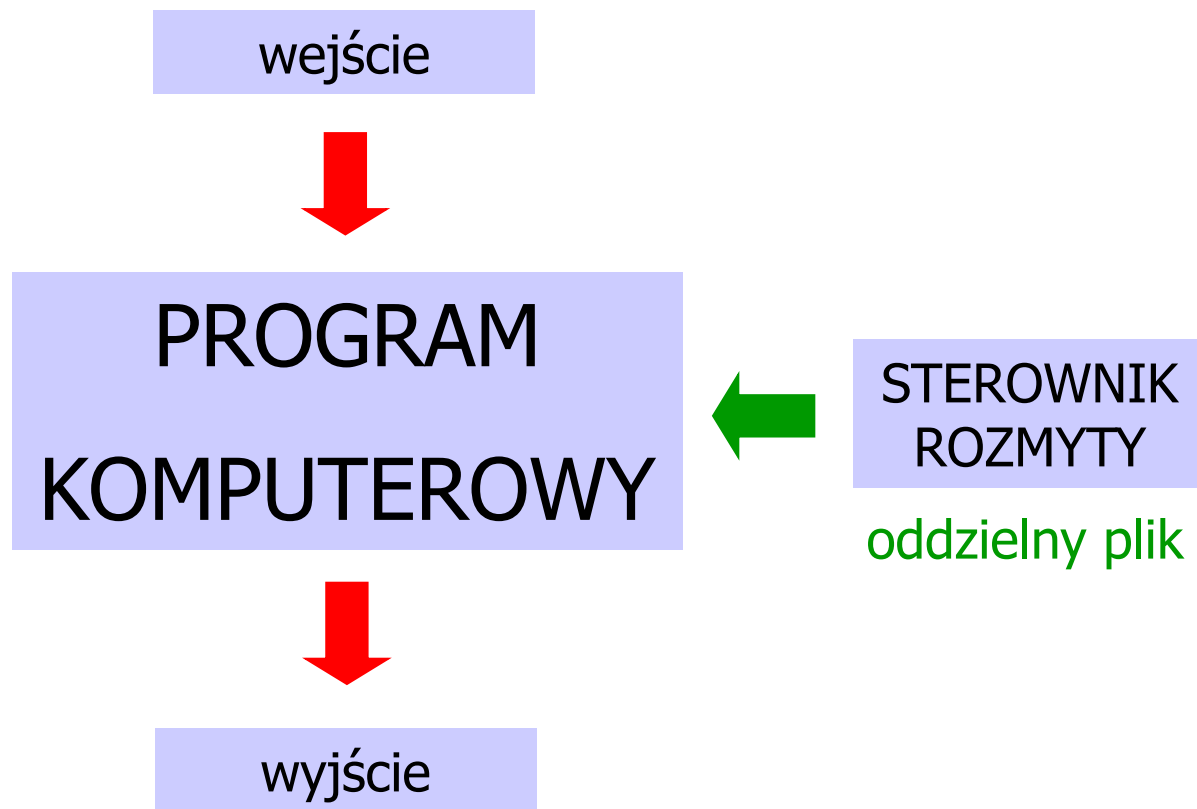


# Sterowniki rozmyte i programowanie



# Fuzzy Control Language

**Fuzzy Control Language** (FCL) to język pozwalający budować (definiować) sterowniki rozmyte.

Definicja sterownika rozmytego zapisana jest w pliku tekstowym z rozszerzeniem **fcl**.

Plik **fcl** zawiera instrukcje określające **parametry sterownika**. Instrukcje te zawarte są w następującym elemencie:

```
FUNCTION_BLOCK
```

```
//instrukcje
```

```
END_FUNCTION_BLOCK
```

# Fuzzy Control Language

Chcemy zbudować przykładowy sterownik rozmyty, który dla otrzymanej na wejściu odległości od przeszkody (odleglosc) wyznaczy nam prędkość (predkosc) pojazdu.

