

**Kauno technologijos universitetas**

Informatikos fakultetas

Intelektikos pagrindai (P176B101)

**1 laboratorinis darbas**

Atliko: IFF-6/11 gr. studentas Nerijus Dulkė

Priėmė: doc. Germanas Budnikas

KAUNAS

2019

# Pirma užduotis

## Užduotis

Duota užduotis:

Pelės:

* Jos baltos ir pilkos
* Jų yra daug

Katinas:

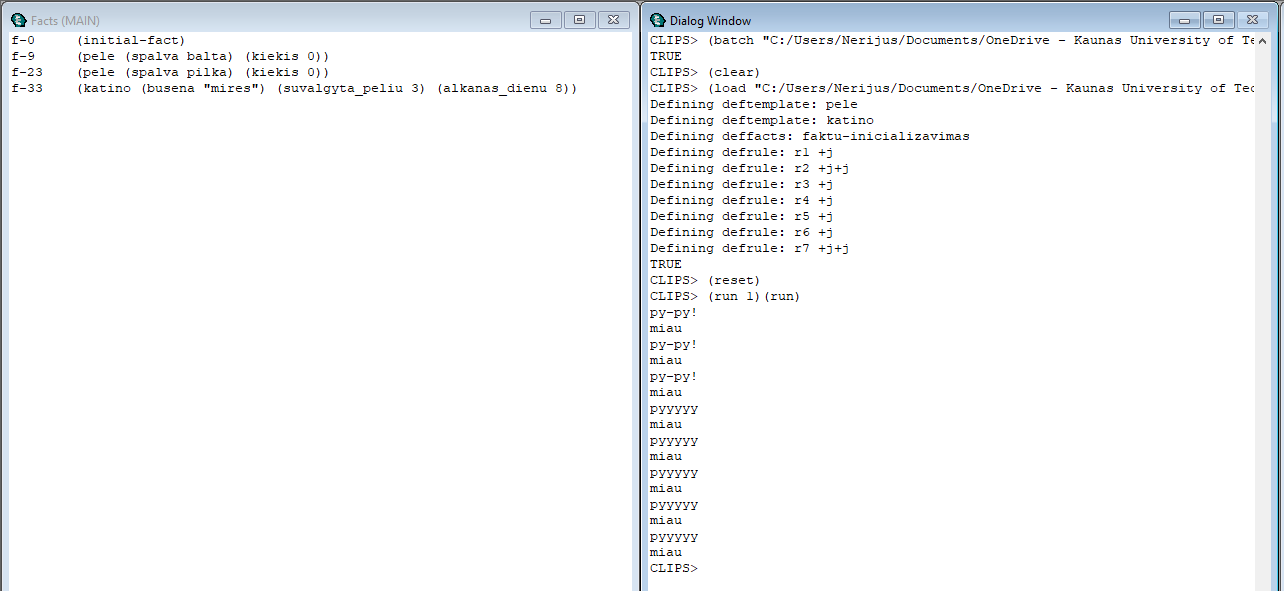
* Alkanas
* Kai katinas alkanas, jis nori valgyti
* Kai katinas nori valgyti ir yra pelių, jis valgo peles baltas ir pilkas
* Prieš mirt baltoji pelė *šaukia* „py-py!“, pilkoji – „pyyyyyy“  
  Po valgio katinas paprastai *sako* „miau“ ;-)
* Kai katinas suvalgo 5 peles, jis tampa storu katinu
* Stori katinai pelių nevalgo

Užduotyje reikėjo savarankiškai atlikti papildymus:

1. Stori katinai po miego tampa alkani ir pamiršta, kiek suvalgė pelių
2. Nevalgę bet vis dar norintys valgyti katinai po 7 dienų miršta

## Realizacija

Išvestis kai programa baigia vykdyti:



Programos kodas:

(deftemplate pele (slot spalva) (slot kiekis) )

(deftemplate katino (slot busena) (slot suvalgyta\_peliu) )

(deftemplate skaitliukas (slot dienu\_kiekis))

(deffacts faktu-inicializavimas

(pele (spalva pilka) (kiekis 5))

(pele (spalva balta) (kiekis 3))

(katino (busena "alkanas") (suvalgyta\_peliu 0))

(skaitliukas (dienu\_kiekis 0))

)

(defrule r1 "Kai katinas alkanas, jis nori valgyti"

