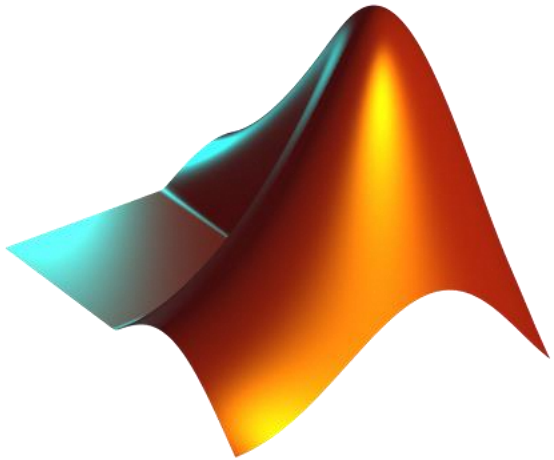
Programowanie w zastosowaniach inżynierskich



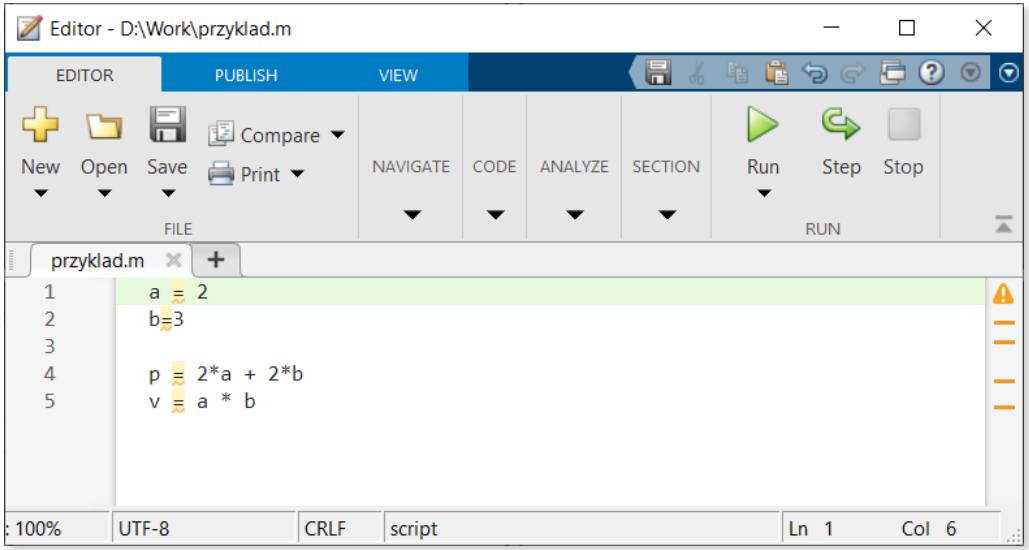
Skrypty, funkcje, instrukcje sterujące

Skrypty i funkcje

Live scripts

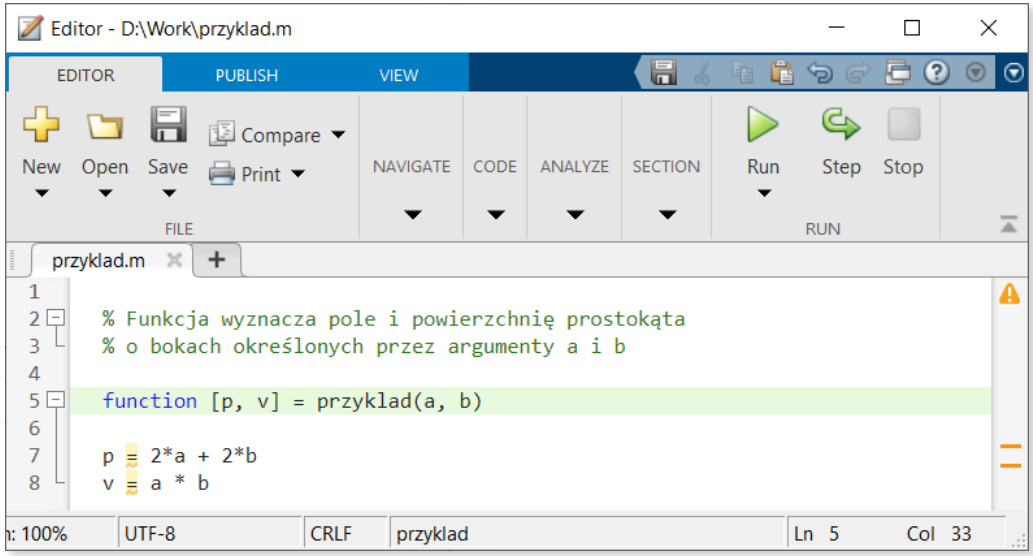
Instrukcja warunkowa

Skrypt (m-plik) – ciąg poleceń Matlba zapisany w pliku tekstowym `*.m`, uruchomienie skryptu powoduje wykonanie poleceń zapisanych w kolejnych wierszach.



- nazwa pliku nie powinna zawierać polskich znaków,
- uruchomienie skryptu następuje po wprowadzeniu nazwy pliku w Command Window (bez rozszerzenia) lub po wybraniu opcji Run w edytorze,
- skrypt może być uruchomiony z innego skryptu przez wprowadzenia jego nazwy,
- wszystkie zmienne utworzone podczas działania skryptu pozostają w obszarze roboczym Matlaba (również po zakończeniu skryptu),
- tekst rozpoczynający się znakiem % jest komentarzem i będzie pomijany podczas wykonania.

Funkcja – rodzaj skryptu Matlaba, który pozwala na zdefiniowanie argumentów wejściowych i wyjściowych.



- nazwa funkcji powinna być zgodna z nazwą pliku, w którym jest zapisana,
- funkcja ma własny, lokalny obszar roboczy, zmienne utworzone podczas jej działania nie są dostępne w globalnym obszarze roboczym Matlaba (są usuwane po zakończeniu funkcji),
