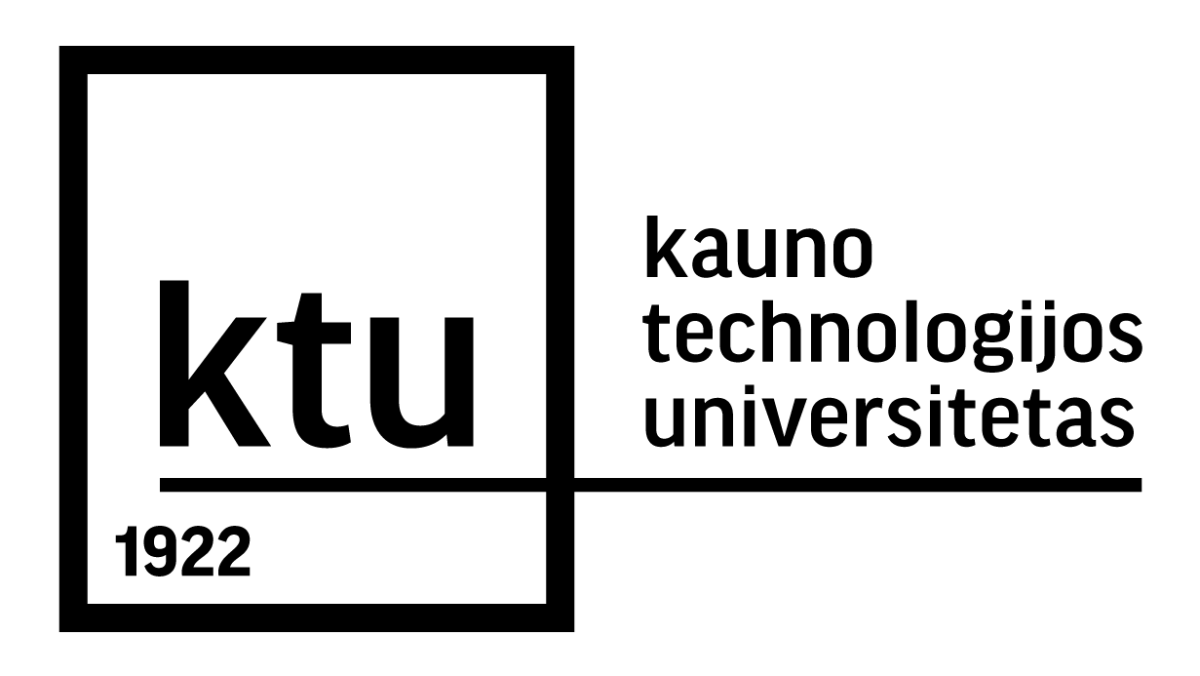
KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS



Intelektikos pagrindai (P176B101)

Laboratorinis darbas Nr.2

Atliko:

IFF-9/8 gr. studentas

Lukas Navašinskas

2022 m. balandžio 5 d.

Priėmė:

lekt. Nečiūnas Audrius

doc. Paulauskaitė-Tarasevičienė Agnė

KAUNAS 2022

Turinys

[1. Užduotis 3](#_Toc100012686)

[1.1. Reikalavimai 3](#_Toc100012687)

[1.2. Užduoties aprašas 3](#_Toc100012688)

[1.3. Įėjimo kintamieji 3](#_Toc100012689)

[1.4. Išėjimo kintamieji 3](#_Toc100012690)

[1.5. Fuzzy taisyklės 4](#_Toc100012691)

[2. Rezultatai 5](#_Toc100012692)

[2.1. Įėjimo reikšmių atvaizdavimas 5](#_Toc100012693)

[2.2. Taisyklių pritaikymo grafikai 6](#_Toc100012694)

[2.3. Atsakymai 8](#_Toc100012695)

[3. Programos kodas 8](#_Toc100012696)

# Užduotis

Sukurkite sprendimo priėmimo sistemą remiantis miglotosios logikos teorija (rekomenduojama taikant Mamdani algoritmą, tačiau gali būti naudojamas ir Sugeno modelis). Duomenys gali būti naudojami realūs, iš atvirų šaltinių arba sugalvoti jūsų pačių (dažniausiai studentai sugalvoja savo duomenis ir patiems aktualią problemą – t.y. jūs tampate ekspertais). Sistemos programinė realizacija turi būti atlikta naudojant Python (arba C šeimos kalbomis).