Wykorzystamy dwie zmienne lingwistyczne:

odleglosc (wejście sterownika)

predkosc (wyjście sterownika)

W języku FCL zapisujemy to następująco:

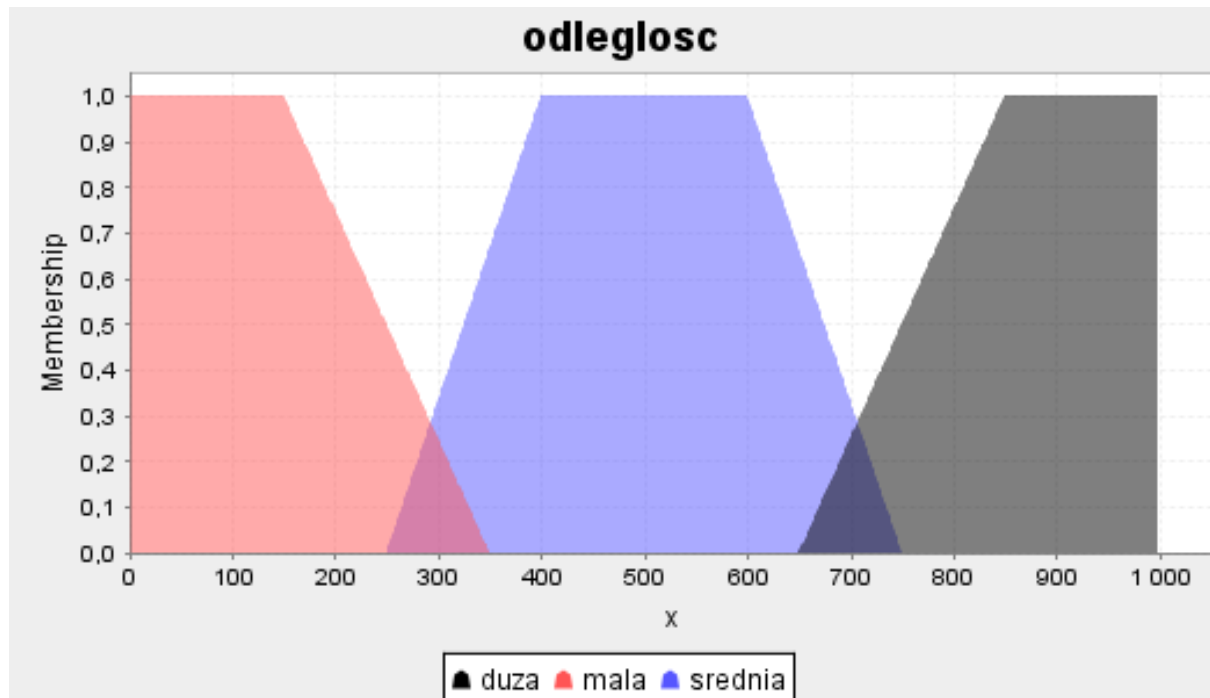
```
VAR_INPUT
odleglosc : REAL;
END_VAR
VAR_OUTPUT
predkosc : REAL;
END_VAR
```

# FCL - wejście

Przyjmijmy, że interesuje nas **odleglosc** w przedziale  $[0,1000]$  (m).

Konkretna wartość zmiennej **odleglosc** będzie podana **na wejściu** naszego sterownika. Wartość ta będzie następnie **rozmyta**.