- opcjonalny komentarz umieszczony przed deklaracją funkcji będzie wyświetlany w przypadku użycia komendy `help`.

Tworzenie i wywołanie funkcji

Deklaracja funkcji

```
function [out1,out2,…,outN] = nazwa(in1,in2,…,inM)
```

- `nazwa` – nazwa funkcji, dozwolone litery, cyfry, znak podkreślenia, zaczyna się od litery, zgodna z nazwą pliku, w którym funkcja jest zapisana,
- `out1,…,outN` – argumenty wyjściowe (opcjonalne, zasady nazewnictwa jak zmienne),
- `in1,…,inM` – argumenty wejściowe (opcjonalne, zasady nazewnictwa jak zmienne),
- w jednym pliku można umieścić deklarację kilku funkcji, pierwsza jest funkcją główną, pozostałe są lokalne, dostępne tylko w pliku, w którym zostały zadeklarowane,
- deklaracja funkcji kończy się słowem **`end`** (opcjonalne gdy plik zawiera jedną funkcje),
- nagłówek funkcji głównej powinien być pierwszym wykonywalnym wierszem skryptu.

Wywołanie funkcji

```
[out1,out2,…,outN] = nazwa(in1,in2,…,inN)
```

- liczba argumentów wejściowych i wyjściowych nie może być większa od określonej w deklaracji (nagłówku) funkcji,
- w przypadku mniejszej liczby parametrów wyjściowych pozostałe wartości zwracane przez funkcję zostaną zignorowane.