## Reikalavimai

* Aiškus užduoties aprašas, t.y., koks uždavinys, pagal kokius duomenis ką reikia paskaičiuoti. Aprašomi kintamųjų matmenys, jie sugalvoti ar paimti iš išorinių šaltinių ir pan.;
* Sistemos įvesčių kiekis ir fuzzy aibių skaičius: nuo 3 × 3 iki 4 × 4;
* Sistemos išvesčių kiekis ir fuzzy aibių skaičius: nuo 1 × 3 iki 2 × 3;
* Suformuotos ir pateiktos logiškos taisyklės naudojant du/tris skirtingus loginius kintamuosius (And, Or, Not).Visos taisyklės turi būti pateiktos ataskaitoje.
* Pateikti metodai panaudoti implikacijai, agregacijai ir defuzifikacijai. Defuzifikacijai reikia panaudoti du skirtingus atsakymo skaičiavimo metodus: Centroid ir MOM( arba LOM).
* Sudarius modelį reikia pateikti 3 testinių įvesčių reikšmių scenarijus ir gautus atsakymų rezultatus.

## Užduoties aprašas

Sukurkite fuzzy sistemos modelį, kuris apskaičiuotų mašinų avarijos tikimybę įvertinus oro temperatūrą, greitį, atstumą tarp automobilių.

## Įėjimo kintamieji

* Oro temperatūra (Co): [-10;30]; fuzzy aibės: šalta, šilta, karšta[;
* Automobilio greitis (km/h): [50;130]; fuzzy aibės: mažas, vidutinis, didelis;
* Atstumas tarp automobilių (m): [1;150]; fuzzy aibės: mažas, vidutinis, didelis;

## Išėjimo kintamieji

* Avarijos tikimybė (%): [0;100]; fuzzy aibės: žema, vidutinė, aukšta

## Fuzzy taisyklės

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Nr. | Oro temperatūra |  | Atstumas tarp automobilių |  | Automobilio greitis |  | Avarijos tikimybė |
| 1 | Šalta | or | Mažas | and | Didelis | then | Aukšta |
| 2 | Šalta | or | Vidutinis | and | Didelis | then | Aukšta |
| 3 | Šalta | or | Didelis | and | Didelis | then | Aukšta |
| 4 | Šilta | or | Mažas | and | Didelis | then | Aukšta |
| 5 | Karšta | or | Mažas | and | Didelis | then | Aukšta |
|  |  |  |  |  |  |  |  |
| 6 | Šilta | or | Vidutinis | and | Didelis | then | Vidutinė |
| 7 | Šilta | or | Didelis | and | Didelis | then | Vidutinė |
| 8 | Karšta | or | Vidutinis | and | Didelis | then | Vidutinė |
| 9 | Karšta | or | Didelis | and | Didelis | then | Vidutinė |
| 10 | Šalta | or | Mažas | and | Mažas | then | Vidutinė |
| 11 | Šalta | or | Vidutinis | and | Mažas | then | Vidutinė |
| 12 | Šilta | or | Mažas | and | Mažas | then | Vidutinė |
| 13 | Šilta | or | Vidutinis | and | Mažas | then | Vidutinė |
|  |  |  |  |  |  |  |  |
| 14 | Karšta | or | Mažas | and | Mažas | then | Maža |
| 15 | Karšta | or | Vidutinis | and | Mažas | then | Maža |
| 16 | Karšta | or | Vidutinis | and | Vidutinis | then | Maža |
| 17 | Karšta | or | Vidutinis | and | Mažas | then | Maža |
| 18 | Šalta | or | Didelis | and | Mažas | then | Maža |
| 19 | Šilta | or | Didelis | and | Mažas | then | Maža |
| 20 | Šilta | or | Didelis | and | Vidutinis | then | Maža |
| 21 | Karšta | or | Didelis | and | Vidutinis | then | Maža |

# Rezultatai

## Įėjimo reikšmių atvaizdavimas

Line chart

Description automatically generated

pav. 1 Įėjimo reikšmių grafikai

## Taisyklių pritaikymo grafikai

A picture containing chart

Description automatically generated

pav. 2 Pritaikytų taisyklių, kai oro temperatūra -10C, greitis 130km/h, atstumas tarp automobilių 5m

