Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Пермский национальный** **исследовательский политехнический университет»**

Факультет: Прикладной математики и механики

Кафедра: Вычислительной математики, механики и биомеханики

Направление: 09.03.02 «Информационные системы и технологии»

Профиль бакалавриата: «Информационные системы и технологии»

**Лабораторные работы**

по дисциплине

«**Интеллектуальные ИСИТ**»

Выполнила студентка гр. ИСТ-19-1бзу

Потеряева М.В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Ф.И.О.)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

*(подпись)*

**Принял (преподаватель по дисциплине):**

асс. Истомин Д.А.

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(должность, ФИО)*

*\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(оценка) (подпись)*

*\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(дата)*

**Пермь, 2022**

Содержание

[Лабораторная работа №1 «Проектирование и реализация продукционной базы знаний» 3](#_Toc103625240)

[Лабораторная работа №2 «Проектирование и реализация оболочки экспертной системы, использующей продукционный способ представления знаний» 15](#_Toc103625241)

[Лабораторная работа №3 «Проектирование базы знаний и реализация ЭС на основе семантических сетей и фреймов» 19](#_Toc103625242)

[Лабораторная работа №4 «Реализация моделей принятия коллективных решений» 22](#_Toc103625243)

[Лабораторная работа №5 «Реализация нейронных сетей для решения задачи распознавания образов» 24](#_Toc103625244)

# Лабораторная работа №1 «Проектирование и реализация продукционной базы знаний»

**Цель работы:** сформировать у студентов способность проектирования и разработки базы знаний, основанной на продукционной модели их представления.

**Задание:**

1. Выбрать предметную область, в рамках которой планируется построить БЗ. Должна решаться задача, например, диагностики или прогнозирования.
2. Визуально оформить граф И/ИЛИ.
3. Описать правила БЗ в текстовом виде в нотации IF-THEN.
4. Ознакомиться с тестовыми примерами, которые идут совместно с поставкой среды CLIPS.
5. В инструменте разработки ЭС CLIPS реализовать сформированные правила.

**Выполнение работы:**

На настоящий момент наибольшее распространение получили системы, основанные на правилах. В таких системах знания представлены не с помощью декларативного представления знаний, а в форме многочисленных правил, которые указывают, какие заключения должны быть сделаны или не сделаны в различных ситуациях.

Каждое правило обозначается именем, далее за именем находится часть IF правила (условная часть или антецедент). Вслед за THEN правила находится список действий, которые должны быть выполнены после запуска правила (консеквент).

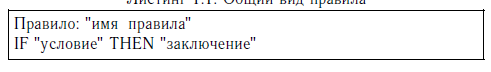
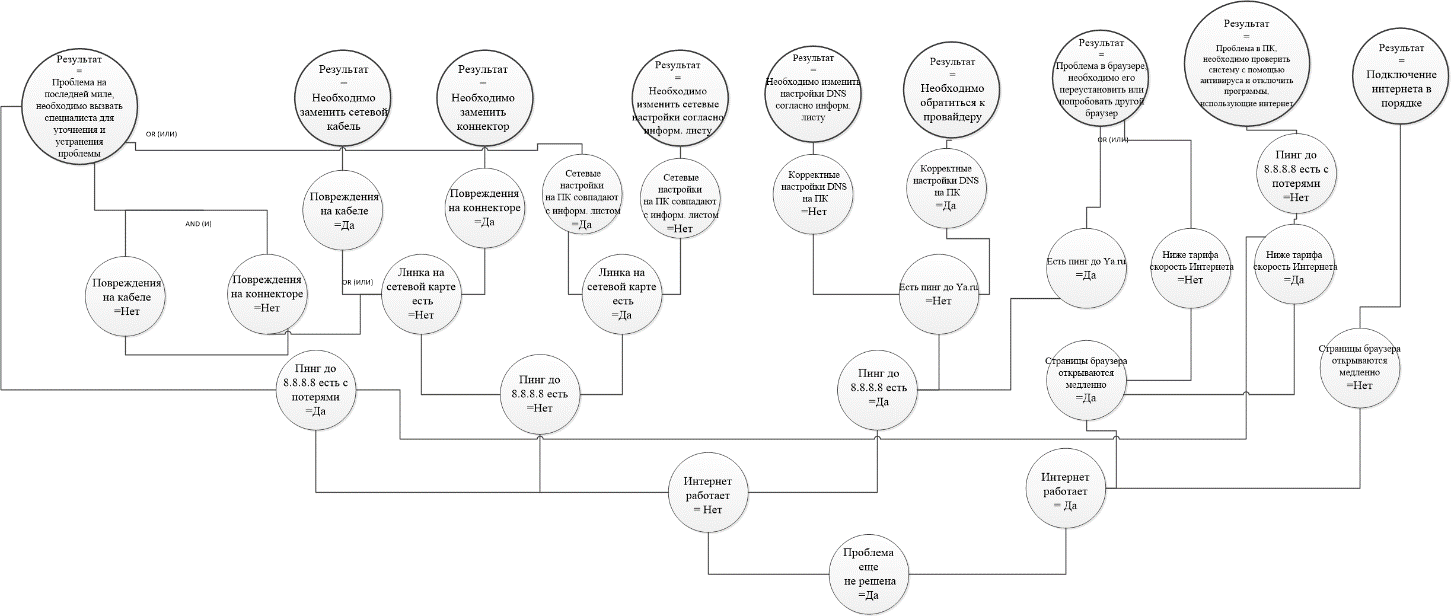


Рисунок 1 Общий вид правила

В системе, основанной на правилах, машина логического вывода (МЛВ) определяет, какие анцеденты правил выполняются согласно фактам. В качестве стратегий решения задач в экспертных системах обычно используются два общих метода логического вывода: прямой логический вывод (от фактов к заключениям) и обратный логический вывод (от гипотезы к фактам).