?fact-id <- (katino (busena ?busena))

(test (eq ?busena "alkanas"))

=>

(modify ?fact-id (busena "nori valgyti"))

)

(defrule r2 "Kai katinas nori valgyti ir yra peliu, jis valgo peles"

?fact-id1 <- (katino (busena "nori valgyti") (suvalgyta\_peliu ?suvalgyta))

?fact-id2 <- (pele (spalva ?spalva) (kiekis ?kiekis))

(test (> ?kiekis 0))

=>

(if (eq ?spalva balta) then (printout t "py-py!" crlf)

else (printout t "pyyyyy" crlf))

(modify ?fact-id2 (kiekis (- ?kiekis 1)) )

(modify ?fact-id1 (suvalgyta\_peliu (+ ?suvalgyta 1)) )

(printout t "miau" crlf)

)

(defrule r3 "kai katinas suvalgo 5 peles, jis tampa storu katinu"

(declare (salience 10))

?fact-id1 <- (katino (busena "nori valgyti") (suvalgyta\_peliu ?suvalgyta))

(test (= ?suvalgyta 5))

=>

(modify ?fact-id1 (busena "storas"))

)

(defrule r4 "Kai katinas storas jis miega"

  ?fact-id <- (katino (busena ?busena))

  (test (eq ?busena "storas"))

  =>

  (bind ?busena "miega")

  (modify ?fact-id (busena ?busena))

)

(defrule r5 "Stori katinai po miego tampa alkani ir pamirsta kiek suvalge peliu"

  ?fact-id <- (katino (busena ?busena))

  (test (eq ?busena "miega"))

  =>

  (modify ?fact-id (busena "alkanas")(suvalgyta\_peliu 0))

)

(defrule r6 "Po 7 dienu norintys valgyt katinai mirsta"

  ?fact-id1 <- (katino (busena ?busena))

  ?fact-id2 <- (skaitliukas (dienu\_kiekis ?dienu\_kiekis))

  (test (eq ?busena "nori valgyti"))

(test (> ?dienu\_kiekis 7))

  =>

  (modify ?fact-id1 (busena "mires"))

)

(defrule r7 "Skaitliukas dideja kai nera peliu"

  ?fact-id1 <- (pele (kiekis ?kiekis))

  ?fact-id2 <- (skaitliukas (dienu\_kiekis ?dienu\_kiekis))

  (test (= ?kiekis 0))

  =>

  (modify ?fact-id2 (dienu\_kiekis (+ ?dienu\_kiekis 1)))

)