Chart

Description automatically generated with medium confidence

pav. 3 Agregacijos grafikas, kai oro temperatūra -10C, greitis 130km/h, atstumas tarp automobilių 5m

Avarijos tikimybė = 81.42857142857143 %, Svorio centras

Avarijos tikimybė = 90.0 %, Maksimumo vidurkis

Chart

Description automatically generated with medium confidence

pav. 4 Pritaikytų taisyklių grafikas, kai oro temperatūra 0C, greitis 130km/h, atstumas tarp automobilių 50m

Chart

Description automatically generated

pav. 5 Agregacijos grafikas, kai oro temperatūra 0C, greitis 130km/h, atstumas tarp automobilių 50m

Avarijos tikimybė = 72.22222222222223 %, Svorio centras

Avarijos tikimybė = 90.0 %, Maksimumo vidurkis

A picture containing chart

Description automatically generated

pav. 6 Pritaikytų taisyklių grafikas, kai oro temperatūra 30C, greitis 50km/h, atstumas tarp automobilių 150m

A picture containing chart

Description automatically generated

pav. 7 Agregacijos grafikas, kai oro temperatūra 30C, greitis 50km/h, atstumas tarp automobilių 150m

Avarijos tikimybė = 18.57142857142857 %, Svorio centras

Avarijos tikimybė = 10.0 %, Maksimumo vidurkis

## Atsakymai

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Oro temperatūra, C | Automobilio greitis, km/h | Atstumas tarp automobilių | Avarijos tikimybė, % (Svorio centras) | Avarijos tikimybė, % (Svorio centras) |
| -10 | 130 | 5 | 81. 429 | 90 |
| 0 | 130 | 50 | 72.222 | 90 |
| 30 | 50 | 150 | 18.571 | 10 |

