[Lab 2] Lab 02: Basic MATLAB Programming

1. Variable Assignment

```
clc;

x = 10;

y = 2.5;

name = 'John';


disp(x);

disp(y);

disp(name);
```

2. Numeric Types

```
clc;

number1 = 5;

number2 = 3.14;


disp(number1);

disp(number2);
```

3. User Input and Output

```
number1 = input('Enter First Number:');

number2 = input('Enter Second Number:');
```

```matlab
disp(number1);

disp(number2);
```

4. Quadratic Expression Evaluation

```matlab
a = input('Enter a: ');

b = input('Enter b: ');

c = input('Enter c: ');

x = input('Enter x: ');


y = a * x^2 + b * x + c;

disp(['y = ' num2str(y)]);
```

5. Quadratic Equation Plot

```matlab
a = input('Enter a: ');

b = input('Enter b: ');

c = input('Enter c: ');


x = linspace(-10, 10, 100);

y = a * x.^2 + b * x + c;


plot(x, y, 'LineWidth', 2);

title('y = ax^2 + bx + c');

xlabel('x'); ylabel('y');

grid on;
```

6. Conditional Statements

```
x = 5; y = 10;


if x > y

    disp('x > y');

else

    disp('y > x');

end
```

7. Roots of Quadratic Equation

```
a = input('a: ');

b = input('b: ');

c = input('c: ');


delta = b^2 - 4*a*c;


if delta > 0

    root1 = (-b + sqrt(delta)) / (2*a);

    root2 = (-b - sqrt(delta)) / (2*a);

    fprintf('Root 1: %.4f\n', root1);

    fprintf('Root 2: %.4f\n', root2);

elseif delta == 0

    root = -b / (2*a);

    fprintf('Root: %.4f\n', root);

else
```

```matlab
    realPart = -b / (2*a);

    imaginaryPart = sqrt(abs(delta)) / (2*a);

    fprintf('Root 1: %.4f + %.4fi\n', realPart, imaginaryPart);

    fprintf('Root 2: %.4f - %.4fi\n', realPart, imaginaryPart);

end
```

8. Matrix Operations

```matlab
A = [1, 2, 3; 4, 5, 6; 7, 8, 9];

B = ones(3, 3);

C = zeros(2, 4);


disp('Matrix A:'); disp(A);

disp('Matrix B:'); disp(B);

disp('Matrix C:'); disp(C);


A_transpose = A';

disp('Transposed A:'); disp(A_transpose);


D = [10, 11, 12; 13, 14, 15; 16, 17, 18];

E = A + D;

disp('A + D:'); disp(E);
```

[Lab 3] Lab 03: Image Processing and Enhancement

1. Load and Display Image

```
I = imread("IMG_001.jpg");

imshow(I);
```

2. Convert RGB to Grayscale

```
gs = rgb2gray(I);

imshow(gs);
```

3. Display Histogram

```
imhist(gs);
```

4. Adjust Image Contrast

```
gsAdj = imadjust(gs);

imshowpair(gs, gsAdj, "montage");
```

5. Save and Reload Adjusted Image

```
imwrite(gsAdj, "adjusted_image.png");

Inew = imread("adjusted_image.png");

imshow(Inew);
```

6. Check Image Size and Color Channels

```
sz = size(I);
```

```
R = I(:,:,1);

G = I(:,:,2);

B = I(:,:,3);
```

7. Compare Two Images Side-by-Side

```
imshowpair(I, I2, "montage");
```

8. Contrast Histogram & Adjustment

```
imhist(gs2);

gs2Adj = imadjust(gs2);

imhist(gs2Adj);
```