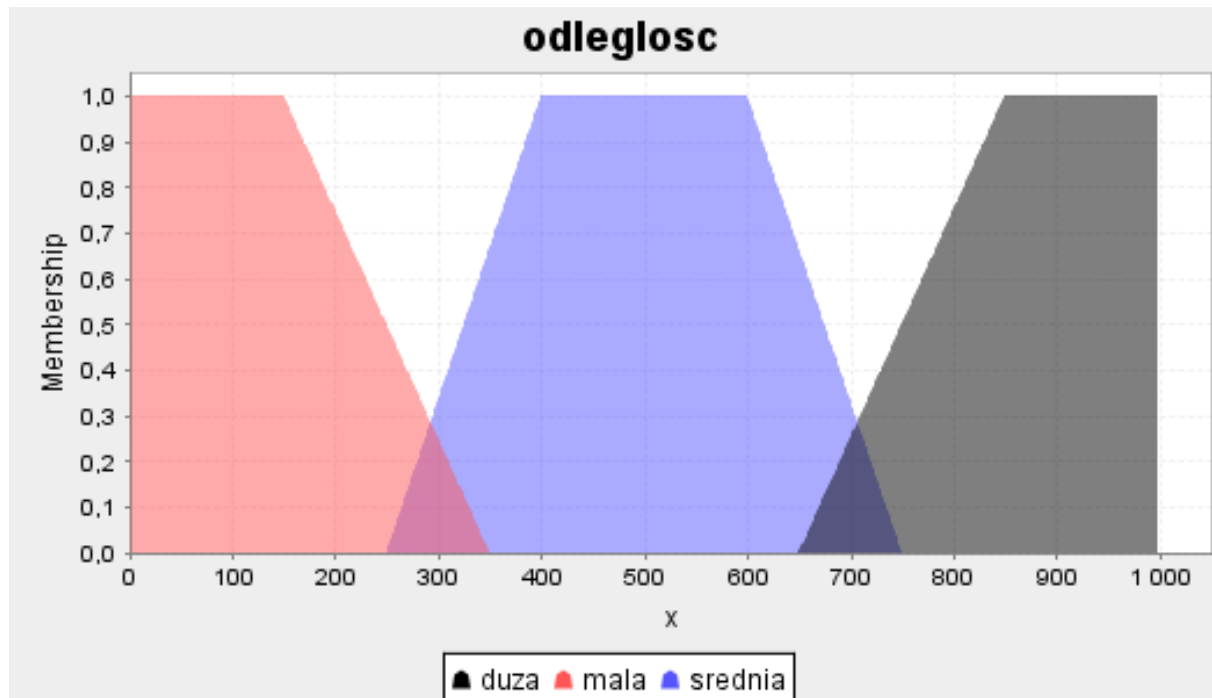
Przyjmijmy, że zmienna **odleglosc** będzie przyjmowała następujące 3 wartości:



# FCL - wejście

## **FUZZIFY odleglosc**

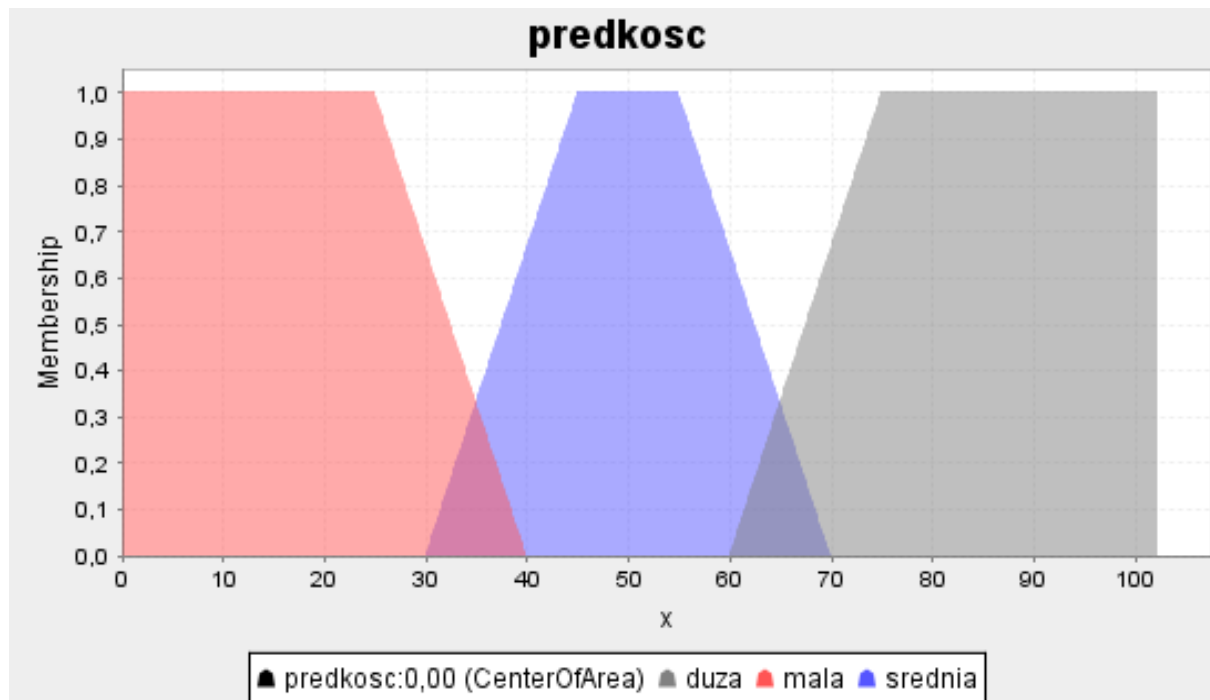
```
TERM mala := (0, 1) (150, 1) (350, 0) ;  
TERM srednia := (250, 0) (400,1) (600,1) (750,0) ;  
TERM duza := (650, 0) (850, 1) (1000, 1) ;  
END_FUZZIFY
```