Funkcje – przykłady

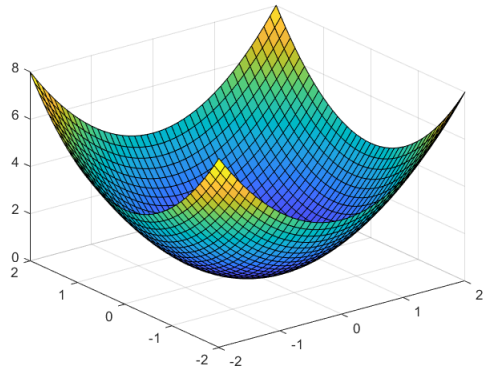
```
function [v] = pole(a, b)
v = a*b;
end

>> pole(2,3)
ans =
      6

>> x = pole(2,3)
x =
     6

>> x = pole(4)
Not enough input arguments.
```

```
function [v,p] = pole_obwod(a, b)
v = a*b;
p = 2*a + 2*b
end

>> pole_obwod(2,3)
ans =
       6

>> x = pole_obwod(2,3)
x =
      6

>> [x,y] = pole_obwod(2,3)
x =
     6
y =
     10
```

Funkcje – przykłady

Zadanie: należy wyznaczyć minimum funkcji

$$z = x^2 + y^2$$



Używana funkcja Matlaba

```
[x,fval] = fminsearch(f, x0)
```

- `x` to znalezione minimum,
- `fval` to wartość funkcji w minimum
- `f` to funkcja, której minimum jest poszukiwane (zadeklarowana jako funkcja Matlaba),
- `x0` to punkt początkowy (punkt od którego rozpoczyna się poszukiwanie minimum).

Uwaga: `fminsearch` operuje na funkcjach jednoargumentowych, jeżeli minimalizowana funkcja jest funkcją wielu zmiennych to należy przekazywać je w postaci wektora.

```
function y = fun(x)
y = x(1)^2 + x(2)^2;
End
```

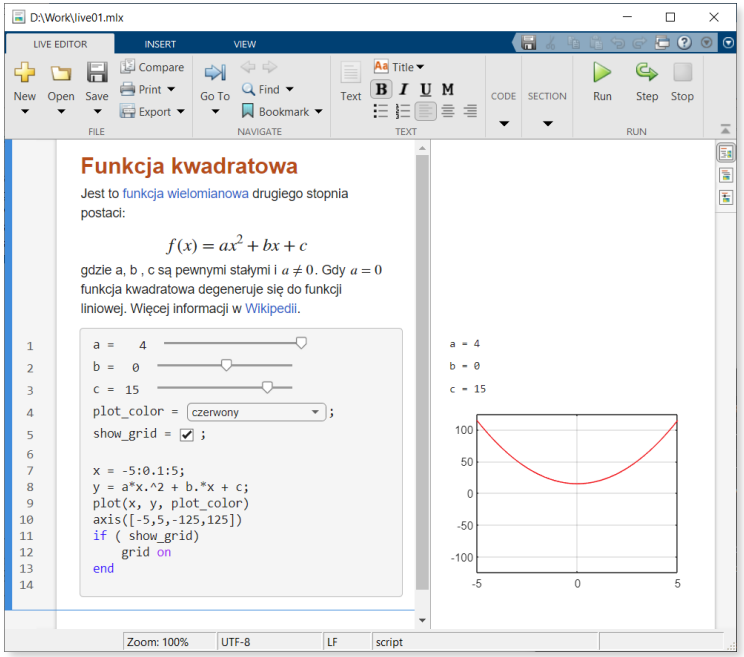
```
>> [x,v] = fminsearch(@fun, [1,1])
x =
    1.0e-04 *
   -0.2102   0.2548
v =
    1.0915e-09
```

Uwaga:
1.0e-04=1*10^(-4)=0.0001
1.0e+04=1*10^4=10000

Live scripts

Live script (aktywny skrypt) to interaktywny dokument MATLAB-a (rozszerzenie mlx), który łączy kod programu ze sformatowanym tekstem, grafiką oraz elementami multimedialnymi w środowisku Live Editor. Cechy aktywnych skryptów:

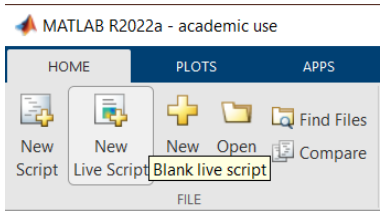
- Umożliwiają wprowadzanie rozbudowanych komentarzy (tekst, grafika, multimedia) opisujących problem i kod będący jego rozwiązaniem;
- Kod może być uzupełniony o interaktywne elementy sterujące, które umożliwiają wybór wartości zmiennych, ustalanie parametrów, itp.
- Kod umieszczony w skrypcie jest uruchamiany w środowisku Live editor;
- Skrypt przechowuje i wyświetla wyniki razem z kodem, który je wytworzył;
- Skrypty mogą być eksportowane do formatów PFD, Word, HTML, LaTeX.



Wymagania

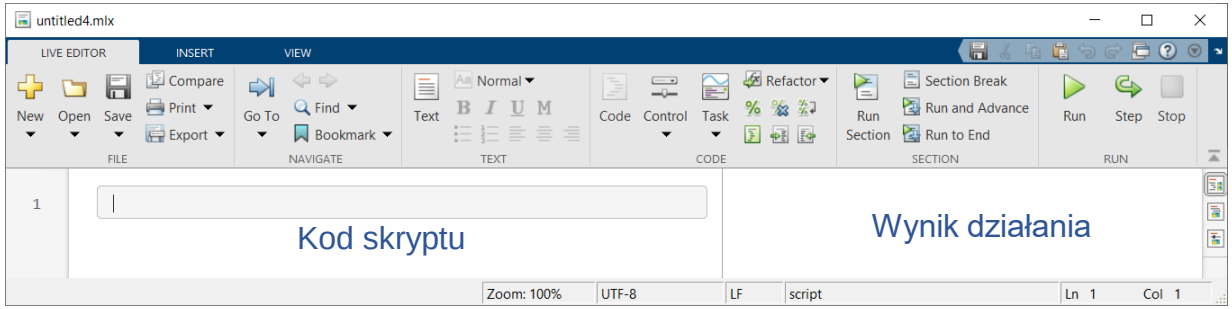
- MATLAB 2016a live scripts
- MATLAB 2018a live functions

