ЛАБОРАТОРНАЯ РАБОТА №3

«РАЗРАБОТКА И ИССЛЕДОВАНИЕ ЭКСПЕРТНОЙ СИСТЕМЫ»

1. Цель работы

Разработка экспертной системы продукционного типа на Прологе, исследование базовых принципов организации экспертных систем.

2. Постановка задачи

2.1. Изучить модели представления знаний по лекционному материалу и учебным пособиям [1,5,8 ].

2.2. Детально изучить организацию продукционной системы, механизмы прямого и обратного вывода, основы построения подсистемы объяснения, примеры реализации ЭС на Прологе [1,2,4,11].

2.3. Выполнить анализ предметной области ЭС в соответствии с вариантом задания:

a) выбрать задачу, решаемую ЭС, и определить цели системы: конечные, промежуточные и вспомогательные;

б) выделить подзадачи, которые следует решить для достижения цели;

в) разработать продукционную базу правил для решения выделенных подзадач.

2.4. Ознакомиться с примерами программных кодов, приведенных в приложении В и по аналогии разработать ЭС продукционного типа для решения задачи в заданной предметной области.

2.5. Создать в среде программирования Пролог проект ЭС и выполнить его отладку.

2.6. Исследовать свойства разработанной системы. Получить протоколы работы системы при доказательстве различных целевых утверждений.

2.7. Зафиксировать результаты работы программы в виде экранных копий.

3. Вариант задания

Реализовать продукционную экспертную систему в соответствии с номером варианта, указанном в таблице 1. При этом количество рассматриваемых объектов предметной области должно быть не менее 10 и характеризующих их атрибутов также — не менее 10. Система должна уметь давать объяснения вывода. Задачу, решаемую ЭС, выбрать самостоятельно с учетом перечня задач, указанного в п.3.2.1, или с помощью каталога ЭС.

Таблица 1 – Предметная область экспертной системы варианта 19

|  |  |
| --- | --- |
| Вариант | Предметная область |
| 9, 24 | Компьютерные игры |

4. Ход работы

Листинг 1 – Код файла db.pl

info:-

nl,

write('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'),nl,

write('\* Expert system \*'),nl,

write('\* Computer games \*'),nl,

write('\* \*'),nl,

write('\*----------------------------\*'),nl,

write('\* Answer on questions : \*'),nl,

write('\* yes, no, why \*'),nl,

write('\* \*'),nl,

write('\* \*'),nl,

write('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'),nl,

write('Write any symbol'),nl,

get0(\_).

% база правил

rule1 :: if [attack(shoot), online] then shooter.

rule2 :: if [attack(shoot), attack(beat)] then violence.

rule3 :: if [open\_world, plot, transport] then adventure.

rule4 :: if [animals, transport, simulator, creater(maxis)] then sims.

rule5 :: if [animals, simulator, creater(maxis)] then spore.

rule6 :: if [shooter, attack(beat)] then csgo.

rule7 :: if [shooter] then tanki-online.

rule8 :: if [open\_world, animals, survival] then minecraft.

rule9 :: if [creater(cd\_project\_red), adventure] then witcher.

rule10 :: if [creater(cd\_project\_red), adventure, violence] then cyberpunk.

rule11 :: if [creater(rockstar), adventure, animals, violence] then rdr.

rule12 :: if [creater(rockstar), adventure, violence] then gta.

rule13 :: if [online, open\_world, animals, survival] then ark.

rule14 :: if [rpg, adventure] then skyrim.

rule15 :: if [plot] then detroit.

% гипотезы

h1 :: hypothesis(cyberpunk).

h2 :: hypothesis(gta).

h3 :: hypothesis(rdr).

h4 :: hypothesis(sims).

h5 :: hypothesis(tanki-online).

h6 :: hypothesis(csgo).

h7 :: hypothesis(skyrim).

h8 :: hypothesis(witcher).

h9 :: hypothesis(detroit).

h10 :: hypothesis(spore).

h11 :: hypothesis(minecraft).

h12 :: hypothesis(ark).

% признаки

q1 :: attribute(attack(shoot)).

q2 :: attribute(attack(beat)).

q5 :: attribute(simulator).

q6 :: attribute(survival).

q7 :: attribute(open\_world).

q8 :: attribute(online).

q9 :: attribute(animals).

q10 :: attribute(transport).

q11 :: attribute(plot).

q12 :: attribute(creater(cd\_project\_red)).

q13 :: attribute(creater(rockstar)).

q14 :: attribute(rpg).

q15 :: attribute(creater(maxis)).