# FCL - wyjście

Przyjmijmy, że interesuje nas **predkosc** w przedziale **[0,100]** (km/h). Konkretna wartość zmiennej **predkosc** będzie zwrócona **na wyjściu** naszego sterownika. Wartość ta będzie efektem wyostrzania.

Przyjmijmy, że zmienna **predkosc** będzie przyjmowała następujące 3 wartości:



# FCL - wyjście

**DEFUZZIFY predkosc**

TERM mala := (0, 1) (25, 1) (40, 0);

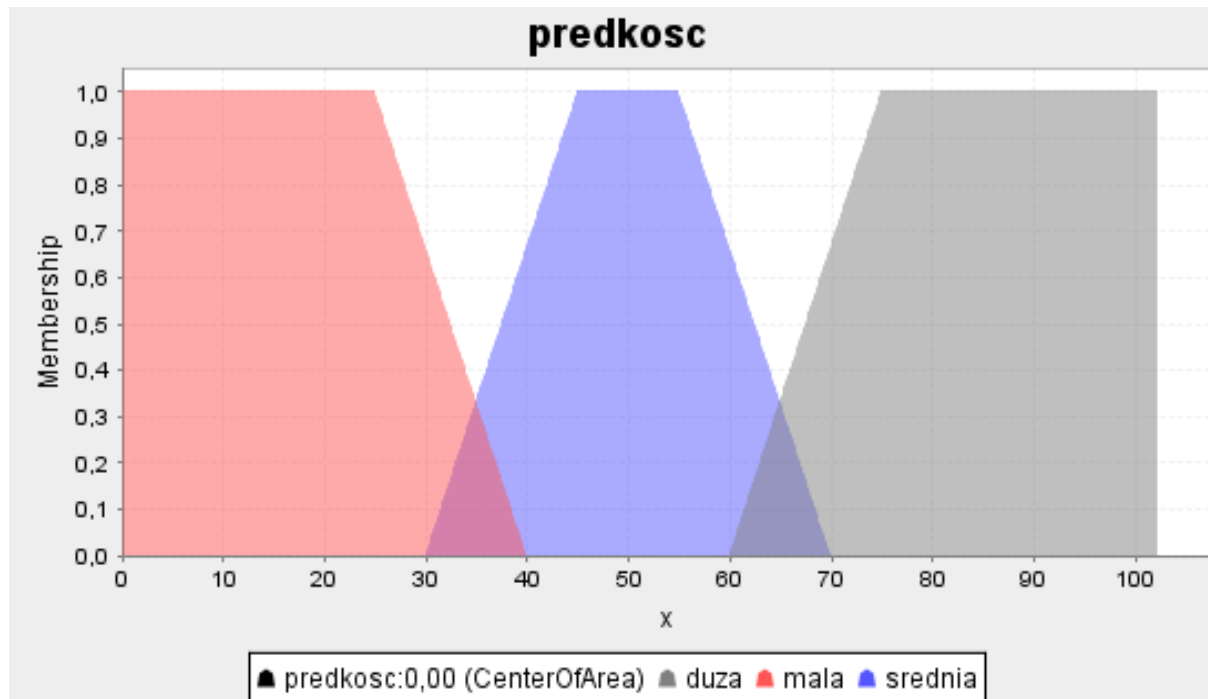
TERM srednia := (30, 0) (45, 1) (55, 1) (70, 0);

TERM duza := (60, 0) (75, 1) (100, 1);

**METHOD : COA;**

**END\_DEFUZZIFY**

**METODA WYOSTRZANIA**



# FCL - wyostrzanie

Metody wyostrzania:

**COG** - Centre of Gravity

**COGS** - Centre of Gravity for Singletons

**COA** - Centre of Area

**LM** - Left Most Maximum

**RM** - Right Most Maximum

Możemy teraz przystąpić do zdefiniowania **bazy reguł**.