# Programos kodas

|  |
| --- |
| Main.py |
| import numpy as np  import skfuzzy as fuzz #pip install scikit-fuzzy  import matplotlib.pyplot as plt  def turn\_off\_top\_right\_axes(ax0, ax1, ax2):      for ax in (ax0, ax1, ax2):          ax.spines['top'].set\_visible(False)          ax.spines['right'].set\_visible(False)          ax.get\_xaxis().tick\_bottom()          ax.get\_yaxis().tick\_left()  def plot\_input\_graphs(x\_carSpeed, x\_weather, x\_crashProb, x\_distanceBetweenCars, carSpeed\_lo, carSpeed\_md, carSpeed\_hi, weather\_lo, weather\_md, weather\_hi, crashProb\_lo,                        crashProb\_md, crashProb\_hi, distanceBetweenCars\_lo, distanceBetweenCars\_md, distanceBetweenCars\_hi ):      fig, (ax0, ax1, ax2, ax3) = plt.subplots(nrows=4, figsize=(8, 9))      ax0.plot(x\_carSpeed, carSpeed\_lo, 'b', linewidth=1.5, label='Šalta')      ax0.plot(x\_carSpeed, carSpeed\_md, 'g', linewidth=1.5, label='Šilta')      ax0.plot(x\_carSpeed, carSpeed\_hi, 'r', linewidth=1.5, label='Karšta')      ax0.set\_title('Automobilio greitis, km/h')      ax0.legend()      ax1.plot(x\_weather, weather\_lo, 'b', linewidth=1.5, label='Mažas')      ax1.plot(x\_weather, weather\_md, 'g', linewidth=1.5, label='Vidutinis')      ax1.plot(x\_weather, weather\_hi, 'r', linewidth=1.5, label='Didelis')      ax1.set\_title('Oro temperatūra, C')      ax1.legend()      ax2.plot(x\_crashProb, crashProb\_lo, 'b', linewidth=1.5, label='Mažas')      ax2.plot(x\_crashProb, crashProb\_md, 'g', linewidth=1.5, label='Vidutinis')      ax2.plot(x\_crashProb, crashProb\_hi, 'r', linewidth=1.5, label='Didelis')      ax2.set\_title('Avarijos tikimybė, %')      ax2.legend()      ax3.plot(x\_distanceBetweenCars, distanceBetweenCars\_lo, 'b', linewidth=1.5, label='Maža')      ax3.plot(x\_distanceBetweenCars, distanceBetweenCars\_md, 'g', linewidth=1.5, label='Vidutinė')      ax3.plot(x\_distanceBetweenCars, distanceBetweenCars\_hi, 'r', linewidth=1.5, label='Aukšta')      ax3.set\_title('Atstumas tarp automobilių, m')      ax3.legend()      plt.tight\_layout()      plt.show()  def plot\_applied\_rules\_graphs(x\_crashProb, crashProb0, crashProb\_lo, crashProb\_md, crashProb\_hi,                              crashProb\_activation\_lo1, crashProb\_activation\_lo2, crashProb\_activation\_lo3, crashProb\_activation\_lo4, crashProb\_activation\_lo5, crashProb\_activation\_lo6, crashProb\_activation\_lo7, crashProb\_activation\_lo8,                              crashProb\_activation\_md1, crashProb\_activation\_md2, crashProb\_activation\_md3, crashProb\_activation\_md4, crashProb\_activation\_md5, crashProb\_activation\_md6, crashProb\_activation\_md7, crashProb\_activation\_md8,                              crashProb\_activation\_hi1, crashProb\_activation\_hi2, crashProb\_activation\_hi3, crashProb\_activation\_hi4, crashProb\_activation\_hi5):      fig, ax0 = plt.subplots(figsize=(8, 3))      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_lo1, facecolor='b', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_lo, 'b', linewidth=0.5, linestyle='--', )      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_lo2, facecolor='b', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_lo, 'b', linewidth=0.5, linestyle='--', )      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_lo3, facecolor='b', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_lo, 'b', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_lo4, facecolor='b', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_lo, 'b', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_lo5, facecolor='b', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_lo, 'b', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_lo6, facecolor='b', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_lo, 'b', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_lo7, facecolor='b', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_lo, 'b', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_lo8, facecolor='g', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_lo, 'b', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_md1, facecolor='g', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_md, 'g', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_md2, facecolor='g', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_md, 'g', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_md3, facecolor='g', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_md, 