В качестве предметной области для проектирования и реализации продукционной базы знаний выбрана область диагностики работы Интернета, а конкретно решается задача выявления проблем работы Интернета и их решение.

Рассмотри построение графа И/ИЛИ, который будет использован для более наглядного представления продукционных знаний. Часть графа будет выглядеть следующим образом.



Выполним описание правил базы знаний (БЗ) в текстовом виде в нотации IF-THEN.

1. IF «Проблема еще не решена» = «Да» THEN «Работает Интернет» = «Вопрос: Работает Интернет (да/нет)?»
2. IF «Интернет работает» = «Нет» AND «Проблема еще не решена» = «Да» THEN «Есть пинг» = «Вопрос: Есть пинг до 8.8.8.8 (да/нет/есть потери)?»
3. IF «Интернет работает» = «Нет» AND «Есть пинг до 8.8.8.8» = «Нет» AND «Проблема еще не решена» = «Да» THEN «Есть линк» = «Вопрос: Есть линк на сетевой карте (да/нет)?»
4. IF Интернет работает» = «Нет» AND «Есть пинг до 8.8.8.8» = «Нет» AND «Проблема еще не решена» = «Да» AND «Есть линк» = «Нет» THEN «Повреждения на кабеле» = «Вопрос: Есть повреждения на кабеле (да/нет)?
5. IF Интернет работает» = «Нет» AND «Есть пинг до 8.8.8.8» = «Нет» AND «Проблема еще не решена» = «Да» AND «Есть линк» = «Нет» THEN «Повреждения на коннекторе» = «Вопрос: Есть повреждения на коннекторе (да/нет)?
6. IF Интернет работает» = «Нет» AND «Есть пинг до 8.8.8.8» = «Нет» AND «Проблема еще не решена» = «Да» AND «Есть линк» = «Да» THEN «Сетевые настройки ПК верные» = «Вопрос: Совпадают сетевые настройки на ПК с информационным листом (да/нет)?
7. IF Интернет работает» = «Нет» AND «Есть пинг до 8.8.8.8» = «Да» AND «Проблема еще не решена» = «Да» THEN «Пинг ya.ru» = «Вопрос: Пингуется ya.ru (да/нет)?
8. IF Интернет работает» = «Нет» AND «Есть пинг до 8.8.8.8» = «Да» AND «Пинг ya.ru» = «Нет» AND «Проблема еще не решена» = «Да» THEN «DNS настройки корректные» = «Вопрос: Корректные настройки DNS на ПК (да/нет)?
9. IF Интернет работает» = «Да» AND «Проблема еще не решена» = «Да» THEN «Загрузка страниц медленная» = «Вопрос: Медленно открываются страницы в браузере (да/нет)?»
10. IF Интернет работает» = «Да» AND «Загрузка страниц медленная» = «Да» AND «Проблема еще не решена» = «Да» THEN «Скорость Интернета ниже тарифа» = «Вопрос: Измеренная скорость ниже тарифа (да/нет)?»
11. IF Интернет работает» = «Да» AND «Загрузка страниц медленная» = «Да» AND THEN «Скорость Интернета ниже тарифа» = «Да» AND «Проблема еще не решена» = «Да» THEN «Пинг 8.8.8.8 с потерями» = «Вопрос: При пинге 8.8.8.8 наблюдаются потери (да/нет)?»
12. IF «Повреждения на коннекторе» = «Да» AND «Проблема еще не решена» = «Да» THEN «Решение найдено: необходимо заменить коннектор»
13. IF «Повреждения на кабеле» = «Да» AND «Проблема еще не решена» = «Да» THEN «Решение найдено: необходимо заменить сетевой кабель»
14. IF «Сетевые настройки ПК верные» = «Нет» AND «Проблема еще не решена» = «Да» THEN «Решение найдено: необходимо изменить сетевые настройки согласно информационному листу»
15. IF «DNS настройки корректные» = «Да» AND «Проблема еще не решена» = «Да» THEN «Решение найдено: необходимо обратиться к провайдеру»
16. F «DNS настройки корректные» = «Нет» AND «Проблема еще не решена» = «Да» THEN «Решение найдено: необходимо изменить настройки DNS согласно информационному листу»
17. IF «Загрузка страниц медленная» = «Нет» AND «Проблема еще не решена» = «Да» THEN «Решение найдено: подключение Интернета в порядке»
18. IF «Пинг 8.8.8.8 с потерями» = «Нет» AND «Проблема еще не решена» = «Да» THEN «Решение найдено: проблема в ПК, необходимо проверить систему с помощью антивируса и отключить программы, использующие интернет»
19. IF «(«Скорость Интернета ниже тарифа» = «Нет» OR «Пинг ya.ru» = «Да») AND «Проблема еще не решена» = «Да» THEN «Решение найдено: проблема в браузере, необходимо его переустановить или попробовать другой браузер»
20. IF «((«Повреждения на кабеле» = «Нет» AND «Повреждения на коннекторе» = «Нет») OR «Сетевые настройки ПК верные» = «Да») AND «Пинг 8.8.8.8 с потерями» = «Да» THEN «Решение найдено: проблема на последней миле, необходимо вызвать специалиста для уточнения и устранения проблемы»

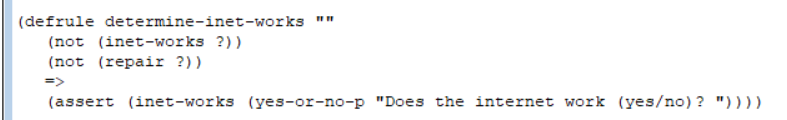
В инструменте разработки ЭС CLIPS реализуем сформированные правила.

