МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федерально автономное образовательное учреждение высшего образования

«Севастопольский государственный университет»

кафедра Информационных систем

Куркчи Ариф Эрнестович

Институт информационных технологий и управления в технических системах

курс 4 группа ИС/б-41-о

09.03.02 Информационные системы и технологии (уровень бакалавриата)

ОТЧЕТ

по лабораторной работе №4

по дисциплине «Методы и системы искусственного интеллекта»

на тему «ЕЯ ДОСТУП К БАЗЕ ДАННЫХ НА ОСНОВЕ АЛГОРИТМА

СОПОСТАВЛЕНИЯ С ОБРАЗЦОМ»

Отметка о зачете \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_

(дата)

Руководитель практикума

старший преподаватель   Волкова А.В.

(должность) (подпись) (инициалы, фамилия)

Севастополь 2017

1. ЦЕЛЬ РАБОТЫ

Исследование алгоритма сопоставления с образцом и особенностей его применения для формирования запросов к базам данных, а также для организации доступа к базам данных на ограниченном подмножестве естественного языка.

1. ПОСТАНОВКА ЗАДАЧИ

Для базы данных, созданной в лабораторной работе 3, необходимо написать на языке Лисп интерфейс, который позволяет выполнять ЕЯ-запросы с помощью алгоритма сопоставления с образцом. Кроме запроса, заданного по варианту задания, предусмотреть 5-6 различных дополнительных запросов.

1. ИСХОДНЫЙ КОД

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | 1 | (defvar \*db\* nil) | | 2 |  | | 3 | (*defun* insert (dest train time) | | 4 | (push (list :dest dest :train train :time time) \*db\*) | | 5 | ) | | 6 |  | | 7 | (*defun* savef (filename) | | 8 | (with-open-file (out filename :direction :output :if-exists :supersede) | | 9 | (with-standard-io-syntax | | 10 | (print \*db\* out) | | 11 | ) | | 12 | ) | | 13 | ) | | 14 |  | | 15 | (*defun* loadf (filename) | | 16 | (with-open-file (in filename) | | 17 | (with-standard-io-syntax | | 18 | (setf \*db\* (read in)) | | 19 | ) | | 20 | ) | | 21 | ) | | 22 |  | | 23 | (*defun* select\* () | | 24 | (format t "~%") | | 25 | (format t "~{~{~a:~10t~a~%~}~%~}" \*db\*) | | 26 | ) | | 27 |  | | 28 | (*defun* where(&key dest train time) | | 29 | #'(lambda (row) | | 30 | (and | | 31 | (if dest  (equal (getf row :dest)  dest)  t) | | 32 | (if train (equal (getf row :train) train) t) | | 33 | (if time  (equal (getf row :time)  time)  t) | | 34 | ) | | 35 | ) | | 36 | ) | | 37 |  | | 38 | (*defun* update (where-func &key dest train time) | | 39 | (setf \*db\* | | 40 | (mapcar | | 41 | #'(lambda (row) | | 42 | (when (funcall where-func row) | | 43 | (if dest  (setf (getf row :dest)  dest)) | | 44 | (if train (setf (getf row :train) train)) | | 45 | (if time  (setf (getf row :time)  time)) | | 46 | ) | | 47 | row | | 48 | ) | | 49 | \*db\* | | 50 | ) | | 51 | ) | | 52 | ) | | 53 |  | | 54 | (*defun* selectByDest (\_dest) | | 55 | (if (eq nil (setf rows (remove-if-not #'(lambda (row) (equal (getf row :dest) \_dest)) \*db\*))) | | 56 | (print "Поезда туда не ездят") | | 57 | (print rows) | | 58 | ) | | 59 | ) | | 60 |  | | 61 | (*defun* match (p d) | | 62 | (cond | | 63 | ;;правило 1 | | 64 | ((and (null p) (null d)) t) | | 65 | ;;правило 2 | | 66 | ((and (null d) | | 67 | (eq (car p) '$) | | 68 | (null (cdr p))) t) | | 69 | ;;один из списков исчерпан | | 70 | ((or (null p) (null d)) nil) | | 71 | ;;правило 3 и правило 4 | | 72 | ((or (equal (car p) '?) | | 73 | (equal (car p) (car d))) | | 74 | (match (cdr p) (cdr d))) | | 75 | ;;правило 5 и 6 | | 76 | ((eq (car p) '$) | | 77 | (cond ((match (cdr p) d) t) | | 78 | ((match p (cdr d)) t))) | | 79 | ;;правило 7 - сопоставление списков,включающих подсписки | | 80 | ((and (not (atom (car p))) | | 81 | (not (atom (car d))) | | 82 | (match (car p) (car d))) | | 83 | (match (cdr p) (cdr d)) ) | | 84 | ;;правило 8 – подстановка значения в переменную | | 85 |  | | 86 | ((and (atom (car p)) | | 87 | (eq (car-letter (car p)) #\?) | | 88 | (match (cdr p)(cdr d))) | | 89 | (set (cdr-name (car p)) (car d)) t) | | 90 | ;;правило 9 - подстановка сегмента значений в переменную | | 91 |  | | 92 | ((and (atom (car p)) | | 93 | (eq (car-letter (car p)) #\$)) | | 94 | (cond ((match (cdr p)(cdr d)) | | 95 | (set (cdr-name (car p)) (list (car d))) | | 96 | t) | | 97 | ((match p (cdr d)) | | 98 | (set (cdr-name (car p)) | | 99 | (cons (car d)(eval (cdr-name (car p))))) | | 100 | t))) | | 101 | ;; правило 10 - обработка пакета ограничений, если в пакете есть «?» | | 102 | ((and (not(atom (car p))) | | 103 | (eq (caar p) 'restrict) | | 104 | (eq (cadar p) '?) | | 105 | (and-to-list | | 106 | (mapcar #'(lambda (pred) | | 107 | (funcall pred (car d))) (cddar p)))) | | 108 | (match (cdr p)(cdr d))) | | 109 | ;; правило 11 - обработка пакета ограничений, если в пакете есть «?V» | | 110 | ;; например: (match '((restrict ?V integerp evenp) b c) '(36 b c)) | | 111 | ((and (not (atom (car p))) | | 112 | (not (atom d)) | | 113 | (eq (caar p) 'restrict) | | 114 | (eq (car-letter (cadar p)) #\?) | | 115 | (and-to-list | | 116 | (mapcar #'(lambda (pred) | | 117 | (funcall pred (car d))) (cddar p))) | | 118 | (match (cdr p)(cdr d))) | | 119 | (set (cdr-name (cadar p)) (car d)) | | 120 | t) | | 121 | )) | | 122 | ;;;;<2. Вспомогательные функции | | 123 | ;;; выделение первой литеры из имени | | 124 | (*defun* car-letter (x) (if (not (numberp x)) (car (coerce (string x) 'list)))) | | 125 | ;;; возвращает имя без первой | | 126 | (*defun* cdr-name (x) | | 127 | (intern (coerce (cdr (coerce (string x) 'list)) 'string))) | | 128 | ;;; проверяет, все ли элементы списка lis имеют значение T | | 129 | (*defun* and-to-list ( lis ) | | 130 | ;lis - список логических значений | | 131 | (let ((res t)) | | 132 | (dolist (temp lis res) | | 133 | (setq res (and res temp))))) | | 134 |  | | 135 | (*defun* get-matches (p database) | | 136 | (remove-if-not #'(lambda (record) (match p record)) database) | | 137 | ) | | 138 |  | | 139 | (*defun* query (q) | | 140 | (cond | | 141 | ((match `($ загрузить $) q) | | 142 | (loadf "test.txt")) | | 143 | ((match `($ сохранить $) q) | | 144 | (savef "test.txt")) | | 145 | ((match `(Добавить $ номер ?train в $ ?dest $ отправляется $ ?time) q) | | 146 | (insert dest train time)) | | 147 | ((match `($ город $ ?dest) q) | | 148 | (setf temp (get-matches `($ :dest ,dest $) \*db\*)) | | 149 | (if (null temp) "Поезда туда не едут" temp)) | | 150 | ((match `($ номер $ ?train) q) | | 151 | (setf temp (get-matches `($ :train ,train $) \*db\*)) | | 152 | (if (null temp) "Поездов с таким номером нет" temp)) | | 153 | ((match `($ отправляется $ ?time) q) | | 154 | (setf temp (get-matches `($ :time ,time $) \*db\*)) | | 155 | (if (null temp) "В такое время никто не едет" temp)) | | 156 | ((match `($ все $) q) | | 157 | (select\*)) | | 158 | ) | | 159 | ) | | 160 |  | | 161 | (*defun* selectQuery (q) | | 162 | (setf temp (query q)) | | 163 | (if (listp temp) | | 164 | (format t "~{~{~a:~10t~a~%~}~%~}" temp) | | 165 | (format t "~a~%" temp) | | 166 | ) | | 167 | ) | |  | |  |

1. РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 1 | (query '(Загрузить базу)) | | 2 |  | | 3 | (format t "Добавим 2 поезда~%") | | 4 | (query '(Добавить поезд номер 17 в "Москва" который отправляется в "11:40")) | | 5 | (query '(Добавить поезд номер 418 в "Симферополь" который отправляется в "9:15")) | | 6 |  | | 7 | (format t "Какие поезда идут в город Симферополь~%") | | 8 | (selectQuery '(Какие поезда идут в город "Симферополь")) | | 9 |  | | 10 | (format t "Найти поезд номер 418~%") | | 11 | (selectQuery '(Найти поезд номер 418)) | | 12 |  | | 13 | (format t "Кто отправляется в 12:20~%") | | 14 | (selectQuery '(Кто отправляется в "12:20")) | | 15 |  | | 16 | (format t "Показать все поезда~%") | | 17 | (selectQuery '(Показать все поезда)) | |  |

Добавим 2 поезда

Какие поезда идут в город Симферополь

DEST: Симферополь

TRAIN: 418

TIME: 9:15

DEST: Симферополь

TRAIN: 3

TIME: 13:00

Найти поезд номер 418

DEST: Симферополь

TRAIN: 418

TIME: 9:15

Кто отправляется в 12:20

DEST: Москва

TRAIN: 1

TIME: 12:20

Показать все поезда

DEST: Симферополь

TRAIN: 418

TIME: 9:15

DEST: Москва

TRAIN: 17

TIME: 11:40

DEST: Симферополь

TRAIN: 3

TIME: 13:00

DEST: Сочи

TRAIN: 2

TIME: 12:30

DEST: Москва

TRAIN: 1

TIME: 12:20

ВЫВОДЫ

В ходе работы был исследован алгоритм сопоставления с образцом и особенности его применения для формирования запросов к базам данных, организован доступ к базе данных с применением ограниченного подмножества естественного языка.