']

size(k)
size(k, 1)
```

```
mean(k)
mean(k, 2)
mean(k(:))

max(f)
max(f, 2)
max(f, [], 2)
max(max(f))
max(f(:))

f*m   error!
f.*m  error!
f*m'
f.*m(1:2,:)
```

Kontrola toka

```
if logički_izraz1
        komande1
elseif logički_izraz2
        komande2
else
        komande3
end
```

```
for indeks = start:korak:stop
        komande
end
```

```
for indeks = niz_indeksa
        komande
end
```

```
while logički_izraz
        komande
end
```

```
switch izraz_koji_se_testira
        case vrednost1
                komande
        case vrednost2
                komande

        case vrednostN
                komande
        otherwise
                komande
end
```

Ćelije i strukture

Ćelije i strukture se koriste za skladištenje raznorodnih podataka unutar jedne promenljive.

Ćelija primer:

```
f = imread('einstein.tif');

slika = {f, 'einstein.png', size(f), mean(f(:))}

figure; imshow(slika{1})

slika{3}(1)
```

Struktura primer:

```
f = imread('einstein.tif');

slika = struct('data', f, ...
          'file_name', 'einstein.png',...
               'size', size(f),...
               'mean', mean(f(:)))

figure; imshow(slika.data)

slika.size(1)
```

Interaktivni upis i ispis podataka

disp(promenljiva)
t = input('Tekst: ')

Primer:

```
a = input('Prvi argument: ');
b = input('Drugi argument: ');
disp(['Zbir: ', num2str(a+b)]);

disp([1 2; 3 4])
```

Funkcije i skripte

```
function [ output arugments list ] = func_name( input arguments list)
%FUNC_NAME One line description
%               More detailed description.
% References:
%               Reference scientific papers or documents.
% Syntax:
%               How this function should be called.
% Inputs:
%               Description of input arguments.
% Outputs:
%               Description of output arguments.
% Example:
%               Example of function usage.
% See also:  other relevant files
%
% Created on:    date (author)
% Last revision: date (author)

%------------- BEGIN CODE --------------

Actual code goes here.
```

```
%One line description
%               More detailed description.
% References:
%               Reference scientific papers or documents.
% See also:  other relevant files
%
% Created on:    date (author)
% Last revision: date (author)

%------------- BEGIN CODE --------------

Actual code goes here.
```

Domaći zadatak – prvi deo

U fajlu **d1img** se nalazi slika nepoznatih dimenzija. Odrediti da li je u pitanju binarna ili tekstualna datoteka. Potom pročitati sliku na adekvatan način i odrediti dimenzije tako da se dobije smislen sadržaj. Rešenje ovog dela domaćeg je potrebno navesti na početku glavnog programa. Potrebno je da ovaj deo koda pročita sliku iz navedenog fajla i upiše je u drugi fajl u jpeg formatu. U izveštaju navesti primer izlazne slike kada se dimenzije podese ispravno kao i jedan primer kada se dimenzije podese pogrešno.

Domaći zadatak – drugi deo

Potrebno je odabrati neku fotografiju na kojoj se nalazi nekoliko ljudi kojima se vidi lice. Fotografiju je potrebno učitati u MATLAB i, u slučaju da je originalna slika u boji, pomoću naredbe **rgb2gray** konvertovati u monohromatsku. Prikazati fotografiju i pomoću opcije ***Data cursor*** za minimum dva lica na slici odrediti koordinate gornjeg levog i donjeg desnog ćoška pravougaonika koji obuhvata lice. Kreirati strukturu ***face_location*** koja sadrži dva polja ***top_left*** i ***bottom_right*** koja sadrže koordinate ćoškova pravougaonika koji opisuje lice. Kako na slici postoji više lica potrebno je formirati niz sa elementima ***face_location*** strukture.

Napisati funkciju ***protect_face*** koja za zadatu lokaciju lica i veličinu kvadrata popločava region lica kvadratima zadatih dimenzija. Svi pikseli unutar jednog kvadrata imaju istu vrednost i ona odgovara srednjoj vrednosti piksela koji se nalaze na lokaciji tog kvadrata u originalnoj slici. U slučaju da širina ili visina regiona lica nije umnožak veličine kvadrata na ivicama regiona lica će se javiti pravougaonici.

Domaći zadatak – drugi deo

Primer poziva funkcije za jedno od lica sa slike**I (corleone.jpg)**
kojom se lice popločava kvadratima širine 40 piksela:

```
face_location(1).top_left = [151, 723];
face_location(1).bottom_right = [739, 1180];

If = protect_face(I, face_location, 40);
```

Domaći zadatak

Napisati glavni program ***domaci1_gg_bbb.m*** (gg - godina, bbb - broj indeksa) u koji su upisane koordinate svih lica koja se štite sa slike (minimum 2) i koji po pokretanju štiti svako lice prekrivanjem kvadratima odgovarajuće širine (odabrati neku razumnu vrednost) i čuva tako modifikovanu sliku u fajl ***nazivslike_protected.jpg***.

Potrebno je takođe sačuvati sve pravougaonike koji obuhvataju lica pre modifikacije u sirove binarne fajlove ***face1.bin***, ***face2.bin***, .. ,***faceN.bin***.

Dakle uspešno napisan glavni program treba da po pokretanju generiše N sirovih binarnih slika(gde je N ukupan broj lica) koje predstavljaju regione lica pre modifikacija kao i celu sliku u jpeg formatu, kvaliteta 90, sa označenim regionima lica.

Izrada ovog domaćeg zadatka nije obavezna i ne nosi poene. Bez obzira na to preporučuje se studentima da urade ovaj domaći zadatak kako bi proverili stečeno znanje i uvežbali korišćenje osnovnih funkcionalnosti Matlaba u obradi slike.