Под функцией в CLIPS понимается фрагмент исполняемого кода, с которым связано уникальное имя и который возвращает полезное значение или имеет полезный побочный эффект (например, вывод информации на экран). Существует несколько типов функций. Пользовательские и системные функции - это фрагменты кода, написанные на внешних языках и связанные со средой CLIPS. Системными называются те функции, которые были определены изначально внутри среды CLIPS. Пользовательскими называются функции, которые были определены вне CLIPS. Конструкция deffunction позволяет определять новые функции непосредственно в среде CLIPS с использованием синтаксиса CLIPS. Функции, определенные таким образом, выглядят и работают подобно остальным функциям, однако они выполняются не напрямую, а интерпретируются средой CLIPS.

Для получения необходимой информации от пользователя создадим правила диагностики, которые в зависимости от той или иной ситуации будут задавать пользователю необходимые вопросы и получать ответ в строго заданной форме. Дальнейшая диагностика будет производиться с учетом предыдущих ответов на вопросы, заданные пользователю. Эти ответы будут формировать описание текущей ситуации с помощью фактов, приведенных выше.

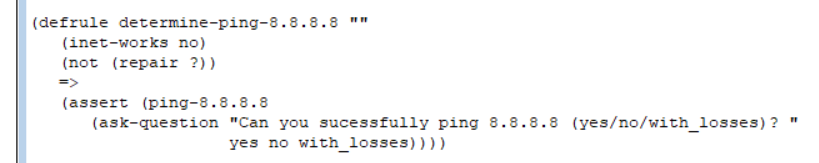
|  |  |
| --- | --- |
|  | Cоздаем функция ask-question с обязательным параметром ?question (вопрос) и групповым параметром $?allowed-values (список допустимых ответов). Функция ask-question задает пользователю вопрос, полученный в переменной ?question, и получает от пользователя ответ, принадлежащий списку допустимых ответов, заданному в $?allowed-values После чего прописывается вывод вопроса и ввод ответа. Далее если введено строковое значение, то символы строки преобразуем в строчные. Используя конструкцию цикла, пока пользователь не введет ответ, принадлежащий списку допустимых значений, повторять ввод.  Будет также очень полезно определить логическую функцию yes-or-no-p, задающую пользователю вопрос и допускающую ответ в виде да/нет. С учетом реализации функции ask-question эта функция примет следующий вид. Функция yes-or-no-p задает пользователю вопрос, полученный в переменной ?question, и получает от пользователя ответ yes(у) или no(n). В случае положительного ответа функция возвращает значение TRUE, иначе - FALSE |

Далее необходимо прописать правила. Первым реализуем правило, определяющее общее состояние работоспособности Интернета.

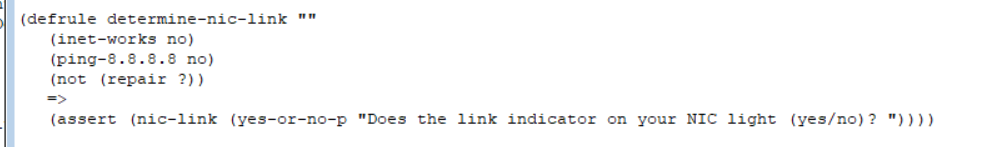


Условный элемент (not (inet-works ?)) гарантирует, что общее состояние работы интернета еще не определено. Если это так, то пользователю задаются соответствующие вопросы и в систему добавляется факт, описывающий текущее общее состояние.

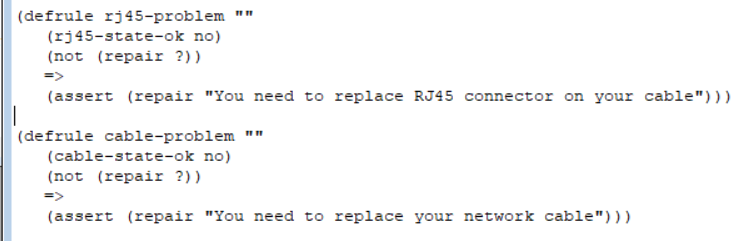
Теперь реализуем правило пингуется ли 8.8.8.8 при неработающем интернете. В данном правиле уже реализуется функция, что ответ может быть не только «да/нет».



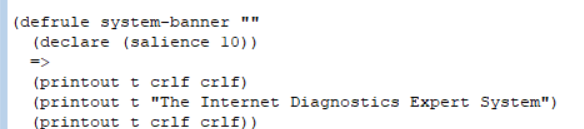
Аналогично, оформляем остальные правила.



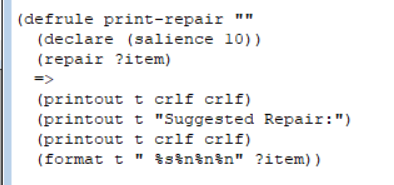
Далее пропишем необходимые результирующие правила. В соответствии с записанными ответами на правила, правило будет выводить результат.



Экспертная система фактически готова к работе. Единственное, чего ей не хватает, — это метода вывода итоговой информации и правила, сообщающего пользователю о начале работы.



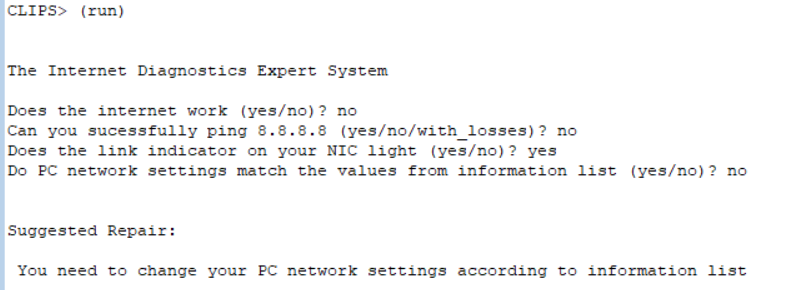
Правило print-repair выводит на экран диагностическое сообщение по устранению найденной неисправности.