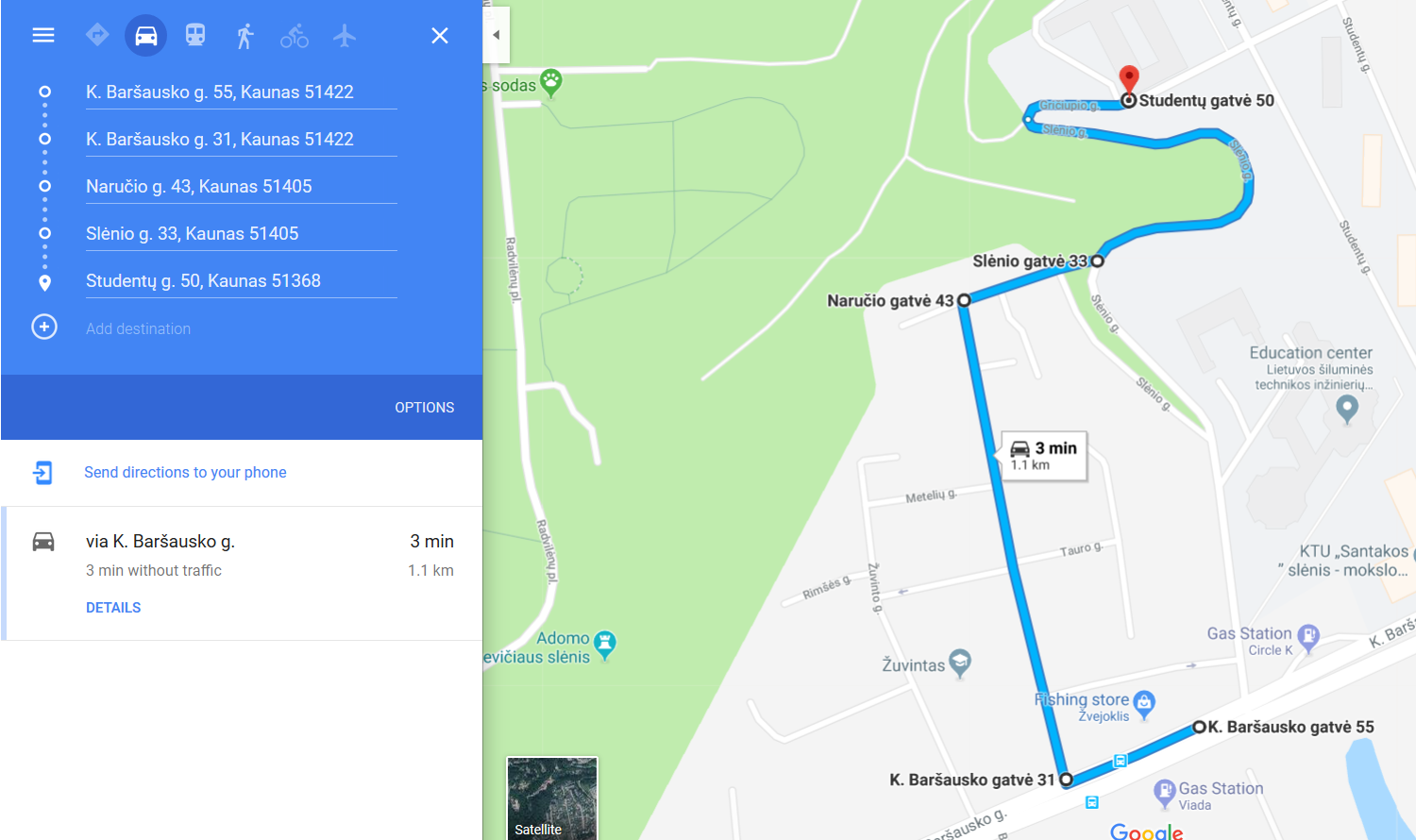
# Individuoli užduotis

## Užduotis

Sukurti taisyklėmis grįstą sistemą, aprašančią autonominės mašinos judėjimą pasirinktu maršrutu. Maršrute, kuris skaidomas į fragmentus ties kiekvieną sankryžą, gali būti kliūčių - pėsčiųjų, šviesoforų, kitų mašinų, spec. tarnybų automobilių. Judėjimas maršruto atkarpoje leidžiamas tik nesant kliūtims. Sistemoje iliustruoti autonominės mašinos judėjimą nurodytu maršrutu informuojant apie pravažiuojamas gatves ir kliūčių sumažėjimą važiuojamoje maršruto atkarpoje. Numatyti galimybę programoje papildyti norimą kiekį kliūčių bei pakoreguoti maršrutą.

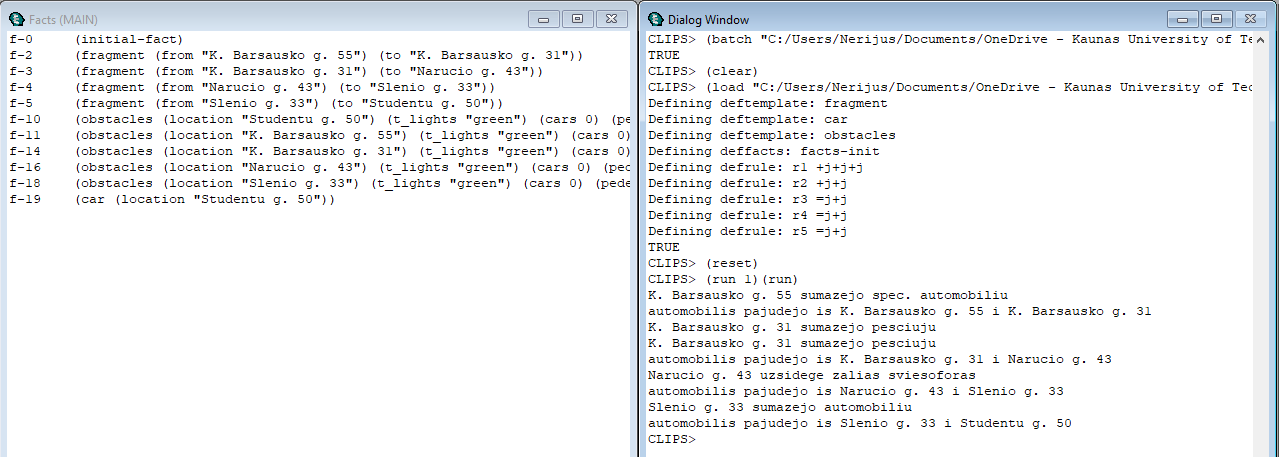
Maršrutui nustatyti parinkti savo namų vietą. Nustatyti judėjimo kryptį per x sankryžų, kur x – raidžių skaičius jūsų varde arba pavardėje (kas trumpiau).