# FCL – baza reguł

Przyjmijmy następującą bazę reguł:

JEŻELI **odleglosc** jest **mala** TO **predkosc** jest **mala**

JEŻELI **odleglosc** jest **srednia** TO **predkosc** jest **srednia**

JEŻELI **odleglosc** jest **duza** TO **predkosc** jest **duza**

W języku FCL zapisujemy to następująco:

```
RULE 1 : IF odleglosc IS mala THEN predkosc IS mala;  
RULE 2 : IF odleglosc IS srednia THEN predkosc IS srednia;  
RULE 3 : IF odleglosc IS duza THEN predkosc IS duza;
```

# FCL – AND i OR

Ponadto musimy określić jeszcze:

- Metodę **AND** i **OR** (do wykorzystania po lewej stronie implikacji)

Mamy do wyboru:

operator OR		operator AND	
keyword for Algorithm	Algorithm	keyword for Algorithm	Algorithm
MAX	$\text{Max } (\mu_1(x), \mu_2(x))$	MIN	$\text{Min}(\mu_1(x), \mu_2(x))$
ASUM	$\mu_1(x) + \mu_2(x) - \mu_1(x) \mu_2(x)$	PROD	$\mu_1(x) \mu_2(x)$
BSUM	$\text{Min}(1, \mu_1(x) + \mu_2(x))$	BDIF	$\text{Max}(0, \mu_1(x) + \mu_2(x) - 1)$

W języku FCL zapisujemy to następująco:

**AND** : MIN;

Wystarczy, że określimy jeden operator!

# FCL - aktywacja

Ponadto musimy określić jeszcze:

- Metodę **aktywacji** (implikacja!)

Mamy do wyboru:

Name	Keyword	Algorithm
<b>Product</b>	<b>PROD</b>	$\mu_1(x) \mu_2(x)$
<b>Minimum</b>	<b>MIN</b>	$\text{Min}(\mu_1(x), \mu_2(x))$

W języku FCL zapisujemy to następująco:

```
ACT : MIN;
```

# FCL - akumulacja

Ponadto musimy określić jeszcze:

- Metodę **akumulacji** (suma zbiorów!)

Mamy do wyboru:

Name	Keyword	Formula
Maximum	MAX	$\text{Max}(\mu_1(X), \mu_2(X))$
Bounded Sum	BSUM	$\text{Min}(1, \mu_1(X) + \mu_2(X))$
Normalised Sum	NSUM	$\frac{\mu_1(X) + \mu_2(X)}{\text{Max}(1, \text{MAX}(\mu_1(X') + \mu_2(X')) )}$

W języku FCL zapisujemy to następująco:

**ACCU** : MAX;

# FCL – blok reguł

Ostatecznie **blok reguł** w FCL wygląda następująco:

```
RULEBLOCK No1
```

```
AND : MIN;
```

```
ACT : MIN;
```

```
ACCU : MAX;
```

```
RULE 1: IF odleglosc IS mala THEN predkosc IS mala;
```

```
RULE 2: IF odleglosc IS srednia THEN predkosc IS srednia;
```