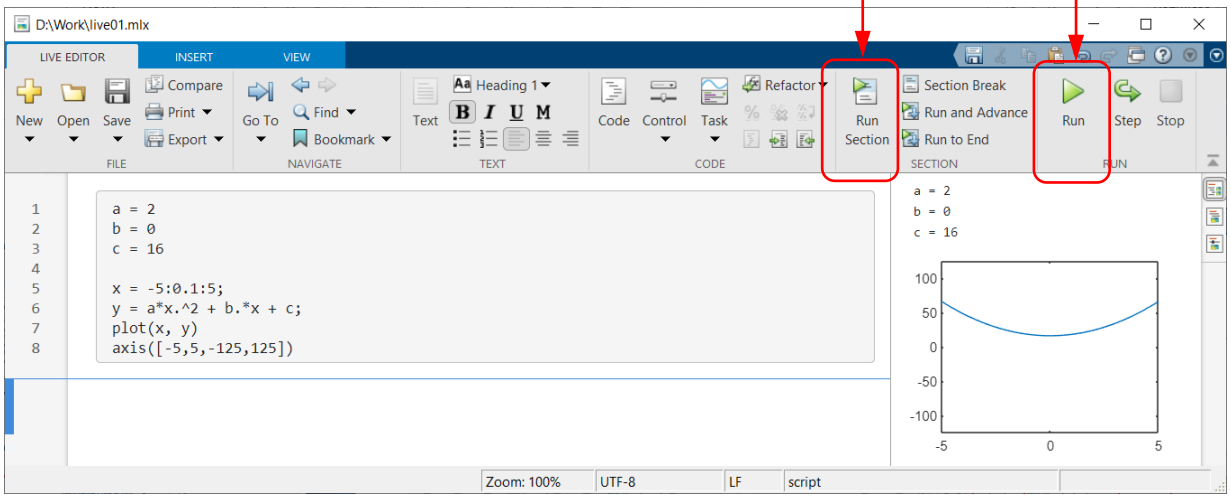
Tworzenie Live script



`Home > New Live Script`

Polecenie (Command window): **`edit script_name.mlx`**





Uruchomienie:
- pojedyncza sekcja
- całość

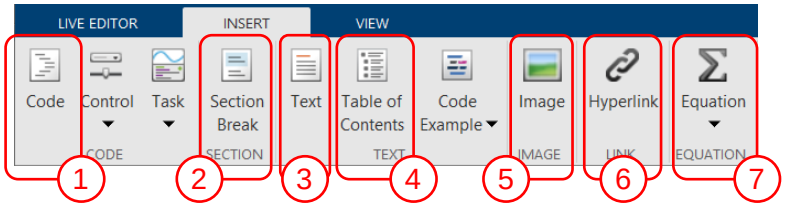
Kod skryptu

```
a = 2
b = 0
c = 16
x = -5:0.1:5;
y = a*x.^2+b.*x+c;
plot(x, y)
axis([-5,5,-125,125])
```

Uwaga: instrukcja `axis` ustawia zakresy osi wykresu.

Live script – komentarze

Wstawianie sekcji i komentarzy

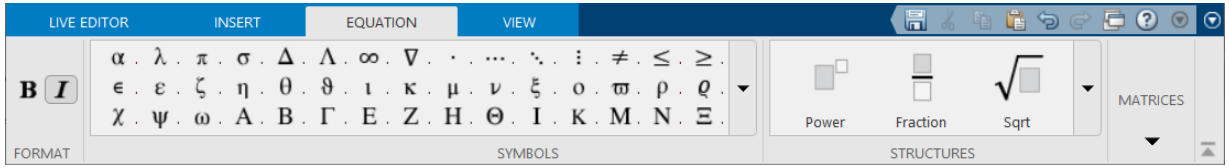


1 – sekcja kodu 2 – koniec sekcji 3 – sekcja tekstowa

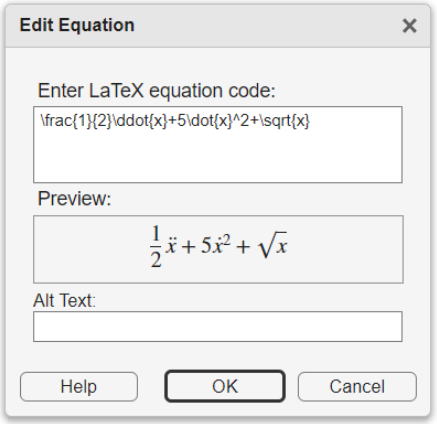
4 – spis treści (lista sekcji z odnośnikami)

5 – grafika 6 – odnośnik 7 – wzór

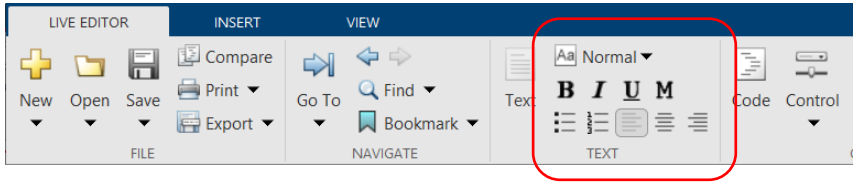
Edytor wzorów (7)

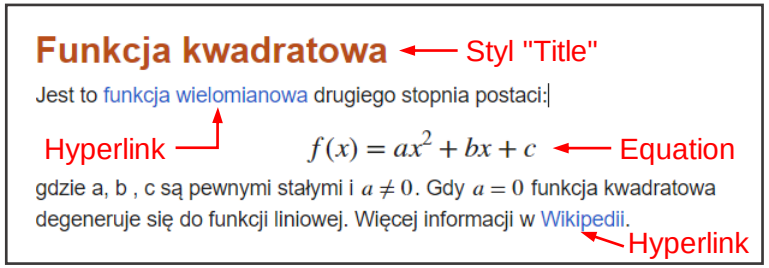


Uwaga: poza edytorem graficznym Live script editor umożliwia wprowadzanie wzorów zgodnie e standardem LaTeX (druga opcja w menu Equation).



