

## Zad.1

---

```
clear variables
clc
close all

X = zeros(4,4);
X(1:2, 3:4) = 1;
X(3:4, 1:2) = 1;
figure(1);
imagesc(X);
%%
Y=zeros(64,64);
indeksy = mod(0:63, 4) + 1;
Y=X(indeksy,indeksy);
figure(2);
imagesc(Y);
%%
Z=[X, X ; X , X]; % 4x4
Z=[Z,Z ; Z, Z]; % 16x16
Z=[Z,Z ; Z, Z]; % 32x32
Z=[Z,Z ; Z, Z]; % 64x64
figure(3);
imagesc(Z);
```

---

### Komentarz:

W pierwszej sekcji utworzyłem matrycę zer o wymiarach 4x4 i dla odpowiednich indeksów podstawilem 1 tworząc tym samym szachownicę i wyświetlając ją w figure 1. W drugiej sekcji stworzyłem matrycę zer o wymiarach 64x64 oraz zmienną indeksy która jest wektorem poziomym o 64 kolumnach, a jego wartości to reszty z dzielenia kolejnych liczb do 63 przez 4 (+1, bo pierwszy indeks w matlabie to 1, a nie 0), aby wektor był wypełniony ciągiem 1,2,3,4,1,2,3,4,1,2 ... i tak dalej. Następnie macierz Y dla odpowiednich indeksów wkłada macierz X tworząc szachownicę potem wyświetloną w figure 2. W 3 sekcji utworzyłem szachownicę 64x64 innym sposobem. Tutaj użyłem sklejania macierzy aż do osiągnięcia zamierzonego wyniku (szybciej jednak by było z pętlą for). Wynik wyświetliłem w figure 3.