Листинг 2 – Код файла laba3.pl

:-dynamic

reported/2.

operators:-

op(950, xfx, then),

op(960, fx, if),

op(970, xfx, '::').

:-operators.

find1(H, Steck, reported(H)):-reported(H, yes).

find1(H, Steck, reported(H)):-requested(H), not(reported(H,\_)), ask(H, Steck).

find1(H, Steck, Fact :: H):-Fact :: H.

find1(H, Steck, Rule :: if D1 then H):-

Rule :: if H1 then H,

find(H1, [Rule | Steck], D1).

find([], Steck, Tree):-Tree=[].

find([H1|T], Steck, [Tree1 | Tree]):-

find1(H1, Steck, Tree1), find(T, Steck, Tree).

requested(H):-Fact :: attribute(H).

ask(H, Steck):-write(H), write('?'), nl, read(O), answer(H, O, Steck).

answer(H, yes, Steck):-assert(reported(H, yes)),!.

answer(H, no, Steck):-assert(reported(H, no)),!, fail.

answer(H, why, []):-!,write(' You are asking too many questions'), nl, ask(H,[]).

answer(P, why, [H]):- !,write(' my hypothesis: '), write(H), nl, ask(P,[]).

answer(H, why, [Rule | Steck]):-!,

Rule :: if H1 then H2,

write(' trying to prove'),

write(H2), nl,

write(' with help of rule: '),

write(Rule), nl,

ask(H, Steck).

how(H, Tree):-how1(H,Tree), !.

how(H,\_):- write(H), tab(2), write(' not proved'), nl.

how1(H,\_):- reported(H, \_),!,

write(H),write(' was vvedeno '), nl.

how1(H, Fact :: H):-!,

write(H),write(' is the fact '), write(Fact), nl.

how1(H, [Rule :: if \_ then H]):-!,

write(H), write(' was proved with the help of rule'), nl,

Rule :: if H1 then H,

show\_rule(Rule :: if H1 then H).

how1(H, [Rule :: if Tree then \_]):-how(H,Tree).

how1(H,[]):-!.

how1(H, [D1|D2]):-how(H,[D1]),!;

how1(H, D2).

show\_rule(Rule :: if H1 then H):-

write(Rule), write(':'), nl,

write('if '), write(H1), nl,

write('then '), write(H), nl.

init:-retractall(reported(\_,\_)).

start:-

reconsult('F:/db.pl'),

info,

go\_exp\_sys.

go\_exp\_sys:- init,

Fact :: hypothesis(H),

find([H], [H], Tree),

write(' solution: '), write(H), nl,

explain(Tree),

back.

explain(Tree):- write(' shall I explain ? [aim/no]: '), nl, read(H),

(H\=no,!,how(H,Tree), explain(Tree));!.

back:- write('Shall we continue to find the solution [yes/no] ?: '), nl, read(no).

На рисунке 1 представлен пример работы разработанной ЭС.

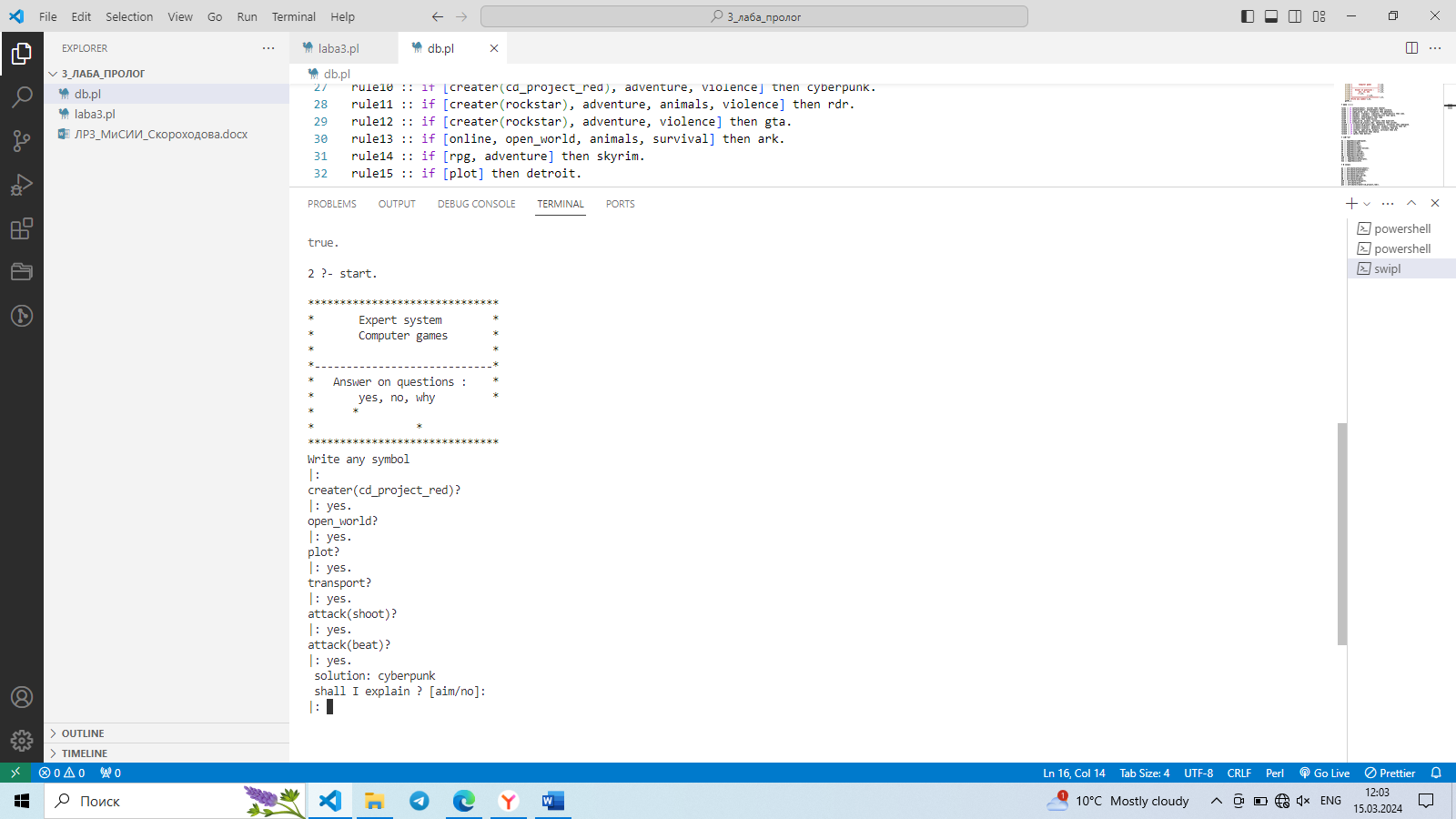


Рисунок 1 – Пример работы ЭС в области игр

4. Выводы

В ходе выполнения лабораторной работы были исследованы базовые принципы организации экспертных систем и разработана экспертная система по заданному варианту. Задача, решаемая разработанной ЭС, - классификация.

По результатам тестирования можно сказать, что при заданных правилах система корректно ищет загаданное решение.

5. Контрольные вопросы

**5.1. Приведите определение экспертной системы.**

Под **экспертной системой** (ЭС) понимают программную систему, аккумулирующую знания эксперта в определенной области и вырабатывающую решения и рекомендации на уровне эксперта.

**5.2. Перечислите типовые задачи, решаемые с помощью ЭС.**

Перечень типовых задач, решаемых ЭС в самых различных областях, включает:

- интерпретацию — извлечение информации из первичных данных (распознавание образов, определение состава вещества и др.);

- диагностику — обнаружение неисправностей и причин их появления в некоторой системе (медицинской, механической, электронной и др.);

- мониторинг — непрерывная интерпретация данных в реальном времени с сигнализацией о выходе тех или иных параметров за допустимые пределы (контроль движения транспорта, наблюдение за состоянием энергетических объектов и др.);

- прогноз — предсказание вероятных последствий на основе прошедших и настоящих событий (предсказание погоды, прогноз ситуаций на финансовых рынках и др.);

- планирование — определение последовательности действий, направленных на достижение заранее поставленных целей (планирование поведения роботов, составление маршрутов движения транспорта и др.);

- проектирование — определение конфигурации системы при заданных ограничениях (синтез электронных схем, оптимальное размещение объектов в ограниченном пространстве и др.);

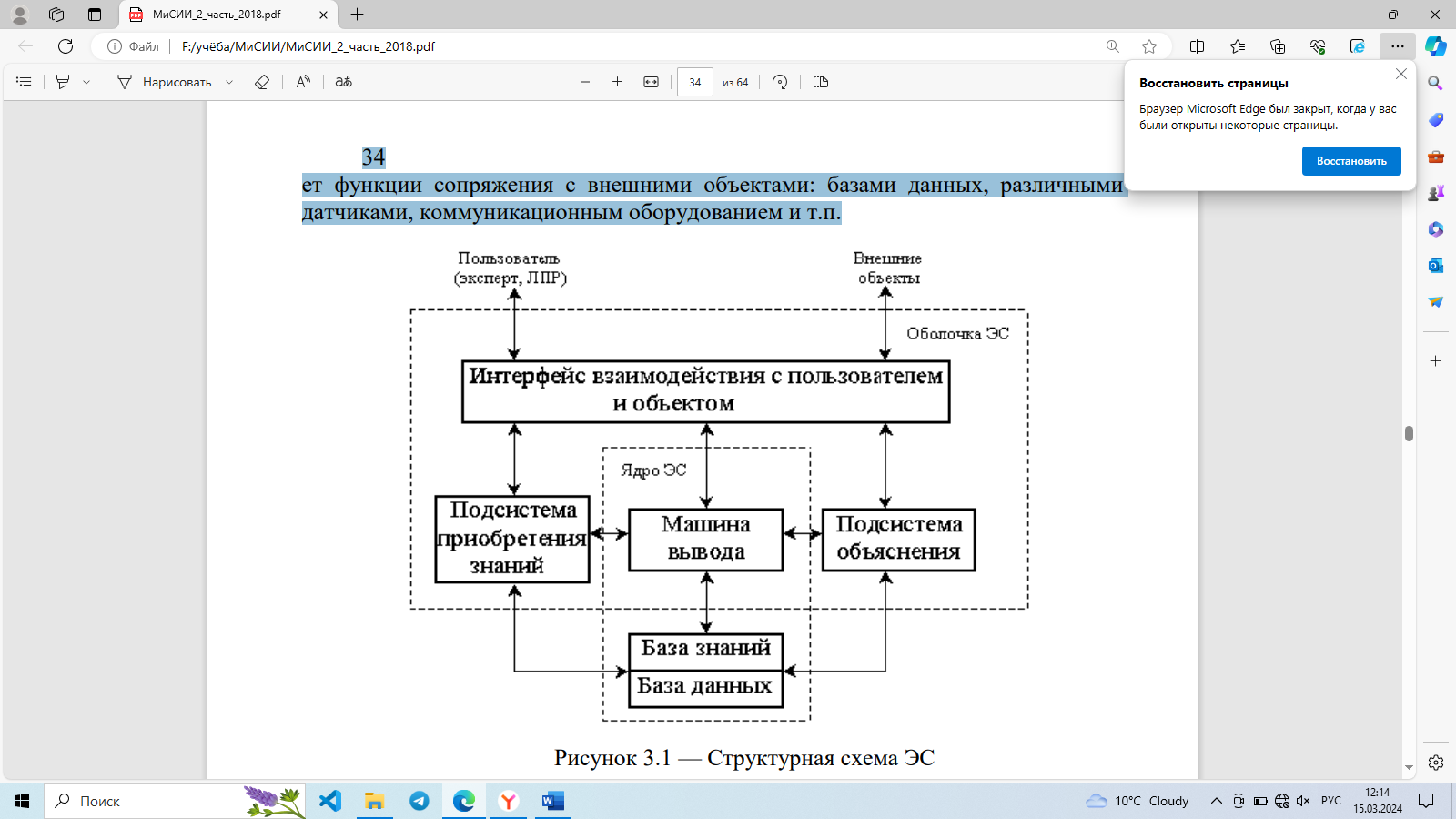
- отладку и ремонт — выполнение последовательности действий по приведению той или иной системы к требуемым режимам функционирования (помощь при отладке программного обеспечения, ремонт инженерных коммуникаций и др.);

- обучение — интерпретация, диагностика и коррекция знаний и умений обучаемого;

- управление — формирование управляющих воздействий, определяющих поведение сложных систем (управление воздушным транспортом, военными действиями, деловой активностью в сфере бизнеса и др.).

**5.3. Назовите основные компоненты ЭС и объясните их функции.**

Типовая архитектура ЭС изображена на рисунке 3.1. Взаимодействие с ЭС осуществляется с помощью интерфейса. Кроме обеспечения взаимодействия с различными категориями пользователей, интерфейс (если необходимо) выполняет функции сопряжения с внешними объектами: базами данных, различными датчиками, коммуникационным оборудованием и т.п.



**Ядро ЭС** образует база данных, база знаний и машина вывода. База данных представляет собой рабочую память, в которой хранятся текущие данные, заключения и другая информация, имеющая отношение к анализируемой системой ситуации. База знаний обеспечивает хранение знаний, представленных с помощью одной из моделей: логической, продукционной, фреймовой, сетевой и др.

**Машина вывода** (подсистема поиска решений), используя данные и знания, организует управление выводом в соответствии с используемой моделью представления знаний. Целью вывода является получение заключений, согласующихся с той информацией, которая содержится в базе знаний.

**Подсистема объяснения** ЭС позволяет пользователю выяснить, как система получила решение задачи, и какие знания были при этом использованы.

**Подсистема приобретения знаний** используется как с целью автоматизации процесса наполнения ЭС знаниями, так и при корректировке базы знаний, при ее обновлении, пополнении или исключении элементов знаний.

**5.4. Назовите основные этапы разработки ЭС, перечислите задачи, решаемые на каждом из этапов.**

**5.5. Объясните термины “приобретение знаний” и “извлечение знаний”.**

**Приобретение знаний (Knowledge Acquisition):**

Приобретение знаний в ЭС означает процесс получения и захвата экспертных знаний от доменных экспертов или других источников. Это включает сбор информации, интервьюирование экспертов, анализ документации и данных, а также формализацию и структурирование полученных знаний для использования в ЭС. Цель приобретения знаний состоит в создании базы знаний, которая будет использоваться ЭС для принятия решений и решения задач в предметной области.

**Извлечение знаний (Knowledge Extraction):**

Извлечение знаний в ЭС относится к процессу получения значимых и полезных знаний из имеющихся данных или информации. Это может включать автоматическое извлечение знаний из текстов, баз данных или других источников информации. В процессе извлечения знаний используются методы обработки естественного языка, статистического анализа, машинного обучения и другие техники для выявления паттернов, закономерностей и связей в данных. Извлеченные знания могут быть преобразованы и добавлены в базу знаний ЭС для улучшения ее функциональности и результатов.

**5.6. Сформулируйте обобщенный алгоритм прямого вывода на правилах-продукциях.**

Прямой вывод начинается с задания исходных данных решаемой задачи, которые фиксируются в виде фактов в рабочей памяти системы. Правила, предпосылки которых сопоставимы с исходными фактами, обеспечивают генерацию новых фактов, добавляемых в рабочую память. Процесс применения правил к новым фактам продолжается, пока не будет получено целевое состояние рабочей памяти

5.7. Сформулируйте обобщенный алгоритм обратного вывода на правилах-продукциях.

5.8. Объясните суть вопросов “почему” и “как”, используемых для формирования объяснений вывода в ЭС.

5.9. Объясните на примере механизм формирования ответа на вопрос типа “как”.

5.10. Объясните на примере механизм формирования ответа на вопрос типа “почему”.

**5.11. Как можно представить базу продукционных правил на языке Пролог?**

Базу продукционных правил можно представить на языке Пролог с использованием фактов и правил. (как в лабе)

**5.12. Приведите пример простейшей реализации ЭС на языке Пролог.**

Простейший способ построения ЭС на Прологе — использование в качестве интерпретатора механизма поиска решений, заложенного вПролог-системе .

/\* структура правила: причина:- неисправность \*/

%правило 1

нагреватель(неисправен):- лампа(светится), плита(холодная).

выключатель(не\_включен):- тока(нет). %правило 2

напряжения(нет):- тока(нет). %правило 3

%правило 4

тока(нет):- плита(холодная), лампа(не\_светится).

%правило 5

лампа(неисправна):- лампа(не\_светится), плита(горячая).

лампа(не\_светится). % признаки неисправности

плита(холодная).

гипотеза (нагреватель(неисправен)). % гипотезы

гипотеза (выключатель(не\_включен)).

гипотеза (напряжения(нет)).

гипотеза (лампа(неисправна)).

найти(Х): - гипотеза (Х), X. % проверка гипотез

? – найти(Х).

Х = выключатель(не\_включен);

Х = напряжения(нет).

5.13. Какие правила положены в основу предиката найти(Н), осуществляющего обратный вывод на множестве продукционных правил?

5.14. Напишите на языке Пролог простейший вариант реализации предиката найти(Н).

5.15. Напишете на языке Пролог фрагмент кода, обеспечивающий вывод вопросов, задаваемых ЭС, и анализ ответов.

5.16. Напишите на языке Пролог фрагмент кода, обрабатывающий вопросы типа «почему».