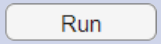
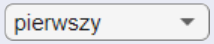
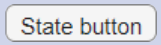
Formatowanie tekstu





Live script – elementy sterujące

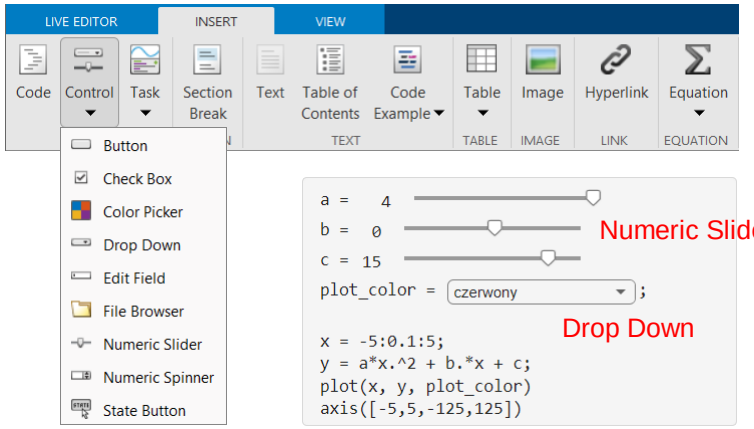
Interactive Controls to zestaw elementów sterujących umożliwiających interaktywną zmianę wartości zmiennych podczas działania skryptu. Zmiana wartości wykonana za pomocą takiego elementu powoduje uruchomienie kodu i wykonanie instrukcji zawartych w bieżącej sekcji lub całym skrypcie (zależnie od ustawień elementu).

Element	*Nazwa*	*Zastosowanie*
	Button	Uruchamiane skryptu
	Check Box	Ustawianie wartości logicznej (True/False)
	Color Picker	Wybór koloru
	Drop-Down List	Wybór jednej z dozwolonych wartości (lista)
	Edit Field	Wprowadzanie wartości z klawiatury
	File Browser	Wybór pliku ze standardowego okna dialogowego
	Numeric Slider	Zmiana wartości numerycznej przy pomocy suwaka
	Numeric Spinner	Zmiana wartości numerycznej w polu edycyjnym
	State Button	Ustawianie wartości logicznej (alternatywa dla Check Box)

Uwaga: zestawienie elementów sterujących dla wersji 2023b

Live script – elementy sterujące

Dodawanie elementów sterujących



W celu dodania elementu ustawiającego wartość zmiennej należy zaznaczyć wartość przypisaną do zmiennej i wybrać odpowiedni element. Powiązanie ze zmienną wykonywane jest automatycznie.

Uwaga: w menu Control aktywne są elementy zgodne z typem zmiennej.

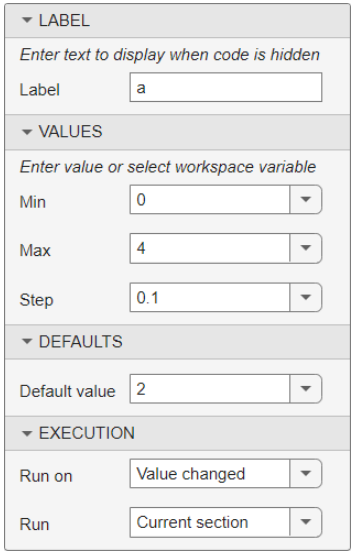
Konfiguracja elementów sterujących

`Menu podręczne > Configure control` lub podwójne kliknięcie

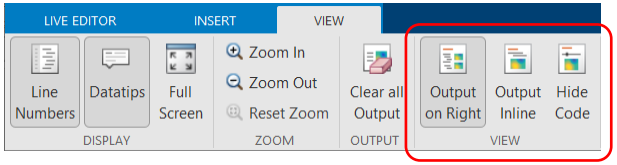
Sekcje okna konfiguracyjnego:

- Label – etykieta wyświetlana gdy kod jest ukryty;
- Values – ustawienia dotyczące wartości (specyficzne dla elementu);
- Defaults – wartość domyślna;
- Execution – sposób działania:
  - Run on – reakcja na zmianę wartości: Value changing lub Value changed ciągła zmiana lub po przestawieniu elementu,
  - Run – zasięg działania.

Konfiguracja *Numeric slider*



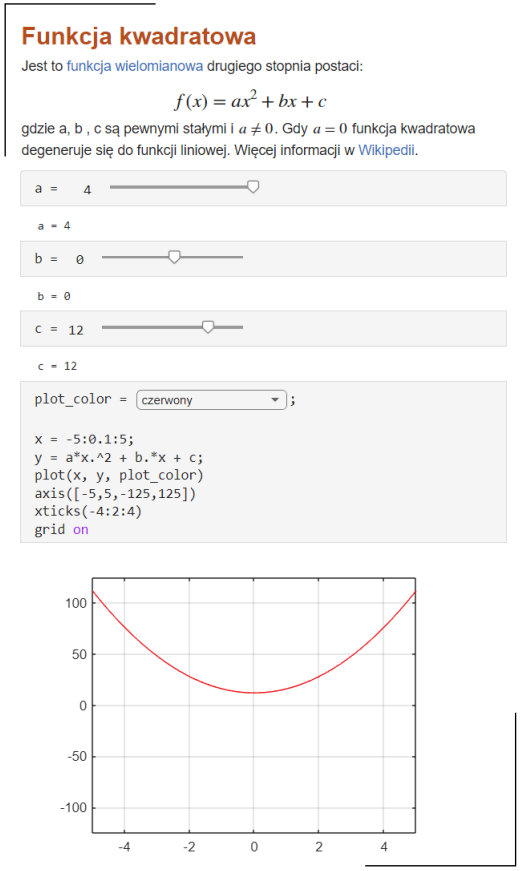
Live script – widoki

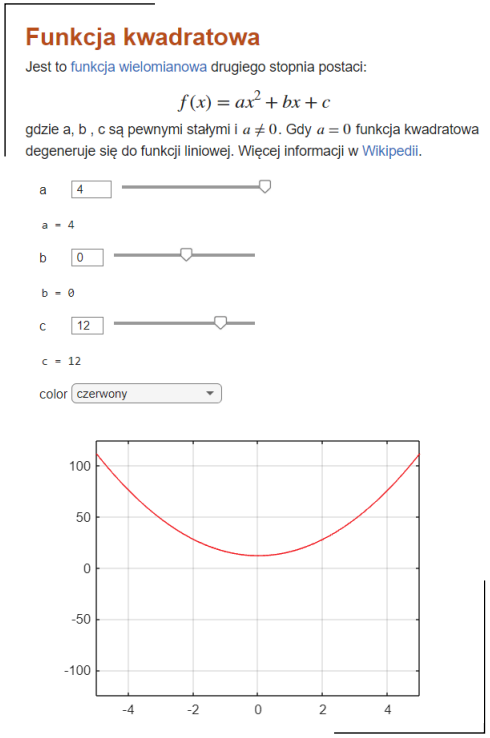


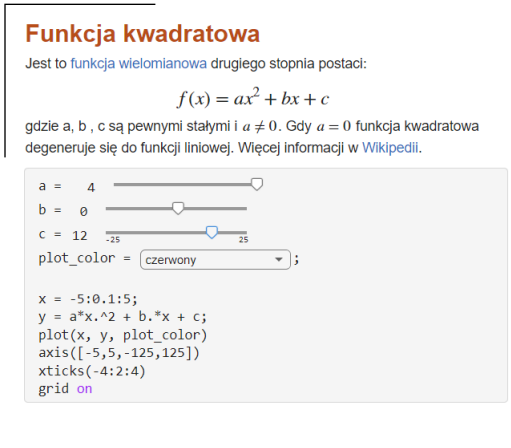
Output on Right wyniki wyświetlane na prawo obok kodu, który je generuje.