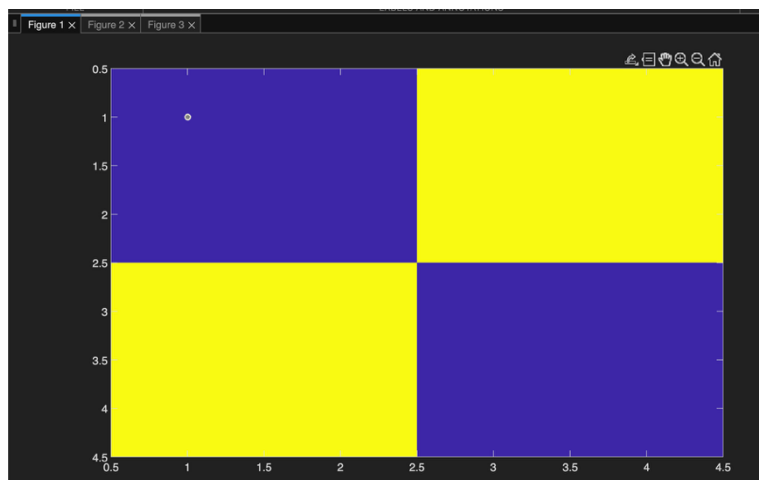


Figure 1

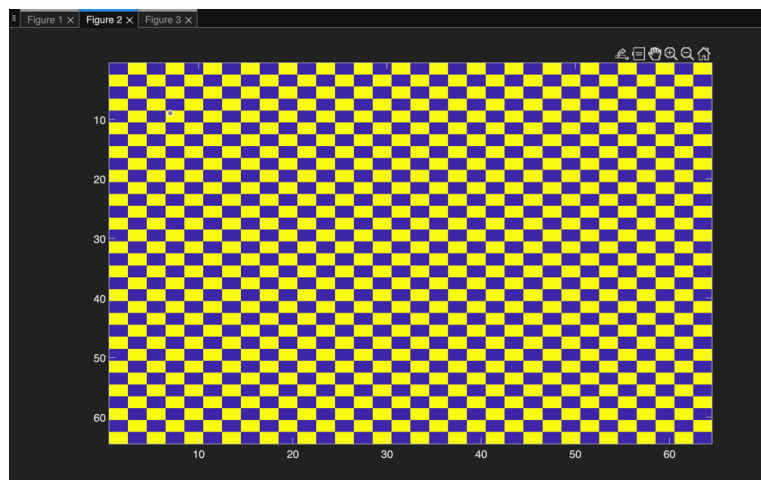


Figure 2

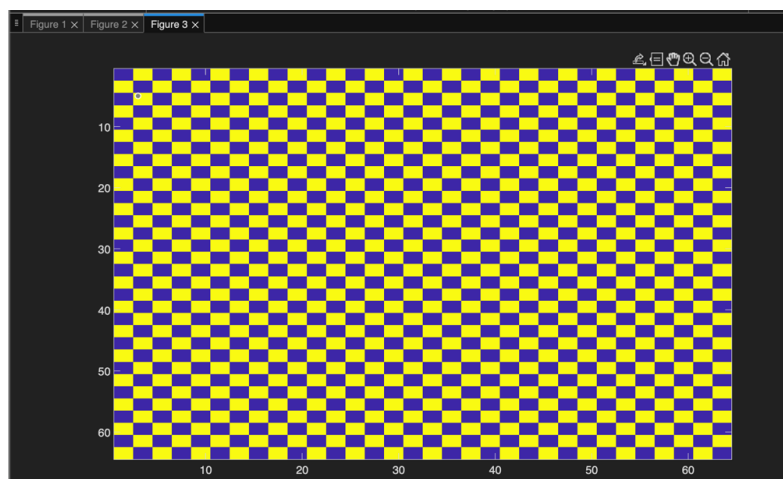


Figure 3

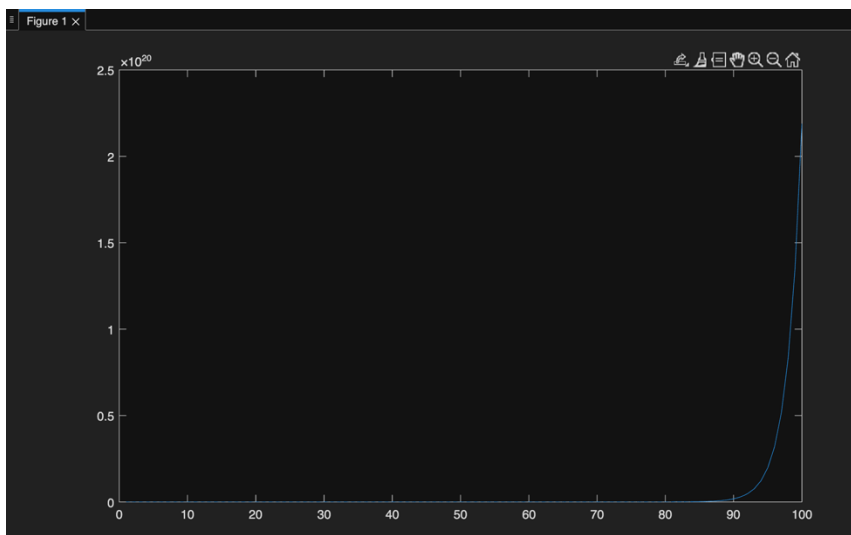
## Zad.2

```
clear variables
clc
close all
f(1)=0;
f(2)=1;
tic;
for i = 3:100
    f(i)=f(i-2)+f(i-1);
end
k=toc;
disp(['Czas bez deklaracji: ',num2str(k)])
plot(f);
```

```
clear variables
clc
close all
f=zeros(1,100);
f(2)=1;
tic;
for i = 3:100
    f(i)=f(i-2)+f(i-1);
end
l=toc;
disp(['Czas z deklaracją ',num2str(l)])
plot(f);
```

### Komentarz:

Powyżej widoczne są dwie wersje skryptu: z deklaracją i bez deklaracji rozmiaru tablicy. Czasy mierzyłem kilkakrotnie (wyniki widoczne na drugim zrzucie ekranu), obie opcje dają podobne rezultaty, ale opcja z deklaracją po kilku kolejnych pomiarach dawała nieznacznie mniejsze wyniki. Na pierwszym zrzucie ekranu widać wykres, a na drugim czasy.



```
Command Window
Czas bez deklaracji: 0.000967
>> zad2
Czas bez deklaracji: 0.0010425
>> zad2
Czas bez deklaracji: 0.001124
>> zad2
Czas bez deklaracji: 0.0010042
>> zad2d
Czas z deklaracją 0.00104
>> zad2d
Czas z deklaracją 0.00098758
>> zad2d
Czas z deklaracją 3.2833e-05
>> zad2d
Czas z deklaracją 1.8292e-05
>> Press [P] to generate code with Copilot
```

## Zad.3

---

```
clear variables
clc
close all
function F = Fibonacci(N)
    if N < 1
        F = [];
        return;
    end

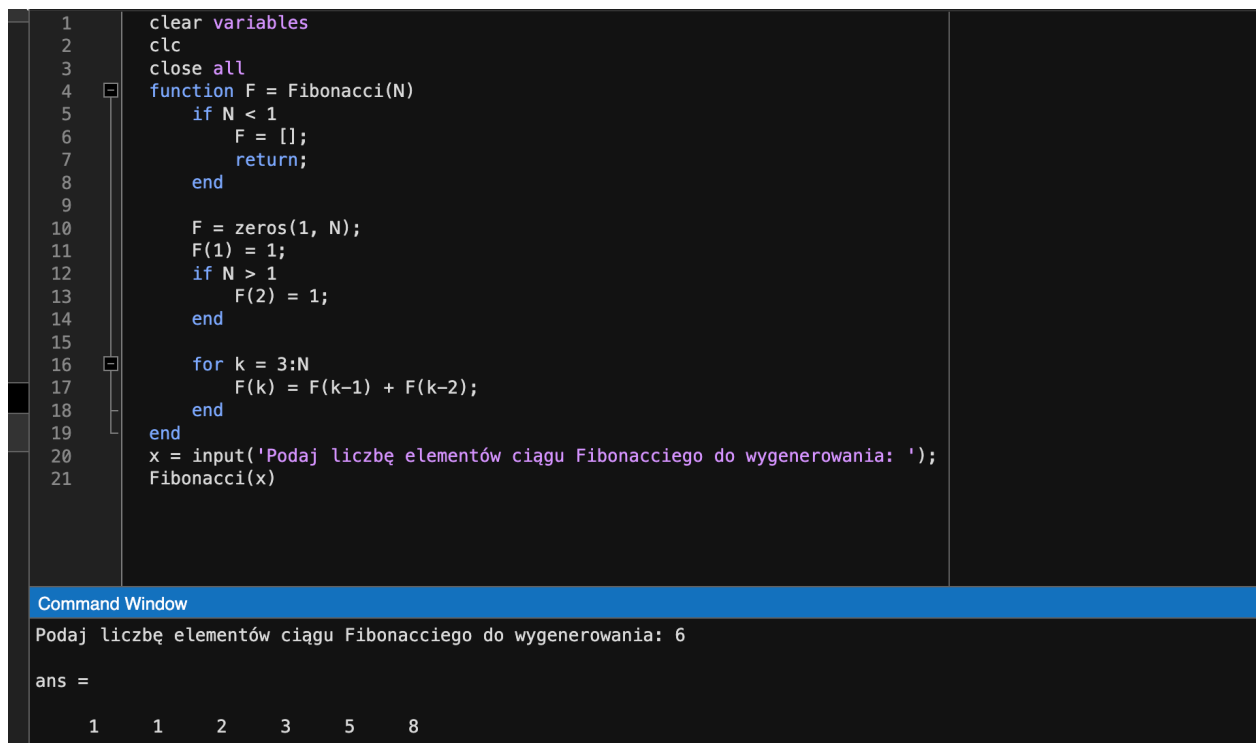
    F = zeros(1, N);
    F(1) = 1;
    if N > 1
        F(2) = 1;
    end

    for k = 3:N
        F(k) = F(k-1) + F(k-2);
    end
end
x = input('Podaj liczbę elementów ciągu Fibonacciego do wygenerowania: ');
Fibonacci(x)
```