Полный листинг программы выглядит следующим образом:

|  |
| --- |
| ;;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ;;\* DEFFUNCTIONS \*  ;;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  (deffunction ask-question (?question $?allowed-values)  (printout t ?question)  (bind ?answer (read))  (if (lexemep ?answer)  then (bind ?answer (lowcase ?answer)))  (while (not (member ?answer ?allowed-values)) do  (printout t ?question)  (bind ?answer (read))  (if (lexemep ?answer)  then (bind ?answer (lowcase ?answer))))  ?answer)  (deffunction yes-or-no-p (?question)  (bind ?response (ask-question ?question yes no y n))  (if (or (eq ?response yes) (eq ?response y))  then yes  else no))  ;;;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ;;;\* QUERY RULES \*  ;;;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  (defrule determine-inet-works ""  (not (inet-works ?))  (not (repair ?))  =>  (assert (inet-works (yes-or-no-p "Does the internet work (yes/no)? "))))    (defrule determine-ping-8.8.8.8 ""  (inet-works no)  (not (repair ?))  =>  (assert (ping-8.8.8.8  (ask-question "Can you sucessfully ping 8.8.8.8 (yes/no/with\_losses)? "  yes no with\_losses))))  (defrule determine-nic-link ""  (inet-works no)  (ping-8.8.8.8 no)  (not (repair ?))  =>  (assert (nic-link (yes-or-no-p "Does the link indicator on your NIC light (yes/no)? "))))    (defrule determine-cable-state ""  (inet-works no)  (ping-8.8.8.8 no)  (not (repair ?))  (nic-link no)  =>  (assert (cable-state-ok (yes-or-no-p "Is the cable OK (yes/no)? "))))    (defrule determine-rj45-state ""  (inet-works no)  (ping-8.8.8.8 no)  (not (repair ?))  (nic-link no)  =>  (assert (rj45-state-ok (yes-or-no-p "Is the RJ45 connector OK (yes/no)? "))))  (defrule determine-net-settings ""  (inet-works no)  (ping-8.8.8.8 no)  (not (repair ?))  (nic-link yes)  =>  (assert (net-settings-ok (yes-or-no-p "Do PC network settings match the values from information list (yes/no)? "))))  (defrule determine-ping-ya.ru ""  (inet-works no)  (ping-8.8.8.8 yes)  (not (repair ?))  =>  (assert (ping-ya.ru (yes-or-no-p "Can you sucessfully ping ya.ru (yes/no)? "))))  (defrule determine-dns-settings ""  (inet-works no)  (ping-8.8.8.8 yes)  (ping-ya.ru no)  (not (repair ?))  =>  (assert (dns-settings-ok (yes-or-no-p "Are PC DNS settings correct (yes/no)? "))))  (defrule determine-slow-web ""  (inet-works yes)  (not (repair ?))  =>  (assert (slow-web (yes-or-no-p "Do internet pages open slowly (yes/no)? "))))  (defrule determine-slow-speed ""  (inet-works yes)  (slow-web yes)  (not (repair ?))  =>  (assert (slow-speed (yes-or-no-p "Please measure your internet speed. Is it much slower than is specified by your tariff (yes/no)? "))))  (defrule determine-losses ""  (inet-works yes)  (slow-web yes)  (slow-speed yes)  (not (repair ?))  =>  (assert (losses-present (yes-or-no-p "Please ping 8.8.8.8. Do you have losses in transmission (yes/no)? "))))      ;;;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ;;;\* REPAIR RULES \*  ;;;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  (defrule rj45-problem ""  (rj45-state-ok no)  (not (repair ?))  =>  (assert (repair "You need to replace RJ45 connector on your cable")))  (defrule cable-problem ""  (cable-state-ok no)  (not (repair ?))  =>  (assert (repair "You need to replace your network cable")))  (defrule pc-settings-problem ""  (net-settings-ok no)  (not (repair ?))  =>  (assert (repair "You need to change your PC network settings according to information list")))  (defrule isp-problem ""  (dns-settings-ok yes)  (not (repair ?))  =>  (assert (repair "There is a problem on the ISP side")))    (defrule pc-dns-problem ""  (dns-settings-ok no)  (not (repair ?))  =>  (assert (repair "You need to change your PC DNS settings according to information list")))  (defrule no-problem ""  (slow-web no)  (not (repair ?))  =>  (assert (repair "Your Internet connection is fine")))  (defrule pc-virus-problem ""  (losses-present no)  (not (repair ?))  =>  (assert (repair "There is a problem with your PC. Please, check your system with an antivirus and disable any other software that may consume Internet bandwidth")))  (defrule browser-problem ""  (or (slow-speed no)  (ping-ya.ru yes))  (not (repair ?))  =>  (assert (repair "There is a problem with your browser. Please reinstall it or try another one")))  (defrule last-mile-problem ""  (or (and (cable-state-ok yes) (rj45-state-ok yes))  (net-settings-ok yes)  (ping-8.8.8.8 with\_losses)  (losses-present yes)  )  =>  (assert (repair "There is a problem on a last mile. You need to call a technician to determine exact point of failure and fix it")))    ;;;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ;;;\* STARTUP AND CONCLUSION RULES \*  ;;;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  (defrule system-banner ""  (declare (salience 10))  =>  (printout t crlf crlf)  (printout t "The Internet Diagnostics Expert System")  (printout t crlf crlf))  (defrule print-repair ""  (declare (salience 10))  (repair ?item)  =>  (printout t crlf crlf)  (printout t "Suggested Repair:")  (printout t crlf crlf)  (format t " %s%n%n%n" ?item)) |

Протестируем работу программу. Загрузим скрипт и запустим выполнение.