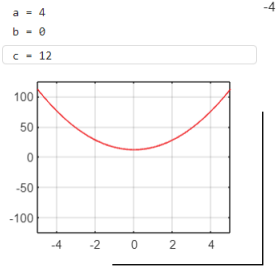
Output Inline każdy wynik wyświetlany bezpośrednio pod kodem.

Hide Code kod ukryty.









Więcej informacji : mathworks.com

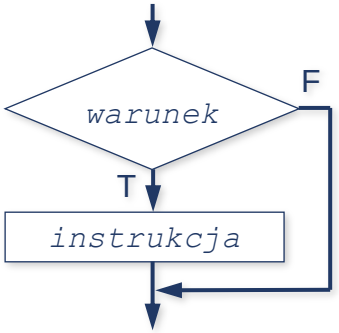
Instrukcja warunkowa

Instrukcja sterująca – element języka programowania, który służy do określenia kolejności wykonania instrukcji zawartych w kodzie programu.

Instrukcja warunkowa – wprowadza rozgałęzienie w kodzie programu, tworząc alternatywne sekwencje instrukcji.

Wariant I

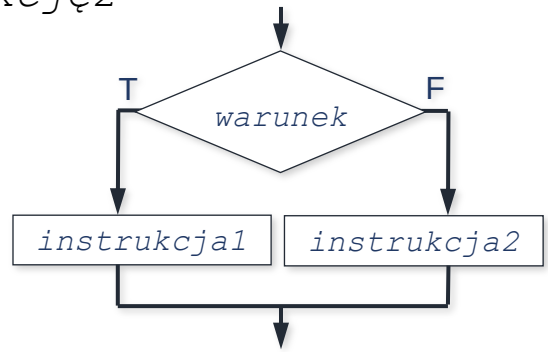
Jeżeli *warunek* jest prawdziwy wykonaj *instrukcję* (grupę instrukcji).




```
if warunek
  instrukcja
end
```

Wariant II

Jeżeli *warunek* jest prawdziwy wykonaj *instrukcję1* w przeciwnym wypadku *instrukcję2*




```
if warunek
  instrukcja1
else
  instrukcja2
end
```

Instrukcja warunkowa – przykład

```
a =   4
b =   0
c =  12
plot_color = czerwony ;
show_grid = ☐ ;

x = -5:0.1:5;
y = a*x.^2 + b.*x + c;
plot(x, y, plot_color)
axis([-5,5,-125,125])
xticks(-4:2:4)
if ( show_grid )
    grid on
end
```

show_grid – zmienna logiczna

Check Box – ustawienie wartości zmiennej show_grid

grid on – polecenie ustawiające widoczność siatki (domyślnie niewidoczna)

xticks – gęstość podziałki w osi x

```
a =   4
b =   0
c =  12
plot_color = czerwony ;
show_grid = ☑ ;

x = -5:0.1:5;
y = a*x.^2 + b.*x + c;
plot(x, y, plot_color)
axis([-5,5,-125,125])
xticks(-4:2:4)
if ( show_grid )
    grid on
end
```


```
if ( show_grid )
  grid on
end
```

Uwaga: Jeżeli zmienna **show_grid** jest ustawiona na `True` (warunek prawdziwy) wykonywana jest instrukcja **grid on**, która ustawia widoczność siatki. Gdy **show_grid** jest ustawiona na `False` (warunek fałszywy) instrukcja **grid on** jest pomijana i siatka pozostaje niewidoczna.