'g', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_md4, facecolor='g', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_md, 'g', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_md5, facecolor='g', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_md, 'g', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_md6, facecolor='g', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_md, 'g', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_md7, facecolor='g', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_md, 'g', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_md8, facecolor='g', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_md, 'g', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_hi1, facecolor='r', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_hi, 'r', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_hi2, facecolor='r', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_hi, 'r', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_hi3, facecolor='r', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_hi, 'r', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_hi4, facecolor='r', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_hi, 'r', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_crashProb, crashProb0, crashProb\_activation\_hi5, facecolor='r', alpha=0.7)      ax0.plot(x\_crashProb, crashProb\_hi, 'r', linewidth=0.5, linestyle='--')      ax0.set\_title('Agregated membership and results')      for ax in (ax0,):          ax.spines['top'].set\_visible(False)          ax.spines['right'].set\_visible(False)          ax.get\_xaxis().tick\_bottom()          ax.get\_yaxis().tick\_left()      plt.tight\_layout()      plt.show()  def apply\_rules(x\_AvarijosTikimybe, automobilioGreitis\_Mazas, automobilioGreitis\_Vidutinis, automobilioGreitis\_Didelis, oroTemperatura\_Salta, oroTemperatura\_Silta, oroTemperatura\_Karsta,                  avarijosTikymybe\_Maza, avarijosTikymybe\_Vidutine, avarijosTikymybe\_Auksta, atstumasTarpAutomobiliu\_Mazas, atstumasTarpAutomobiliu\_Vidutinis, atstumasTarpAutomobiliu\_Didelis):      #Nr 1      active\_rule1 = np.fmin(np.fmax(oroTemperatura\_Salta,atstumasTarpAutomobiliu\_Mazas),automobilioGreitis\_Didelis)      crashProb\_activation\_hi1 = np.fmin(active\_rule1, avarijosTikymybe\_Auksta)      #Nr 2      active\_rule2 = np.fmin(np.fmax(oroTemperatura\_Salta,atstumasTarpAutomobiliu\_Vidutinis),automobilioGreitis\_Didelis)      crashProb\_activation\_hi2 = np.fmin(active\_rule2, avarijosTikymybe\_Auksta)      #Nr 3      active\_rule3 = np.fmin(np.fmax(oroTemperatura\_Salta,atstumasTarpAutomobiliu\_Didelis),automobilioGreitis\_Didelis)      crashProb\_activation\_hi3 = np.fmin(active\_rule3, avarijosTikymybe\_Auksta)      #Nr 4      active\_rule4 = np.fmin(np.fmax(oroTemperatura\_Silta,atstumasTarpAutomobiliu\_Mazas),automobilioGreitis\_Didelis)      crashProb\_activation\_hi4 = np.fmin(active\_rule4, avarijosTikymybe\_Auksta)      #Nr 5      active\_rule5 = np.fmin(np.fmax(oroTemperatura\_Karsta,atstumasTarpAutomobiliu\_Mazas),automobilioGreitis\_Didelis)      crashProb\_activation\_hi5 = np.fmin(active\_rule5, avarijosTikymybe\_Auksta)        #Nr 6      active\_rule6 = np.fmin(np.fmax(oroTemperatura\_Silta,atstumasTarpAutomobiliu\_Vidutinis),automobilioGreitis\_Didelis)      crashProb\_activation\_md1 = np.fmin(active\_rule6, avarijosTikymybe\_Vidutine)      #Nr 7      active\_rule7 = np.fmin(np.fmax(oroTemperatura\_Silta,atstumasTarpAutomobiliu\_Didelis),automobilioGreitis\_Didelis)      crashProb\_activation\_md2 = np.fmin(active\_rule7, avarijosTikymybe\_Vidutine)      #Nr 8      active\_rule8 = np.fmin(np.fmax(oroTemperatura\_Karsta,atstumasTarpAutomobiliu\_Vidutinis),automobilioGreitis\_Didelis)      crashProb\_activation\_md3 = np.fmin(active\_rule8, avarijosTikymybe\_Vidutine)      #Nr 9      active\_rule9 = np.fmin(np.fmax(oroTemperatura\_Karsta,atstumasTarpAutomobiliu\_Didelis),automobilioGreitis\_Didelis)      crashProb\_activation\_md4 = np.fmin(active\_rule9, avarijosTikymybe\_Vidutine)      #Nr 10      active\_rule10 = np.fmin(np.fmax(oroTemperatura\_Salta,atstumasTarpAutomobiliu\_Mazas),automobilioGreitis\_Mazas)      crashProb\_activation\_md5 = np.fmin(active\_rule10, avarijosTikymybe\_Vidutine)      #Nr 11      active\_rule11 = np.fmin(np.fmax(oroTemperatura\_Salta,atstumasTarpAutomobiliu\_Vidutinis),automobilioGreitis\_Mazas)      crashProb\_activation\_md6 = np.fmin(active\_rule11, avarijosTikymybe\_Vidutine)      #Nr 12      active\_rule12 = np.fmin(np.fmax(oroTemperatura\_Silta,atstumasTarpAutomobiliu\_Mazas),automobilioGreitis\_Mazas)      crashProb\_activation\_md7 = np.fmin(active\_rule12, avarijosTikymybe\_Vidutine)      #Nr 13      active\_rule13 = np.fmin(np.fmax(oroTemperatura\_Silta,atstumasTarpAutomobiliu\_Vidutinis),automobilioGreitis\_Mazas)      crashProb\_activation\_md8 = np.fmin(active\_rule13, avarijosTikymybe\_Vidutine)      #Nr 14      active\_rule14 = np.fmin(np.fmax(oroTemperatura\_Karsta,atstumasTarpAutomobiliu\_Mazas),automobilioGreitis\_Mazas)      crashProb\_activation\_lo1 = np.fmin(active\_rule14, avarijosTikymybe\_Maza)      #Nr 15      active\_rule15 = np.fmin(np.fmax(oroTemperatura\_Karsta,atstumasTarpAutomobiliu\_Vidutinis),automobilioGreitis\_Mazas)      crashProb\_activation\_lo2 = np.fmin(active\_rule15, avarijosTikymybe\_Maza)      #Nr 16      active\_rule16 = np.fmin(np.fmax(oroTemperatura\_Karsta,atstumasTarpAutomobiliu\_Vidutinis),automobilioGreitis\_Vidutinis)      crashProb\_activation\_lo3 = np.fmin(active\_rule16, avarijosTikymybe\_Maza)      #Nr 17      active\_rule17 = np.fmin(np.fmax(oroTemperatura\_Karsta,atstumasTarpAutomobiliu\_Vidutinis),automobilioGreitis\_Mazas)      crashProb\_activation\_lo4 = np.fmin(active\_rule17, avarijosTikymybe\_Maza)      #Nr 18      active\_rule18 = np.fmin(np.fmax(oroTemperatura\_Salta,atstumasTarpAutomobiliu\_Didelis),automobilioGreitis\_Mazas)      crashProb\_activation\_lo5 = np.fmin(active\_rule18, avarijosTikymybe\_Maza)      #Nr 19      active\_rule19 = np.fmin(np.fmax(oroTemperatura\_Silta,atstumasTarpAutomobiliu\_Didelis),automobilioGreitis\_Mazas)      crashProb\_activation\_lo6 = np.fmin(active\_rule19, avarijosTikymybe\_Maza)      #Nr 20      active\_rule20 = np.fmin(np.fmax(oroTemperatura\_Silta,atstumasTarpAutomobiliu\_Didelis),automobilioGreitis\_Vidutinis)      crashProb\_activation\_lo7 = np.fmin(active\_rule20, avarijosTikymybe\_Maza)      #Nr 21      active\_rule21 = np.fmin(np.fmax(oroTemperatura\_Karsta,atstumasTarpAutomobiliu\_Didelis),automobilioGreitis\_Vidutinis)      crashProb\_activation\_lo8 = np.fmin(active\_rule21, avarijosTikymybe\_Maza)      crashProb0 = np.zeros\_like(x\_AvarijosTikimybe)      plot\_applied\_rules\_graphs(x\_AvarijosTikimybe, crashProb0, avarijosTikymybe\_Maza, avarijosTikymybe\_Vidutine, avarijosTikymybe\_Auksta,                              crashProb\_activation\_lo1, crashProb\_activation\_lo2, crashProb\_activation\_lo3, crashProb\_activation\_lo4, crashProb\_activation\_lo5, crashProb\_activation\_lo6, crashProb\_activation\_lo7, crashProb\_activation\_lo8,                              crashProb\_activation\_md1, crashProb\_activation\_md2, crashProb\_activation\_md3, crashProb\_activation\_md4, crashProb\_activation\_md5, crashProb\_activation\_md6, crashProb\_activation\_md7, crashProb\_activation\_md8,                              crashProb\_activation\_hi1, crashProb\_activation\_hi2, crashProb\_activation\_hi3, crashProb\_activation\_hi4, crashProb\_activation\_hi5)      aggregated\_lo = np.fmax(crashProb\_activation\_lo1,                              np.fmax(crashProb\_activation\_lo2,                                      np.fmax(crashProb\_activation\_lo3,                                              np.fmax(crashProb\_activation\_lo4,                                                  np.fmax(crashProb\_activation\_lo5,                                                          np.fmax(crashProb\_activation\_lo6,                                                                  np.fmax(crashProb\_activation\_lo7,crashProb\_activation\_lo8)))))))      aggregated\_md = np.fmax(crashProb\_activation\_md1,                              np.fmax(crashProb\_activation\_md2,                                      np.fmax(crashProb\_activation\_md3,                                              np.fmax(crashProb\_activation\_md4,                                                  np.fmax(crashProb\_activation\_md5,                                                          np.fmax(crashProb\_activation\_md6,                                                                  np.fmax(crashProb\_activation\_md7,crashProb\_activation\_md8)))))))      