**Вывод:** Недостатком языка продукционных правил можно считать отсутствие явных связей между правилами и целями, к достижению которых необходимо стремится. Таким образом, для активизации одного из продукционных правил необходимо проверка всей продукционной базы знаний, что при больших объемах информации приводит к существенным затратам временных и технических ресурсов интеллектуальной системы. Возможность решения этой проблемы заключается в разработке перспективных продукционных баз знаний. Отличительной чертой и основным преимуществом продукционной базы знаний является простота анализа, дополнения, модификации и аннулирования определенных продукционных правил.

# Лабораторная работа №2 «Проектирование и реализация оболочки экспертной системы, использующей продукционный способ представления знаний»

**Цель работы:** сформировать у студентов способность построения учебного прототипа экспертной системы, основанной на знаниях, включающего основные компоненты ИИС.

**Задание:**

1. Взять предметную область и БЗ из лабораторной работы №1.
2. В текстовом виде описать БЗ в выбранной нотации.
3. Спроектировать интерфейс модулей реализуемой оболочки ЭС.
4. Начать разработку в системе хранения контроля версий.

**Выполнение работы:**

Экспертная система - это программное средство, использующее экспертные знания для обеспечения высокоэффективного решения неформализованных задач в узкой предметной области. Основу ЭС составляет база знаний (БЗ) о предметной области, которая накапливается в процессе построения и эксплуатации ЭС.

Главным отличием ЭС и систем искусственного интеллекта от систем обработки данных состоит в том, что в них используется символьный, а не числовой способ представления данных, а в качестве методов обработки информации применяются процедуры логического вывода и эвристического поиска решений.

В системе, основанной на правилах, машина логического вывода (МЛВ) определяет, какие антецеденты правил выполняются согласно фактам.

В качестве предметной области для проектирования и реализации оболочки ЭС, использующей продукционный способ представления знаний выбрана из ЛР№1, а именно область диагностики работы Интернета (выявления проблем работы Интернета и их решение).

Код программы, использующей МЛВ, на Python:

|  |
| --- |
| import json  class Facts\_Bank:  ops = { #  'and': [2, lambda x, y: x and y],  'or': [2, lambda x, y: x or y],  'not': [1, lambda x: not x]  }  def \_\_init\_\_(self):  self.working\_space = {} # fact:parameter  self.\_\_last\_fact = None  def is\_true(self, antecedent):  stack = []  for elem in antecedent:  if elem['fact'] in Facts\_Bank.ops:  op = Facts\_Bank.ops[elem['fact']]  stack.append(op[1](\*(stack.pop() for i in range(op[0]))))  else:  fact\_true\_false = elem['fact'] in self.working\_space and \  (not elem['parameter']['required'] or elem['parameter']['required']  and elem['parameter']['value'] == self.working\_space[elem['fact']])  stack.append(fact\_true\_false)  return stack.pop()  def add\_fact(self, qonsequent):  values = list(qonsequent['value'])  if len(values) > 1:  value = self.ask\_question(qonsequent['question'], values)  else:  value = values[0]  fact = qonsequent['fact']  self.working\_space[fact] = value  self.\_\_last\_fact = (fact,value)  def ask\_question(self, question, answers):  answer = ''  while not answer in answers:  answer = input('{} ({}) '.format(question, ','.join(answers)))  return answer  def fact\_present(self, fact\_name):  return fact\_name in self.working\_space  @property  def last\_fact(self):  return f'{self.\_\_last\_fact[0]} : {self.\_\_last\_fact[1]}'  def print\_hi(name):  # Use a breakpoint in the code line below to debug your script.  print(f'Hi, {name}') # Press Ctrl+F8 to toggle the breakpoint.  # Press the green button in the gutter to run the script.  if \_\_name\_\_ == '\_\_main\_\_':  with open('kb.json', 'r') as kb\_file:  rules = json.load(kb\_file)['rules']  fb = Facts\_Bank()  used\_rules=[]  while True: # Цикл для обработки  accepted\_rules = []  for rule in rules:  if fb.is\_true(rules[rule]['antecedent']):  accepted\_rules.append(rule)  if not accepted\_rules:  break  active\_rule = max(accepted\_rules, key=lambda x: not (x in used\_rules))  active\_fact = rules[active\_rule]['qonsequent']  if fb.fact\_present(active\_fact['fact']):  break  fb.add\_fact(active\_fact)  used\_rules.append(active\_rule)  print(fb.last\_fact) |

Правила по аналогии со скриптом CLIPS записаны в файле JSON в следующем виде:

"determine-nic-link" : {

"antecedent" : [

{

"fact" : "repair",

"parameter" : {

"required" : false,

"value" : null

}

},

{

"fact" : "not"

},

{

"fact" : "inet-works",

"parameter" : {

"required" : true,

"value" : "no"

}

},

{

"fact" : "and"

},

{

"fact" : "determine-ping-8",

"parameter" : {

"required" : true,

"value" : "no"

}

},

{

"fact" : "and"

}

],

"qonsequent" : {

"fact" : "determine-nic-link",

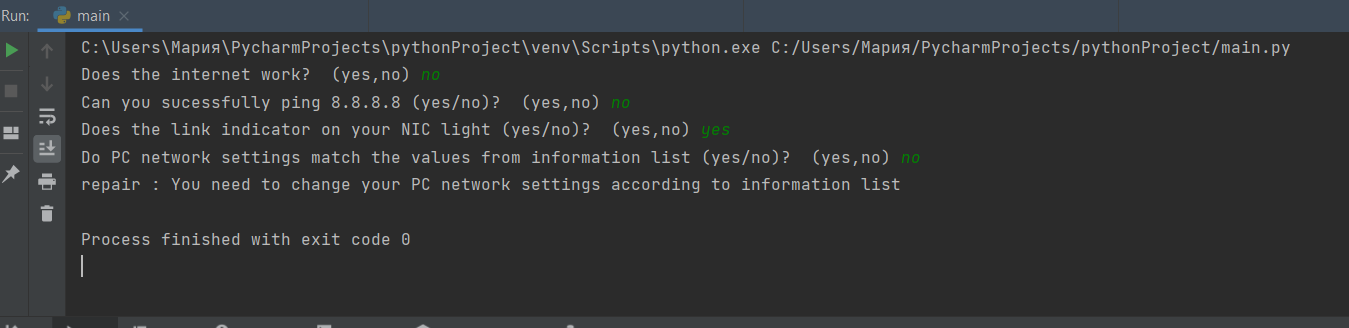
"question" : "Does the link indicator on your NIC light (yes/no)? ",