---

Komentarz:

Skrypt tworzy funkcję MyFibonnacci, następnie prosi użytkownika o podanie ile elementów chce wyświetlić i wyświetla wynik dla pożądanej liczby elementów.



```
1 clear variables
2 clc
3 close all
4 function F = Fibonacci(N)
5     if N < 1
6         F = [];
7         return;
8     end
9
10    F = zeros(1, N);
11    F(1) = 1;
12    if N > 1
13        F(2) = 1;
14    end
15
16    for k = 3:N
17        F(k) = F(k-1) + F(k-2);
18    end
19 end
20 x = input('Podaj liczbę elementów ciągu Fibonacciego do wygenerowania: ');
21 Fibonacci(x)
```

Command Window

Podaj liczbę elementów ciągu Fibonacciego do wygenerowania: 6

ans =

1    1    2    3    5    8

## Zad.4

---

```
clear variables
clc
close all

x = -5:0.5:15;
y = -5:0.5:15;
[X, Y] = meshgrid(x, y);

Z = (X - 5).^2 + (Y - 3).^2;
figure;
mesh(X, Y, Z);

title('Wykres funkcji  $f(x, y) = (x - 5)^2 + (y - 3)^2$ ');
xlabel('Oś X');
ylabel('Oś Y');
zlabel('f(x, y)');
grid on;

%%

clear variables
clc
close all

t = 0:0.05:100;

A = 1; B = 1;
a = 3; b = 4;
delta = pi/2;

x = A * sin(a * t + delta);
y = B * sin(b * t);
z = t;

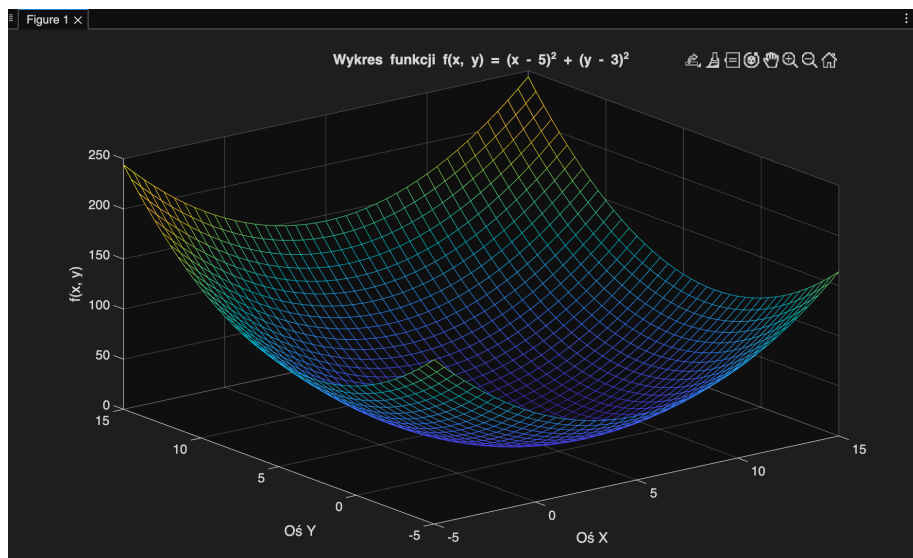
figure;
plot(x, y);
title('Krzywa Lissajous 2D');
grid on;

figure;
plot3(x, y, z);
title('Krzywa Lissajous w 3D');
xlabel('x'); ylabel('y'); zlabel('czas');
grid on;
```