aggregated\_hi = np.fmax(crashProb\_activation\_hi1,                              np.fmax(crashProb\_activation\_hi2,                                      np.fmax(crashProb\_activation\_hi3,                                              np.fmax(crashProb\_activation\_hi4,crashProb\_activation\_hi5))))      aggregated = np.fmax(aggregated\_lo,                           np.fmax(aggregated\_md, aggregated\_hi))      crashProb = fuzz.defuzz(x\_AvarijosTikimybe, aggregated, 'centroid')      crashProb\_activation = fuzz.interp\_membership(x\_AvarijosTikimybe, aggregated, crashProb)  # for plot      print("Avarijos tikimybė = "+str(fuzz.defuzz(x\_AvarijosTikimybe, aggregated, 'centroid'))+" %, Svorio centras")      print("Avarijos tikimybė = "+str(fuzz.defuzz(x\_AvarijosTikimybe, aggregated, 'mom'))+" %, Maksimumo vidurkis")      fig, ax0 = plt.subplots(figsize=(8, 3))      ax0.plot(x\_AvarijosTikimybe, avarijosTikymybe\_Maza, 'b', linewidth=0.5, linestyle='--', )      ax0.plot(x\_AvarijosTikimybe, avarijosTikymybe\_Vidutine, 'g', linewidth=0.5, linestyle='--')      ax0.plot(x\_AvarijosTikimybe, avarijosTikymybe\_Auksta, 'r', linewidth=0.5, linestyle='--')      ax0.fill\_between(x\_AvarijosTikimybe, crashProb0, aggregated, facecolor='Orange', alpha=0.7)      ax0.plot([crashProb, crashProb], [0, crashProb\_activation], 'k', linewidth=1.5, alpha=0.9)      ax0.set\_title('Aggregated membership and result (line)')      for ax in (ax0,):          ax.spines['top'].set\_visible(False)          ax.spines['right'].set\_visible(False)          ax.get\_xaxis().tick\_bottom()          ax.get\_yaxis().tick\_left()      plt.tight\_layout()      plt.show()  def execute():      # -----------DATA-------------#      x\_carSpeed = np.arange(50, 131, 1)      x\_weather = np.arange(-10, 31, 1)      x\_distanceBetweenCars = np.arange(1,151,1)      x\_crashProb = np.arange(0, 101, 1)      # ----------------------------#      # -------GRAPHSDATA-----------#      carSpeed\_lo = fuzz.trapmf(x\_carSpeed, [50, 50, 70, 90])      carSpeed\_md = fuzz.trimf(x\_carSpeed, [70, 90, 110])      carSpeed\_hi = fuzz.trapmf(x\_carSpeed, [90, 110, 130, 130])      weather\_lo = fuzz.trapmf(x\_weather, [-10, -10, 0, 10])      weather\_md = fuzz.trimf(x\_weather, [0, 10, 20])      weather\_hi = fuzz.trapmf(x\_weather, [10, 20, 30, 30])      crashProb\_lo = fuzz.trapmf(x\_crashProb, [0, 0, 20, 50])      crashProb\_md = fuzz.trimf(x\_crashProb, [20, 50, 80])      crashProb\_hi = fuzz.trapmf(x\_crashProb, [50, 80, 100, 100])      distanceBetweenCars\_lo = fuzz.trapmf(x\_distanceBetweenCars,[0, 0, 37.5, 75])      distanceBetweenCars\_md = fuzz.trimf(x\_distanceBetweenCars,[37.5, 75, 112.5])      distanceBetweenCars\_hi = fuzz.trapmf(x\_distanceBetweenCars,[75,112.5,150,150])      # -----------------------------#      plot\_input\_graphs(x\_carSpeed, x\_weather, x\_crashProb, x\_distanceBetweenCars, carSpeed\_lo, carSpeed\_md, carSpeed\_hi, weather\_lo, weather\_md,                        weather\_hi, crashProb\_lo, crashProb\_md, crashProb\_hi,distanceBetweenCars\_lo, distanceBetweenCars\_md, distanceBetweenCars\_hi)      temp = 30      speed = 50      distance = 150      carSpeed\_level\_lo = fuzz.interp\_membership(x\_carSpeed, carSpeed\_lo, speed)      carSpeed\_level\_md = fuzz.interp\_membership(x\_carSpeed, carSpeed\_md, speed)      carSpeed\_level\_hi = fuzz.interp\_membership(x\_carSpeed, carSpeed\_hi, speed)      weather\_level\_lo = fuzz.interp\_membership(x\_weather, weather\_lo, temp)      weather\_level\_md = fuzz.interp\_membership(x\_weather, weather\_md, temp)      weather\_level\_hi = fuzz.interp\_membership(x\_weather, weather\_hi, temp)      distanceBetweenCars\_level\_lo = fuzz.interp\_membership(x\_distanceBetweenCars, distanceBetweenCars\_lo, distance)      distanceBetweenCars\_level\_md = fuzz.interp\_membership(x\_distanceBetweenCars, distanceBetweenCars\_md, distance)      distanceBetweenCars\_level\_hi = fuzz.interp\_membership(x\_distanceBetweenCars, distanceBetweenCars\_hi, distance)      apply\_rules(x\_crashProb, carSpeed\_level\_lo, carSpeed\_level\_md, carSpeed\_level\_hi, weather\_level\_lo, weather\_level\_md,                  weather\_level\_hi, crashProb\_lo, crashProb\_md, crashProb\_hi, distanceBetweenCars\_level\_lo,distanceBetweenCars\_level\_md, distanceBetweenCars\_level\_hi)  if \_\_name\_\_ == "\_\_main\_\_":      execute() |