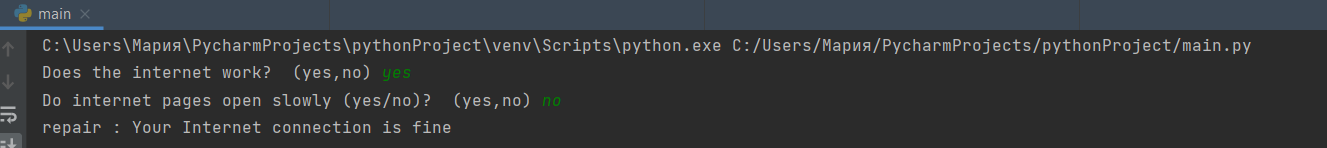
"value" : ["yes", "no"]

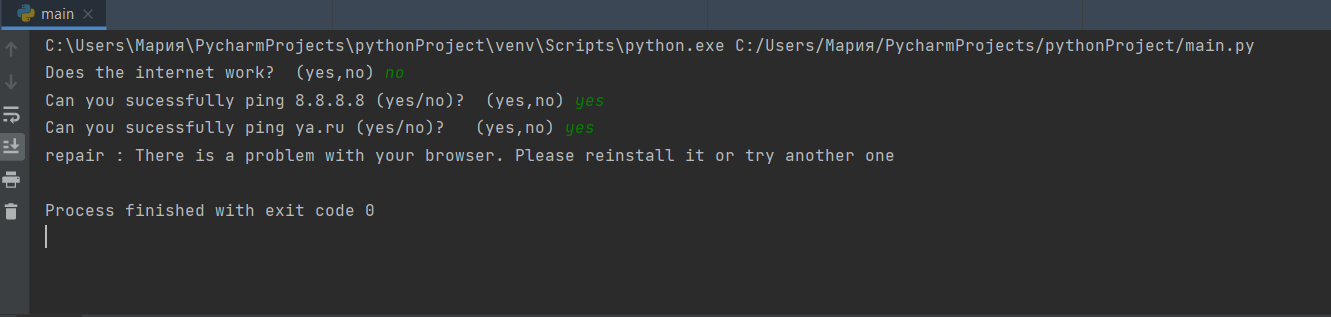
}

},

Результаты выполнения программы выглядит следующим образом:







**Вывод:** в результате выполнения работы была выполнена разработка прототипа экспертной системы, позволяющей осуществлять логический вывод. Основу ЭС составляет база знаний о предметной области, которая накапливается в процессе построения и эксплуатации ЭС.

# Лабораторная работа №3 «Проектирование базы знаний и реализация ЭС на основе семантических сетей и фреймов»

**Цель работы:** сформировать у студентов способность построения прототипа ЭС, использующей различные модели представления знаний.

**Задание:**

1. Взять предметную область и БЗ из лабораторной работы №1.
2. Спроектировать БЗ.
3. Начать разработку в системе контроля версий.

**Выполнение работы:**

Структура семантической сети отображается графически с помощью узлов и соединяющих их дуг. Узлы иногда именуются объектами, а дуги-связями, или ребрами. Связи в семантической сети применяются для представления отношений, а узлы, как правило, служат для представления физических объектов, концепций или ситуаций.

Наиболее распространенные иерархические связи: IS-A, A-KINF-OF (AKO). Связь IS–A означает «является экземпляром» данного класса и указывает на конкретный экземпляр некоторого класса. В данном случае связь AKO используется для обозначения отношения одного класса с другим, PART-OF (является частью).

В качестве предметной области для проектирования базы знаний и реализации ЭС на основе семантических сетей выбрана из ЛР№1, а именно область диагностики работы Интернета (выявления проблем работы Интернета и их решение). Спроектированная БЗ представлена в ЛР№1.

****

Типы запросов определяются типами связей между вершинами. Составим несколько запрос в данной семантической сети.

1. Является ли «Отсутствие пинга до 8.8.8.8» «неисправностью»?
2. Какие разновидности «Симптомов» проблем с Интерном существуют?
3. Что включает в себя «Сеть клиента»?
4. На что могут оказывать влияние «Неисправности» в сети?

**Вывод:** Семантическая сеть — это система знаний, имеющая определенный смысл в виде целостного образа сети, узлы которой соответствуют понятиям и объектам, а дуги — отношениям между понятиями и объектами. Сама по себе семантическая сеть является моделью памяти и не раскрывает, каким образом осуществляется представления знаний. Поэтому семантические сети должны рассматриваться как метод представления знаний с возможностями структурирования этих знаний, процедурами их использования и механизмом вывода.

# Лабораторная работа №4 «Реализация моделей принятия коллективных решений»

**Цель работы:** сформировать у студентов способность построения экспертной системы поддержки принятия коллективных решений.

**Задание:**

1. Изучить модели принятия коллективных решений: модель Кондорсе (явный победитель, правило Копленда, правило Симпсона), модель Борда.

**Выполнение работы:**

Правила голосования решает задачу принятия коллективного решения: несколько агентов (выборщиков) осуществляют набор из нескольких исходов (кандидатов). Проблема возникает при количестве кандидатов больше двух.

Рассмотрим наиболее популярные модели:

1. Модель Кондорсе. Победителем по Кондорсе называется кандидат a, который побеждает любого другого кандидата при попарном сравнении по правилу большинства (т.е. число выборщиков, считающих, что a лучше b, больше, чем число выборщиков, считающих, что b лучше a). Однако победитель по Кондорсе может не существовать.

Профиль голосования с парадоксом по Кондорсе

|  |  |  |
| --- | --- | --- |
| 8 | 7 | 6 |
| a | b | c |
| b | c | a |
| c | a | b |

Для определения победителя может применяться одно из возможных правил:

* Правило Копленда. Сравним кандидата a с любым другим кандидатом x. Начислим ему +1, если для большинства a лучше x, −1, если для большинства x лучше a, и 0 при равенстве. Суммируя общее количество очков по всем x, x 6= a, получаем оценку Копленда для a. Избирается кандидат с наивысшей из таких оценок.
* Правило Симпсона. Рассмотрим кандидата a, любого другого кандидата x и обозначим через N(a, x) число выборщиков, для которых a лучше x. Оценкой Симпсона для a называется минимальное из чисел N(a, x) по всем x, x 6= a. Избирается кандидат с наивысшей из таких оценок.

2. Модель Борда. Каждый выборщик объявляет свои предпочтения, ранжируя p кандидатов от лучшего к худшему (безразличие запрещается). Кандидат не получает очков за последнее место, получает p − 1 очков за первое место. Побеждает кандидат с наибольшей суммой очков.

**Вывод:** Одной из простейших и наиболее часто встречающихся на практике является модель Кондорсе. В последнее время широкое распространение получила модель Борда. Данные модели не учитывают многокритериальность выбора при сравнении альтернатив. Другими словами, не учитываются предпочтения экспертов в рамках каждой альтернативы, определяемой набором частных критериев оптимальности. Кроме того, не учитывается важность (квалификация) экспертов при многокритериальной оценке каждой альтернативы. Преимуществом модели Борда является то, что всегда существует наилучшая альтернатива (или несколько наилучших альтернатив, если они набирают одинаковое количество баллов). При этом, если решение по модели Кондорсе существует, то оно, как правило, совпадает с решением по модели Борда.

# Лабораторная работа №5 «Реализация нейронных сетей для решения задачи распознавания образов»

**Цель работы:** сформировать у студентов способность построения нейросетевой интеллектуальной системы для решения задач распознавания образов.

**Задание:**

1. Реализация приложения, решающая задачу распознавания с использованием искусственных нейронных сетей.

**Выполнение работы:**

Традиционным подходом к построению механизмов рассуждения является использование дедуктивного логического вывода на правилах (rule-based reasoning), который применяется в экспертных системах продукционного и логического типа. При таком подходе необходимо заранее сформулировать весь набор закономерностей, описывающих предметную область. Альтернативный подход основан на концепции обучения по примерам (casebased reasoning). В этом случае при построении интеллектуальной системы не требуется заранее знать обо всех закономерностях исследуемой области, но необходимо располагать достаточным количеством примеров для настройки разрабатываемой адаптивной системы, которая после обучения будет способна получать требуемые результаты с определенной степенью достоверности.

Серьезное развитие нейрокибернетика получила в трудах американского нейрофизиолога Ф. Розенблатта, который предложил свою модель нейронной сети в 1958 году и продемонстрировал созданное на ее основе электронное устройство, названное персептроном. Ф. Розенблатт ввел возможность модификации межнейронных связей, что сделало ИНС обучаемой. Первые персептроны были способны распознавать некоторые буквы латинского алфавита. Впоследствии модель персептрона была значительно усовершенствована, а наиболее удачным ее применением стали задачи автоматической классификации.

Алгоритм обучения персептрона включает следующие шаги:

1. Системе предъявляется эталонный образ.
2. Если результат распознавания совпадает с заданным, весовые коэффициенты связей не изменяются.
3. Если ИНС неправильно распознает результат, то весовым коэффициентам дается приращение в сторону повышения качества распознавания.

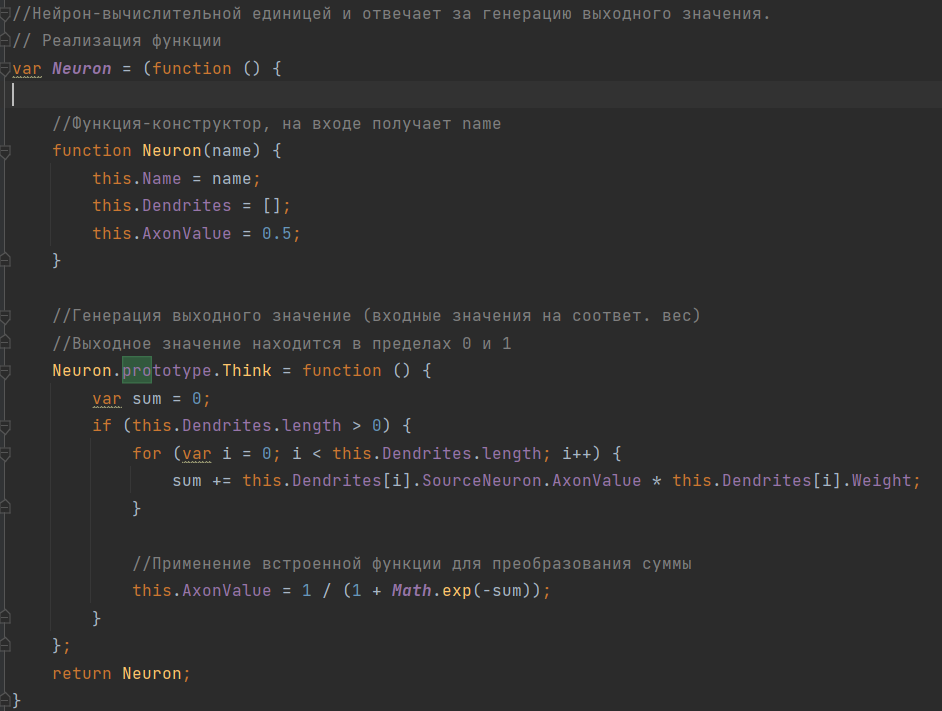
Нейронные сети нашли широкое применение в различных направлениях. Существуют архитектуры нейронных сетей, решающие задачу распознавания изображений наиболее эффективно, например, сверточные нейронные сети.

Для того, чтобы передать изображение в искусственную нейронную сеть (ИНС), нужно преобразовать его в матрицу размером NxN, где каждый элемент матрицы имеет значение 1 или 0, исходя из контуров распознавания символа. После чего каждый пиксель изображения передается на вход сети. Для того, чтобы ИНСТ могла распознавать образы, ее нужно соответствующим образом обучить.

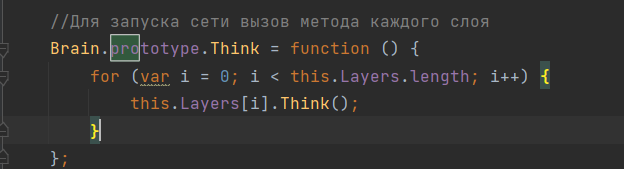
Для реализации нейронной сети можно использовать принципы объектно-ориентированного программирования и выделить следующие классы:

1. Нейрон;
2. Слой;
3. Синапс-связь между слоями.

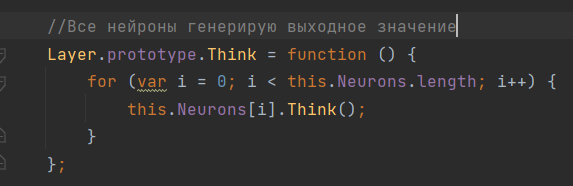
Отобразим основные моменты. Код класса Neuron.js



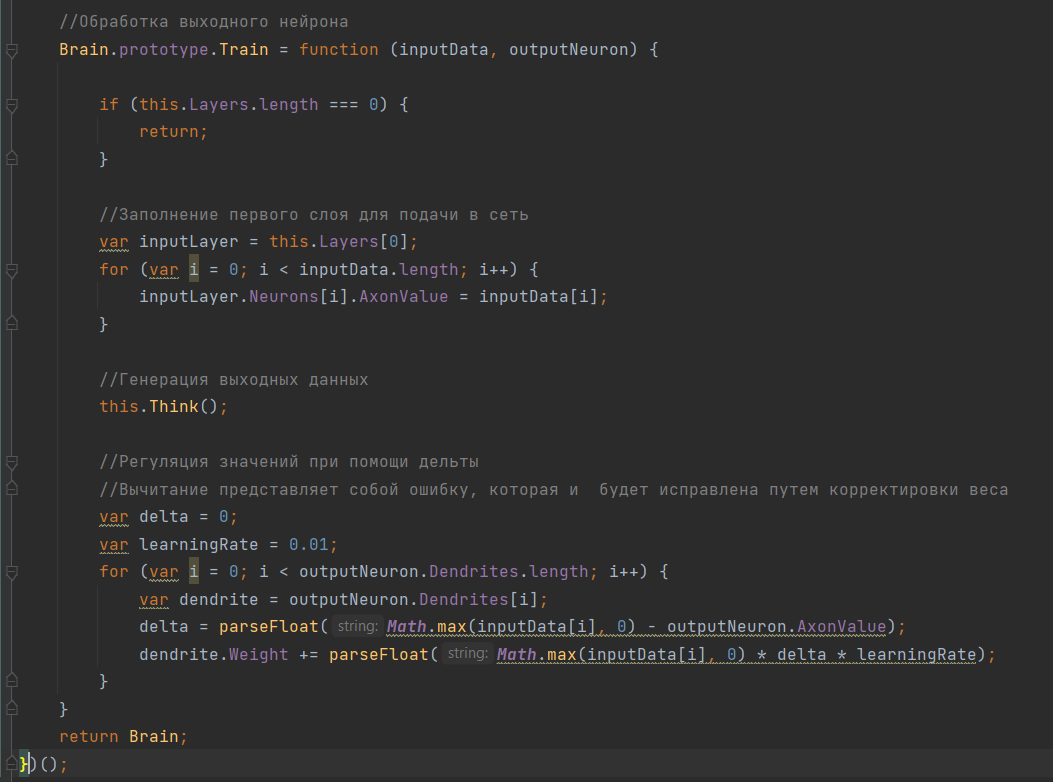
Для запуска сети, необходимо вызвать метод Think() каждого слоя.



Затем каждый слой вызывает метод Think() нейрона.



После чего необходимо только прописать, каким образом будет происходить обучение сети. Как и человеческий мозг, ИНС сначала должна хорошо понимать, что считается, например, символом «А». Поэтому пользователю нужно сначала нарисовать на холсте несколько хороших букв «А». В цикле обучения сетевая сеть будет работать как обычно, но вместо того, чтобы показывать пользователю выходные данные, она вычисляет разницу (дельту) между выходным значением и входным значением. Дельта или значение «ошибки» указывает, насколько «неправильный» вес дендрита. Чтобы скорректировать вес, он будет увеличен на произведение значения ошибки, входного значения и постоянной скорости обучения.



**Вывод:** Простейшей формой искусственных нейронных сетей является так называемая нейронная сеть с прямой связью. Нейроны в такой сети организованы слоями. Нейронная сеть может иметь много слоев, в зависимости от требований. В многослойных (слоистых) нейронных сетях нейроны объединяются в слои. Слой содержит совокупность нейронов с едиными входными сигналами. Число нейронов в слое может быть любым и не зависит от количества нейронов в других слоях. В общем случае сеть состоит из слоев, пронумерованных слева направо. Внешние входные сигналы подаются на входы нейронов входного слоя, а выходами сети являются выходные сигналы последнего слоя.