Pasirinktas maršrutas:



## Realizacija

Išvestis kai programa baigia vykdyti:



Programos kodas:

(deftemplate fragment (slot from) (slot to))

(deftemplate car (slot location))

(deftemplate obstacles (slot location) (slot t\_lights) (slot cars) (slot pedestrians) (slot spec\_service))

(deffacts facts-init

    (car (location "K. Barsausko g. 55"))

    (fragment (from "K. Barsausko g. 55") (to "K. Barsausko g. 31"))

    (fragment (from "K. Barsausko g. 31") (to "Narucio g. 43"))

    (fragment (from "Narucio g. 43") (to "Slenio g. 33"))

    (fragment (from "Slenio g. 33") (to "Studentu g. 50"))

    (obstacles (location "K. Barsausko g. 55")  (t\_lights "green")  (cars 0) (pedestrians 0) (spec\_service 1))

    (obstacles (location "K. Barsausko g. 31")  (t\_lights "green")  (cars 0) (pedestrians 2) (spec\_service 0))

    (obstacles (location "Narucio g. 43")       (t\_lights "red")    (cars 0) (pedestrians 0) (spec\_service 0))

    (obstacles (location "Slenio g. 33")        (t\_lights "green")  (cars 1) (pedestrians 0) (spec\_service 0))

    (obstacles (location "Studentu g. 50")      (t\_lights "green")  (cars 0) (pedestrians 0) (spec\_service 0))

)

(defrule r1 "Move"

    ?fact-id1 <- (fragment (from ?from)(to ?to))

    ?fact-id2 <- (car (location ?location))

    ?fact-id3 <- (obstacles (location ?obs\_location) (t\_lights "green") (cars 0) (pedestrians 0) (spec\_service 0))

    (test (eq ?location ?from))

    (test (eq ?location ?obs\_location))

    =>

    (modify ?fact-id2 (location ?to))

    (printout t "automobilis pajudejo is " ?from " i " ?to crlf)

)

(defrule r2 "Toggle traffic lights"

    ?fact-id1 <- (car (location ?location))

    ?fact-id2 <- (obstacles (location ?obs\_location) (t\_lights "red"))

    (test (eq ?location ?obs\_location))

    =>

    (modify ?fact-id2 (t\_lights "green"))

    (printout t ?location " uzsidege zalias sviesoforas" crlf)

)

(defrule r3 "Move cars"

    ?fact-id1 <- (car (location ?location))

    ?fact-id2 <- (obstacles (location ?obs\_location) (cars ?cars))

    (test (eq ?location ?obs\_location))

    (test (> ?cars 0))

    =>

    (modify ?fact-id2 (cars (- ?cars 1)))

    (printout t ?location " sumazejo automobiliu" crlf)

)

(defrule r4 "Move spec\_services"

    ?fact-id1 <- (car (location ?location))

    ?fact-id2 <- (obstacles (location ?obs\_location) (spec\_service ?spec\_service))

    (test (eq ?location ?obs\_location))

    (test (> ?spec\_service 0))

    =>

    (modify ?fact-id2 (spec\_service (- ?spec\_service 1)))

    (printout t ?location " sumazejo spec. automobiliu" crlf)

)

(defrule r5 "Move pedestrians"

    ?fact-id1 <- (car (location ?location))

    ?fact-id2 <- (obstacles (location ?obs\_location) (pedestrians ?pedestrians))

    (test (eq ?location ?obs\_location))

    (test (> ?pedestrians 0))

    =>

    (modify ?fact-id2 (pedestrians (- ?pedestrians 1)))

    (printout t ?location " sumazejo pesciuju" crlf)

)