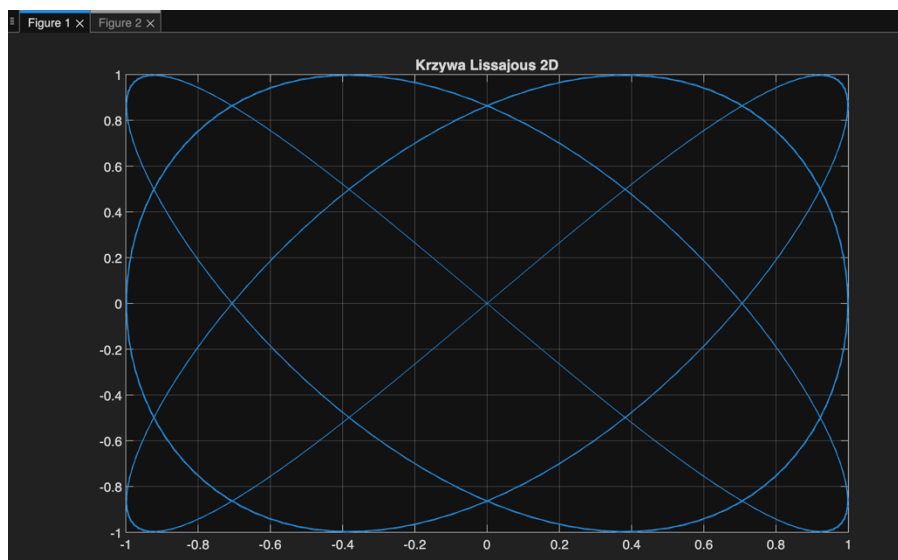
---

Komentarz:

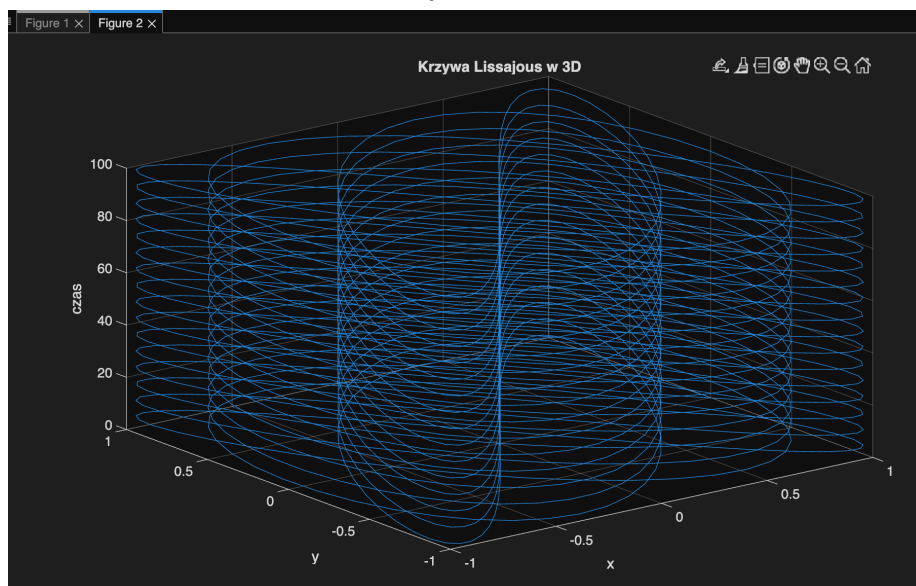
W pierwszej sekcji skrypt wyświetla wykres funkcji zadanej w zadaniu za pomocą mesh. W drugiej sekcji skrypt w Figure 1 wyświetli krzywą Lissajous w 2D, a w Figure 2 w 3D za pomocą odpowiednio plot i plot3.



Sekcja 1



Sekcja 2 : 2D



Sekcja 2 : 3D