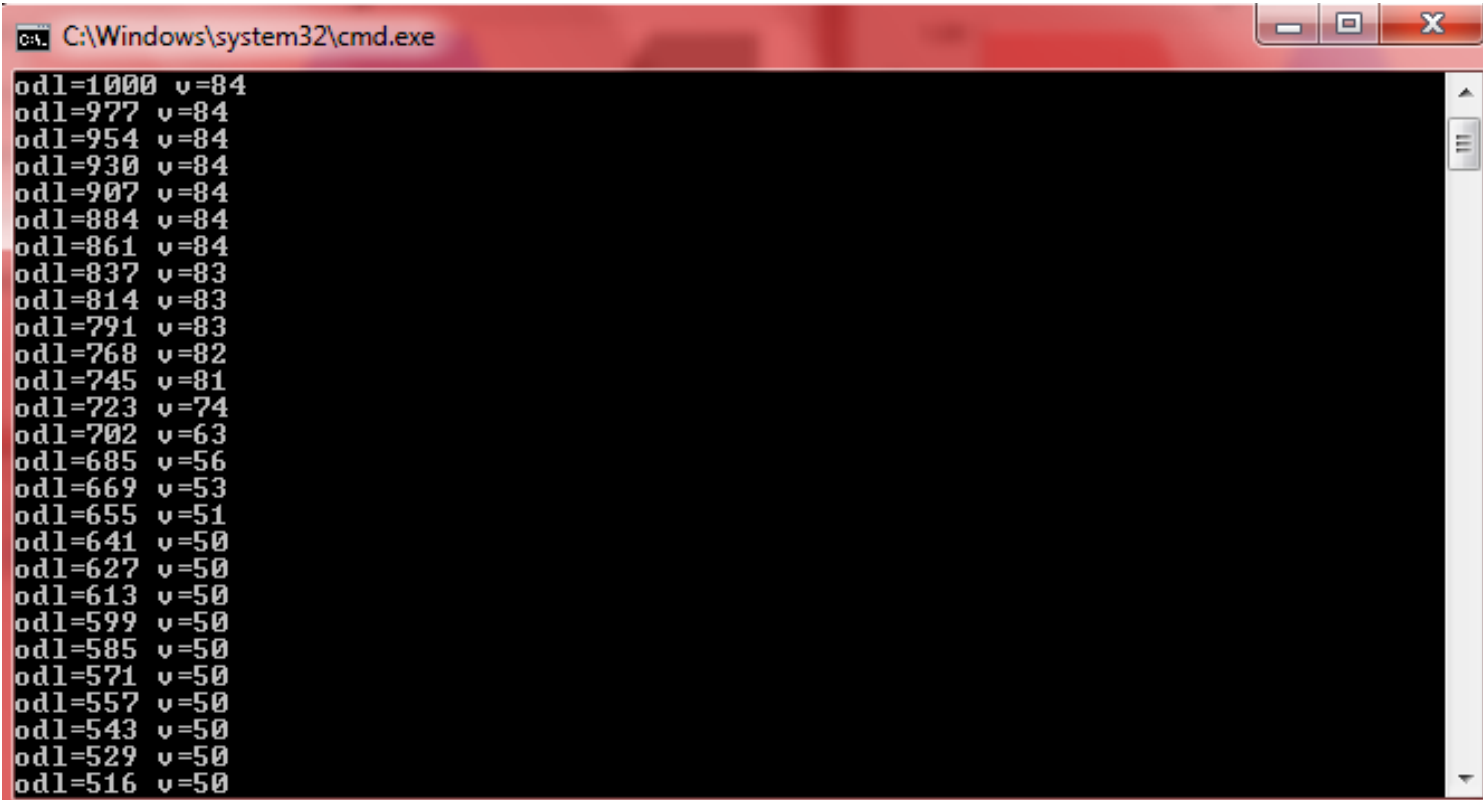
```
RULE 3 : IF odleglosc IS duza THEN predkosc IS duza;
```

```
END_RULEBLOCK
```

Jak zobaczymy bloków takich może być kilka!

## FCL – przykładowe wyjście

W naszym przykładowym programie otrzymujemy na wyjściu kolejno:



```
C:\Windows\system32\cmd.exe
odl=1000 v=84
odl=977 v=84
odl=954 v=84
odl=930 v=84
odl=907 v=84
odl=884 v=84
odl=861 v=84
odl=837 v=83
odl=814 v=83
odl=791 v=83
odl=768 v=82
odl=745 v=81
odl=723 v=74
odl=702 v=63
odl=685 v=56
odl=669 v=53
odl=655 v=51
odl=641 v=50
odl=627 v=50
odl=613 v=50
odl=599 v=50
odl=585 v=50
odl=571 v=50
odl=557 v=50
odl=543 v=50
odl=529 v=50
odl=516 v=50
```

## FCL – przykładowe wyjście

Ostatecznie po przejściu kilkudziesięciu iteracji:

