МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КРЕМЕНЧУЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ МИХАЙЛА ОСТРОГРАДСЬКОГО

МЕТОДИЧНІ ВКАЗІВКИ
ЩОДО ВИКОНАННЯ РОЗРАХУНКОВО-ГРАФІЧНОЇ РОБОТИ
З НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
«ДОСЛІДЖЕННЯ ТА АНАЛІЗ КОМП'ЮТЕРНИХ СИСТЕМ ШТУЧНОГО ІНТЕЛЕКТУ»
ДЛЯ СТУДЕНТІВ ДЕННОЇ ФОРМИ НАВЧАННЯ
ЗІ СПЕЦІАЛЬНОСТІ 123 – «КОМП'ЮТЕРНА ІНЖЕНЕРІЯ»
ОСВІТНЬОГО СТУПЕНЯ «МАГІСТР»

КРЕМЕНЧУК 2018

Методичні вказівки щодо виконання розрахунково-графічної роботи з навчальної дисципліни «Дослідження та аналіз комп'ютерних систем штучного інтелекту» для студентів денної форми навчання зі спеціальності 123 – «Комп'ютерна інженерія» освітнього ступеня «Магістр»

Укладачі: к.т.н., доцент П. П. Костенко, ст. викл. А. Л. Юдіна

Рецензент к. т. н., доц., В. М. Сидоренко

Кафедра комп'ютерних та інформаційних систем

Затверджено методичною радою Кременчуцького національного університету імені Михайла Остроградського
Протокол № ____ від «____» _____ 201_ р.
Голова методичної ради _____ проф. В. В. Костін

ЗМІСТ

ВСТУП

Методичні вказівки складено на підставі робочої навчальної програми з дисципліни «Дослідження та аналіз комп'ютерних систем штучного інтелекту» для студентів денної форми навчання зі спеціальності 123 – «Комп'ютерна інженерія» освітнього ступеня «Магістр»

У результаті вивчення навчальної дисципліни студент повинен:

- **Знати** тенденції розвитку науки і техніки в галузі комп'ютерної інженерії та систем штучного інтелекту; способи представлення знань і методи пошуку, структуру експертних систем і методи їх розробки, аналізу та дослідження, основні методи, що використовуються при створенні, аналізі та дослідженнях штучних нейронних мереж, методи проектування експертних систем та штучних нейронних мереж із застосуванням різноманітних алгоритмічних мов.
- **Вміти** використовувати основні методи представлення і рішення інтелектуальних задач, орієнтуватися в різноманітних методах розробки експертних систем та нейронних мереж, використовувати існуючи методи їх аналізу та дослідження, розробляти алгоритми функціонування експертних систем, розробляти типові процедури штучного інтелекту

Виконання розрахунково-графічної роботи дозволяє поглибити теоретичні знання та закріпити практичні навички, отримані студентами під час вивчення дисципліни «Комп'ютерні системи штучного інтелекту», а також показати вміння студентів самостійно працювати з додатковими матеріалами щодо побудови та використання систем штучного інтелекту.

Розрахункова робота є окремим видом навчальної роботи, що виконується протягом семестру з дня видачі завдання, результати виконання оформлюються у вигляді пояснювальної записки. Робота захищається студентом до початку екзаменаційної сесії. Варіанти завдань індивідуальні для кожного студента, варіант обирається за номером у журналі групи.

1 КОРОТКІ ТЕОРЕТИЧНІ ВІДОМОСТІ

Експертна система (ЕС) – це програма (комплекс програм), що моделює роботу людини – експерта в конкретній предметній галузі, яка є чітко визначеною. Головне призначення ЕС – проведення консультацій у тій предметній галузі, для якої спроектована експертна система.

У складі ЕС виділяють три головні компонента (рис. 1):

1. База знань (БЗ) – центральна частина ЕС. Вона містить сукупність фактів та знань (правил) для виводу інших знань. Інформація з БЗ використовується експертною системою для виведення експертного висновку під час консультації. Зазвичай БЗ розташовують окремо від програми, на диску або іншому носієві.

2. Механізм висновку (МВ). МВ містить описи способів застосування, що містяться в БЗ. Під час консультації МВ запускає ЕС у роботу, виконує правила, визначає прийнятність знайденого рішення і передає результати в СІК.

3. Система інтерфейсу користувача(СІК). СІК є тією частиною ЕС, яка взаємодіє з користувачем. У функції СІК входить: прийом інформації від користувача, передача результатів користувачеві в найбільш зручній для нього формі, пояснення отриманих результатів ЕС (видача довідкової інформації по виведенню результатів).

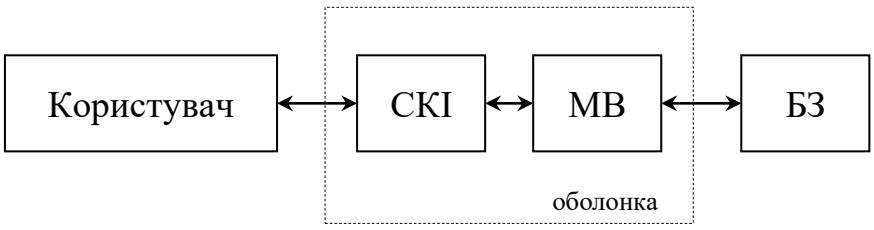

Рисунок 1 – Загальна структура експертної системи

Залежно від способів класифікації і розміщення інформації у Базі знань розрізняють: продукційну, мережну і фреймову моделі представлення знань.

Мережна модель заснована на поданні знань у вигляді мережі, вершини якої відповідають поняттям, а дуги — стосункам між ними.

В основу фреймової моделі покладено логічне угрупування атрибутів об’єкту, при цьому для зберігання і обробки логічні групи описуються у фреймах.

Продукційна модель ґрунтується на правилах виду «якщо – то» і дозволяє поміщати фрагменти фактичного знання в правила мови Пролог. Саме так будуються ЕС, що базуються на правилах. Під час ЕС, що базується на логіці, БЗ надається сукупністю тверджень у вигляді фактів. Виведення експертного висновку у цьому разі будується на підгрунті стандартних засобів роботи зі списками.

Виведення експертного висновку

Під висновком в ЕС розуміється доказ того, що з множини припущень виходить деякий висновок. Прийнята логіка отримання висновку специфікується правилами виводу. Вивід здійснюється за допомогою пошуку і зіставлення за зразком.

У ЕС, що базується на правилах, запити користувача трансформуються у форму, зіставну з формою правил БЗ. Механізм висновку ініціалізує процес зіставлення, починаючи з «верхнього» правила. Звернення до правила називається «викликом». Виклик відповідних правил у процесі зіставлення продовжується доти, поки не відбулося зіставлення або не вичерпана вся БЗ, а значення не знайдене. Якщо МВ виявляє, що можна викликати більш за одне правило, то запускається процес вирішення конфлікту. Під час вирішення конфлікту пріоритет віддається зазвичай тим правилам, які конкретніші, або правилам, які враховують більше поточних даних.

У ЕС, що базується на логіці, трансформовані запити є значеннями, які співставляють із списками значень, що знаходяться у твердженнях БЗ.

Внутрішні програми уніфікації Турбо-Прологу і Visual Prolog'у дозволяють вирішувати задачі пошуку і зіставлення за зразком без написання додаткових правил. Як у системі, що базується на правилах, так і в системі, що базується на логіці, користувач отримує відповіді на свої запити відповідно до закладеної в ЕС логіки. Для програмної реалізації механізму виведення експертного

висновку досить лише написати необхідні специфікації.

Експертна система, що базується на правилах, і її реалізація.

ЕС, що базується на правилах, дозволяє проектувальникові будувати правила, які природним чином об'єднують у групи пов'язані фрагменти знань. Взаємна незалежність продукційних правил робить базу правил семантично модульною і здібною до розвитку.

Реалізована на Турбо-Пролозі (або Visual Prolog'i) ЕС на правилах містить множину правил, які викликаються за допомогою вхідних даних у момент зіставлення. Разом з множиною правил МВ, ЕС має у своєму складі інтерпретатор, який вибирає і активізує різні модулі системи. Робота цього інтерпретатора описується послідовністю трьох кроків :

1. Інтерпретатор співставляє зразок правила з елементами даних в БЗ.
2. Якщо можна викликати більш за одне правило, то для вибору правила використовується механізм вирішення конфлікту.
3. Вибране правило застосовується для пошуку відповіді на поставлене питання.

Цей трикроковий процес є циклічним і називається циклом «розпізнавання-дія». Ствердна відповідь ЕС є результатом виконання одного з продукційних правил, вибір правила проводиться| відповідно до вхідних даних.

Написання на Турбо-Пролозі (або Visual| Prolog'i) ЕС, що базується на правилах, починається з декларації БД. БД зберігає відповіді користувача на питання СІК. Ці дані є ствердними або заперечними відповідями. Далі будуються продукційні правила, що описують фрагменти фактичного знання.

Приклад (для ЕС ідентифікації і вибору породи собак).

```
dog_is(«англійський бульдог»): —
it_is(«короткошерста»),
positive(«має»,«зріст менше 22 дюймів»),
positive( «має», «звисаючий хвіст»),
positive(«має|», «хороший характер»), !.

/* Аналогічно описуються дані про інші породи собак */

dog_is(«кокер-спаніель»):—
```

it_is(«довгошерста»),
positive(«має», «зріст менше 22 дюймів»),
positive(«має», «звисаючий хвіст»),
positive(«має», «довгі вуха»),
positive(«має», «хороший характер»),!.

Причому, з метою обмеження простору пошуку описуючи пов'язані фрагменти знань, правила можуть бути згруповані через уведення в базу допоміжних правил для ідентифікації категорій. Наприклад, в ЕС вибору породи собаки, це буде правило it_is, яке ідентифікує породу собаки за ознакою належності групі довгошерстих або короткошерстих:

it_is(«короткошерста»): —
positive(«собака має», «коротку шерсть»), !.
it_is(«довгошерста»): —
positive(«собака має», «довгу шерсть»), !.

У наведеному прикладі основні дані для вибору породи собаки є загальновідомими, оскільки вибрані поширені породи собак. Набір ознак для класифікації порід вибраний на підставі таких міркувань:

- усі використовувані атрибути є необхідними для ідентифікації породи;
- жоден з атрибутів не є характерним для всіх порід одночасно.

Правила МВ зіставляють дані користувача з даними в продукційних правилах (правила negative і positive в даній ЕС), а також зберігають «трасу» ствердних і заперечних відповідей (правило remember для додавання в БЗ тверджень з відповідями 1 (так) і 2 (ні)), які використовуються у зіставленні зі зразком :

/* Механізм виведення експертного висновку.

xpositive(X,Y) і xnegative(X,Y) — предикати динамічної БД, зберігають, відповідно, ствердні і заперечні відповіді користувача на питання, які СІК задає на підставі значень аргументів предиката positive в тілі чергового варіанту правила dog_is*/

positive(X, Y): —
xpositive(X,Y)!.
positive(X, Y): — not(negative(X,Y)) !, ask(X,Y).
negative|(X, Y) — xnegative|(X,Y) !.
/* Предикат ask(X,Y) створює стандартне діалогове вікно для отримання

відповіді користувача на питання Так/Ні */

```
ask|(X,Y):—
concat(«Питання : », X, Temp),
concat(Temp, « », Temp1),
concat(Temp1, Y, Temp2),
concat(Temp2, «?», Quest),
Reply1=dlg_Ask(«Консультація», Quest, [«Так», «Ні»]),
Reply=Reply1+1,
remernber(X, Y, Reply).
```

```
/*Занесення відповідей користувача до динамічної БД */
remember(X, Y, 1):—!,
assertz(xpositive(X, Y)).
remember(X, Y. 2):—!,
assertz(xnegative(X, Y)), fail.
```

Експертна система, що базується на логіці, і її реалізація

Тут БЗ складається з тверджень у вигляді тверджень логіки предикатів. Частину тверджень описують об'єкти, інша частина - умови або атрибути, які характеризують різні об'кти. Кількість ознак визначає ступінь точності класифікації. Інтерпретатор всередині системи виконує свої функції на підставі слідкуючої схеми :

1. Система містить у БЗ речення, які управляють пошуком і співставленням. Інтерпретатор співставляє ці речення з елементами даних у БД.

2. Якщо існує можливість виклику більш одного правила, то для вирішення конфлікту система використовує можливості механізму внутрішньої уніфікації Прологу.

3. Система отримує результати процесу уніфікації автоматично, тому вони передаються на потрібний (логічний) пристрій виведення інформації. Так само, як і в ЕС, що базується на правилах, цей циклічний процес є процесом «розпізнавання – дія».

Основна відмінність структури ЕС, що базується на логіці, полягає в описі об'єктів і атрибутів у вигляді фактів.

```
/* Умови – характеристики різних  порід */

cond (1, «короткошерста порода»).
cond (2, «довгошерста порода»).
cond (3, «зріст менше 22 дюймів»).
cond (4, «зріст більше 30 дюймів»).
cond (5, «звисаючий хвіст»).
cond (6, «довгі вуха»).
cond (8, «вага більше 100 фунтів»).

/* Дані про типи порід */

topic («короткошерста»).
topic(«довгошерста»).

/* Дані про конкретні породи */

rule(1, «собака», «короткошерстий», [1]).
rule(2, «собака», «довгошерстий», [2]).
rule(3, «короткошерста», «англійський бульдог», [3,5.7]).
rule(4, «короткошерста», «гончак», [3,6.7]).
rule(5, «короткошерста», «данський дог», [5,6,7,8]).
rule(6, «короткошерста», «американський фокстер'єр», [4.6,7|).
rule(7, «довгошерста», «кокер-спаніель», [3,5.6,7]).
rule(8, «довгошерста», «ірландський сетер», [4,6]).
rule(9, «довгошерста», «коллі», [4,5,7]).
rule(10, «довгошерста», «сенбернар», [5,7,8]).
```

Третій аргумент предиката rule – список цілих чисел – номерів умов з речень типу cond. Кожне речення типу cond задає свою ознаку з використовуваних тут для класифікації порід собак. У ЕС, що базується на логіці, інтерпретатор використовує ці номери умов, щоб робити відповідний вибір.

МВ містить правила обробки списків атрибутів в описах об’єктів. За допомогою застосування правила go МВ проглядає твердження БЗ rule і БЗ cond для того, щоб з’ясувати за допомогою правила check наявність або відсутність відповідних значень даних.

```
/* Початкове правило механізму висновку */

go(_, Mygoa1 _, _)): —
  not(rule((_, Mygoa1 _, _)), !,
concat(«Порода, що рекомендується: », Mygoa1, Temp),
concat(Temp, «.», Result),
 dlg_Note|(«Експертний висновок : », Result).
```

```
go (History, Mygoal): —
rule(Rule_number, Mygoal, Type_of_breed, Conditions),
      check(Rule_number, History, Conditions),
      go([Rule_number|History], Type_of_breed).
```

/*Співпівставлення вхідних даних користувача зі списками атрибутів окремих порід собак */

```
check(Rule_rmmber, History, [Breed_cond|Rest_breed_cond_list]): —
      yes(Breed_cond), !, check(Rule_number, History, Rest_breed_cond_list).
check(_, _, [Breed_cond|_]): — no(Breed_cond),!, fail.
check(Rule  number, History, [Breed_cond| Rest_breed_cond_list]): —
cond(Breed_cond, Text),
ask_question(Breed_cond, Text),
check(Rule__number, History, Rest_breed_cond_list).
check(_, _, []).
do_answer(Cond_number, 1): — !, assertz(yes(Cond_number)).
do_answer(Cond_number, 2): —!, assertz(no(Cond_nurnber)), fail.
```

/*Запит і отримання відповідей yes і no від користувача */

```
ask_question(Breed_cond, Text): —
concat(«Питання: », Text. Temp),
concat|(Temp, « », Temp1),
concat(Temp1 «?». Quest),
Response1 = dlg_Ask(«Консультація», Quest, [«Так»,»Ні»]),
Response=Responsel+l, do_answer(Breed_cond, Response).
```

Правило go намагається співставити об’єкти, класифіковані за допомогою номерів умов. Якщо зіставлення відбувається, то цей модуль програми повинен додати в БЗ зіставлені значення і продовжити процес з новими даними, отриманими від користувача Якщо зіставлення не відбувається, механізм зупиняє поточний процес і вибирає для зіставлення іншу трасу. Пошук і співставлення продовжується до тих пір, поки не вичерпані всі можливості.

Перевагою ЕС, що базується на логіці, є можливість зберігання фактів Бази знань в твердженнях динамічної Бази Даних При цьому База знань може бути розміщена у файлі на диску, що дозволяє зробити її незалежною від програмного коду.

2 ВИМОГИ ЩОДО НАПИСАННЯ ТА ОФОРМЛЕННЯ РОЗРАХУНКОВОЇ РОБОТИ

2.1 Загальні вимоги щодо оформлення

Розрахунково-графічна робота (далі – РГР) має бути надрукована на аркушах білого паперу формату А4 (210*297 мм) за допомогою комп'ютерної техніки на одній стороні аркушу. Шрифт тексту записки 14 пт. Times New Roman, назви розділів – шрифт Arial, всі літери великі, назви підрозділів – шрифт Arial, написання літер як у реченнях.

Послідовність розміщення матеріалу в РГР має бути такою:

– титульний аркуш;

– перелік умовних скорочень;

– зміст;

– вступ;

– аналіз предметної галузі;

– розробка бази знань;

– програмна реалізація експертної системи;

– висновок;

– список використаної літератури;

– додатки.

На титульному аркуші, переліку скорочень та змісті присутня контурна рамка. Зразок титульного аркуша наведено у Додатку А.

Кожен аркуш РГР, окрім першого аркушу розділу «Вступ», має мати рамку та основний напис відповідно до ДСТУ 34.201-89 (рис. Б2). На першому аркуші розділу «Вступ» виконується основний напис за формою, наведеною на рис. Б1.

У графах основного напису, наведеного на рис. А1, вказують (номери граф на рисунку показані в дужках):

– у графі 1 – найменування теми РГР;

– у графі 2 – шифр РГР;

– у графі 7 – порядковий номер аркушу, нумерація сторінок наскрізна по РР (нумерують сторінки, починаючи з титульної, номер проставляють лише на

сторінках, на яких присутня рамка основного напису);

– у графі 8 – загальна кількість аркушів РР;

– у графі 9 – скорочені назви кафедри та навчального закладу;

– у графі 11 – прізвища осіб, що підписали документ;

– у графі 12 – підписи осіб, прізвища яких зазначені у графі 11;

– у графі 13 – дату підписання документу.

Графи 4, 14 – 18 не заповнюються.

Відстань від лівого та правого країв аркушу до тексту має складати 25 та 10 мм відповідно, від верхнього та нижнього краю аркушу до тексту – 10 мм та 25 мм відповідно. Відступ абзацу – 12,7 мм.

РГР повинна бути написана чіткою та зрозумілою літературною мовою без граматичних та стилістичних помилок.

Текст РГР викладається, як правило, у безособовій формі, наприклад, «... завданням передбачено...» чи «... завданням передбачається...». При описанні операцій, які виконуються людиною, рекомендується використовувати третю особу множини чи однини, наприклад, «... вводять необхідні дані», «... користувач системи має змогу...».

2.2 Вимоги щодо змістовної частини

У РР необхідно розробити експертну систему для заданої предметної галузі. Для розробки ЕС можна використовувати будь-яку мову програмування, проте рекомендованою є мова Prolog, оскільки вона дозволяє розробити мінімальний програмний код завдяки вбудованій «машині висновку». Така програма матиме невеликий програмний текст та виконуватиме всі необхідні функціональні дії.

У роботі необхідно провести аналіз заданої предметної галузі, описати, для кого призначена розроблювана система та які функціональні можливості вона матиме. Далі слід вибрати структурну модель бази знань (продукційна система, семантична мережа, фрейми та їх комбінація, інші моделі) і розробити структуру даних системи, визначити структуру бази даних, типи даних та їх найменування, виходячи з типу розроблюваної ЕС, розробити механізм

заповнення, модифікації та збереження бази даних. Після цього слід розробити базу знань та здійснити опис зібраних знань формальними способами на базі вибраної моделі. Далі розробляється структура даних для ведення бази правил, а також механізм заповнення, модифікації та збереження бази правил.

Наступним кроком є розробка алгоритмів обробки даних та знань. Для продукційної системи – це, в першу чергу, алгоритми прямого та зворотного ланцюгу роз'яснення. Далі розробляють блок роз'яснень прийнятого рішення, інтерфейс системи, діалогові вікна та меню.

Кількість об'єктів експертної системи має бути не менша 12, а кількість атрибутів, що їх описують, не менша 8.

3 ВАРІАНТИ ЗАВДАНЬ ДЛЯ ВИКОНАННЯ РОЗРАХУНКОВО-ГРАФІЧНОЇ РОБОТИ

Варіант предметної галузі обирається студентом за номером у журналі згідно з табл. 1.1. Студент може сам запропонувати варіант предметної галузі й узгодити його з викладачем за умови, що даний варіант не наданий у табл. 1.1.

Таблиця 1.1 – Варіанти предметної галузі

№	Назва предметної галузі
1	audio-системи
2	алгоритми обробки сигналів
3	алгоритми пошуку
4	алгоритми сортування
5	алгоритми кластерізації
6	алгоритми шифрування
7	бази даних
8	браузери
9	відео-системи
10	водні транспортні засоби

Продовження таблиці 1.1

11	діагностика комп’ютера
12	запам’ятовуючі пристрої
13	інструменти
14	комп’ютери
15	комп’ютерна техніка
16	комп’ютерні віруси
17	комп’ютерні ігри
18	комп’ютерні мережі
19	комп’ютерні системи
20	комп'ютерні мережі
21	материнські плати
22	матеріали для ремонту
23	мережне обладнення
24	мікропроцесори
25	мобільні телефони
26	мови програмування
27	монітори
28	операційні пристрої
29	операційні системи
30	офісна техніка
31	периферійне обладнання
32	побутова техніка
33	програмне забезпечення
34	текстові редактори
35	технічна література

4. КРИТЕРІЇ ОЦІНЮВАННЯ РОЗРАХУНКОВО-ГРАФІЧНОЇ РОБОТИ

Загальна оцінка, яку може отримати студент, за виконання курсового проекту, становить 20 балів та складається з чотирьох основних частин

- представлена на захист експертна система,
- зміст та оформлення пояснювальної записки,
- захист роботи,
- відповіді на додаткові запитання підчас захисту.

Пояснювальна записка має містити детальний опис етапів створення та програмного коду системи з дотриманням діючих норм і стандартів на оформлення технічної документації.

Представлений на захист ілюстративний матеріал повинен достатньою мірою відображати хід виконання РГР та результати проектування.

У доповіді під час захисту студент повинен продемонструвати теоретичні та практичні знання в галузі розробки експертних систем.

У разі успішної відповіді студента на додаткові питання, які виявлять його поглиблені знання щодо проектування систем штучного інтелекту, студент може отримати додаткові бали під час захисту. Розподіл балів по відповідних складових частинах наведено у табл. 4.1.

Таблиця 4.1 – Нарахування балів для оцінювання курсового проекту

Пояснювальна записка	Ілюстративна частина	Захист роботи	Відповіді на додаткові питання	Сума
до 5	до 5	до 10	до 5	20

СПИСОК ЛІТЕРАТУРИ

1. Глибовець М.М. Штучний інтелект. Підручник для студ. вищ. навч. закладів, що навчаються за спец. Компютер. науки та Приклад. математика. / М.М. Глибовець О.В. Олецький. – К.: Вид.дім КМ Академія, 2002. – 336с.
2. Гаврилова Т. А., Хорошевский В. Ф. Базы знаний интеллектуальных систем. / Т. А.Гаврилова, В. Ф. Хорошевский. – Спб.: Питер, 2001. – 380 с.: ил.
3. Хорошевский В. Ф. Языковые средства программирования // В кн.: Искусственный интеллект. Кн.3. Программные и аапаратные средства. / Хорошевский В. Ф. – М: Радио и связь, 1990.
4. Лорьер Ж.-Л. Системы искусственного интеллекта: Пер. с франц. / Лорьер Ж.-Л. – М.: Мир, 1991. – 568 с.
5. Джексон П. Введение в экспертные системы/ Джексон П. – М. : Издательский дом «Вильяме», 2001.
6. Адаменко А.Н. Логическое программирование и Visual Prolog./ А.Н. Адаменко, А.М. Кучуков. – СПб.: БХВ-Петербург, 2003. – 921 с.
4. Маллас Дж. Реляционный язык Пролог и его применение./ Маллас Дж. – М.: Наука, 1990. – 464 с.
5. Братко И. Программирование на языке Пролог для искусственного интеллекта: Пер. с анг. / Братко И. – М.: Мир, 1990. – 560 с.
6. Алгоритмы искусственного интеллекта на языке PROLOG, 3-е издание. : Пер. с англ. — М. : Издательский дом «Вильяме», 2001. — 640 с. :ил.
7. Нейлор К. Как построить свою экспертную систему: Пер. с англ. / Нейлор К. – М.: Энергоатомиздат, 1991. - 286с.: ил.
8. Гаврилова Т.А. Извлечение и структурирование знаний для експертных систем./ Т.А. Гаврилова, К.Р.Червинская. — М.: Радио и связь, 1992. – 200 с.
9. Выявление експертных знаний / О.И. Ларичев, А.И. Мечитов, Е.М. Мошкович, Е.М. Фуремс. – М.: Наука, 1989. — 128 с.

Додаток А

Зразок оформлення титульної сторінки РГР

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КРЕМЕНЧУЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ МИХАЙЛА ОСТРОГРАДСЬКОГО

Кафедра комп'ютерних та інформаційних систем

РОЗРАХУНКОВО-ГРАФІЧНА РОБОТА
З НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
«ДОСЛІДЖЕННЯ ТА АНАЛІЗ
КОМП'ЮТЕРНИХ СИСТЕМ ШТУЧНОГО ІНТЕЛЕКТУ»

<номер залікової книжки>

Виконав студент гр.<шифр> <ПІБ>

Перевірив <ПІБ>

Кременчук 2018

Додаток Б

Форми основного надпису

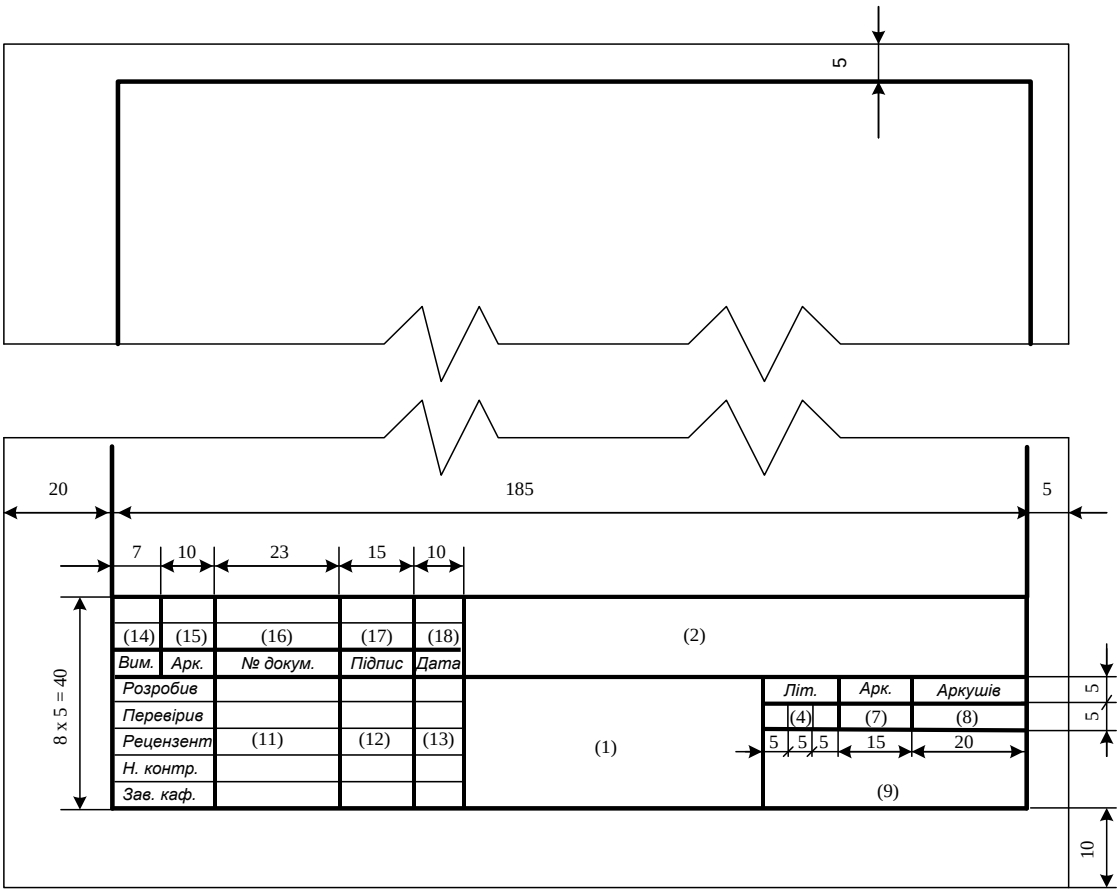

Рисунок Б.1 – Основний напис на першому аркуші розділу

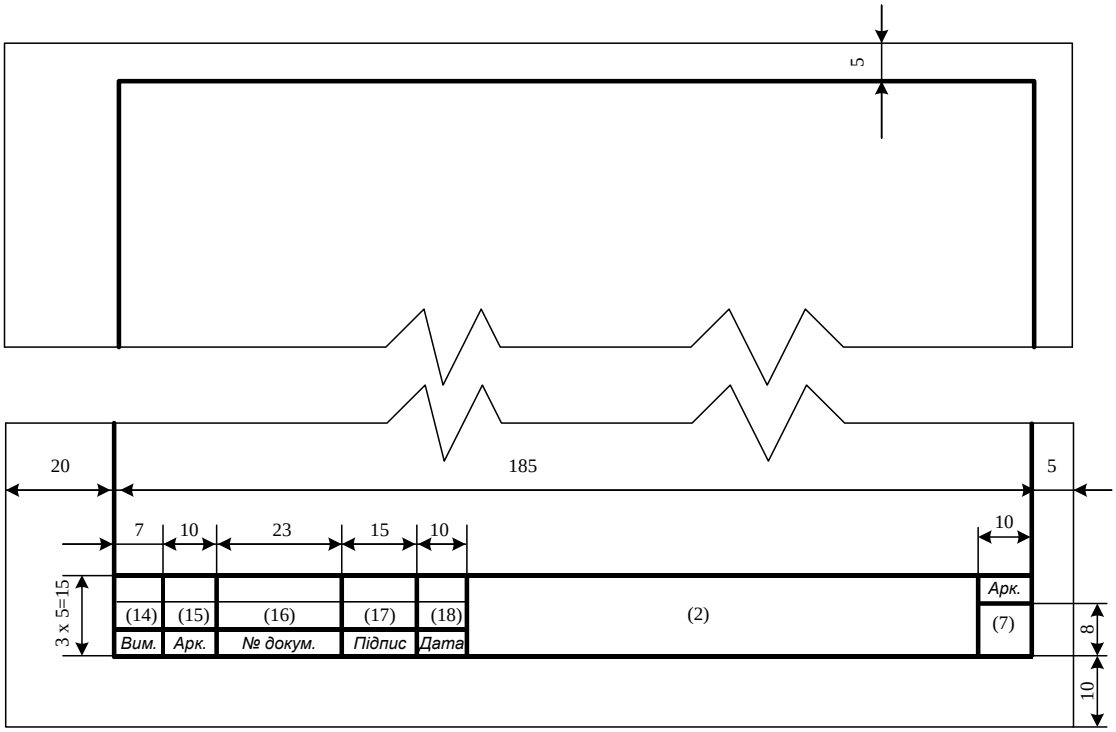

Рисунок Б.2 – Основний напис на наступних аркушах записки

Додаток В

Приклад програмної реалізації експертної системи на мові Turbo Prolog

```
/* ------------------------------------------------------------------ */
%trace правило обратный_вывод cc объединить
% trace find get vivod правило con
nowarnings

DOMAINS   s = string  i = integer  c=char  file=f  sp=k*  k=STRING
    a = integer  si=a*

DATABASE   правило(i,sp,sp)

PREDICATES
    доказать(sp,sp,i,sp,sp)
    c(i,sp,sp,sp,sp,sp)
    cc(i,sp,sp,sp,sp,sp)
          ввод_условий(i,sp,i)
        ввод_выводов(i,sp,i)
          прямой_вывод_(sp,sp,sp,i)
          обратный_вывод(sp,sp)
    список_выводов(sp,i)
    список_условий(sp,i)
    соединить(sp,sp,sp)
    объединить(sp,sp,sp)
    сод_ся(sp,sp,i,sp,sp)
    поиск_в_БП(sp,sp,i,sp,sp)
    меню
        пункт(c)
        item(c)
        вывод_усл(i,sp,sp)
        просмотр_списка(i,sp,sp)
    con(sp,sp,sp)
        vivod(sp,sp)
        find(sp,sp,i,sp,sp)
        get(sp,sp,sp,i)

CLAUSES
/*************************************************************/
/*************************************************************/
вывод_усл(N,L,K):- write(" -",N), N1=N-1, вывод_усл(N,L1,Q), L=[Str|L1] .

список_выводов([],0).
список_выводов(Z,K):- write("   Вводите доказываемый вывод : "),
```

```
readln(Str), nl,
      N1=K-1, список_выводов(Z1,N1), Z=[Str|Z1].
    список_условий([],0).
    список_условий(Z,KL):- write("  Вводите условие : "), readln(Str), nl,
      N1=KL-1, список_условий(Z1,N1), Z=[Str|Z1].
    /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */
    просмотр_списка(0,[],[]).
    просмотр_списка(N,L,Q):-
        write("-------------------------------------------------------"),
     nl, write("                        ПРАВИЛО - ", N),nl,
        write("   Если :"), nl, write("         ",L),nl,nl,nl,
        write("     То :"), nl, write("         ",Q),nl,
        write("-------------------------------------------------------"),
        nl,nl, readchar(_), fail.
    /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */
    /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */
    ввод_условий(N,[],0).
    ввод_условий(N,L,K):- write("  Введите условие : "), readln(Str), N1=K-1,
ввод_условий(N,L1,N1), L=[Str|L1].

    ввод_выводов(N,[],0).
    ввод_выводов(N,Q,M):-write("  Вводите вывод :"), readln(Str),
                  N2=M-1, ввод_выводов(N,Q1,N2), Q=[Str|Q1].
    /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */
    /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */
                  /*   Прямой вывод    */
    прямой_вывод_(Temp,Test,Prn,0).
    прямой_вывод_(Temp,Test,Prn,Flag):-  Temp=Test.
    прямой_вывод_(Temp,Sp,Print,Flag):- правило(N,L,Q),
      сод_ся(Temp,Temp,N,L,Q), c(N,Temp,Spisok1,Q,Q,Prn), nl,
      write("\t\t\t\t     вывод - ",Prn, "; "), readchar(SS),
      прямой_вывод_(Spisok1,Temp,Prn,N).

    /*          ----------------------------------------------          */
    /* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */
    соединить(Spisok,Spisok1,[]).
    соединить(Spisok,Spisok1,Q):- Q=[H|T],  Spisok1=[H|Spisok], Q1=T,
      соединить(Spisok,Spisok1,Q1).

    c(0,Spisok,Spisok1,[],[],Prn).
    c(N,Spisok,Spisok1,[],Q,Prn).
    c(N,Spisok,Spisok1,Sp,Q,Q):- Sp=[H|T],  Spisok1=[H|Spisok], Q1=T,
      c(N,Spisok,Spisok1,Q1,Q,Q).

    сод_ся(Spisok,S,N,[],Q):-retract(правило(N,_,_)).
```

```
сод_ся(Spisok,[],N,[],Q).
сод_ся(Spisok,Temp,N,[HL|TL],Q):- Temp=[H|T], H=HL,
соединить(Spisok,Spisok1,Q),
  сод_ся(Spisok1,Spisok1,N,[],Q).

сод_ся(Spisok,Temp,N,L,Q):- Temp=[H|T], сод_ся(Spisok,T,N,L,Q).

/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */
поиск_в_БП(Spisok,S,N,L,[]):-retract(правило(N,_,_)).
поиск_в_БП(Spisok,[],N,L,[]).
поиск_в_БП(Spisok,Temp,N,L,[HQ|TQ]):- Temp=[H|T], H=HQ,
  объединить(Spisok,Spisok1,L),
  con(Spisok,L,Spisok), write("/n con ",Spisok),readchar(E),
       поиск_в_БП(Spisok,Spisok,N,L,[]).

поиск_в_БП(Spisok,Temp,N,L,Q):- Temp=[H|T],
поиск_в_БП(Spisok,T,N,L,Q).

объединить(Spisok,Spisok1,[]).
объединить(Spisok,Spisok1,L):- L=[H|T],  Spisok1=[H|Spisok],
  объединить(Spisok1,Sp,T).
cc(0,Spisok,Spisok1,[],[],Prn).
cc(N,Spisok,Spisok1,[],L,Prn).
cc(N,Spisok,Spisok1,Sp,L,L):- Sp=[H|T],  Spisok1=[H|Spisok], L1=T,
  cc(N,Spisok,Spisok1,L1,L,L).
доказать(Uslov,Vivod,0,L,Q).
доказать(Uslov,Vivod,N,L,Q):- Vivod=Q, Uslov=L,nl,nl,
  write(" \t\t\t #####   Обратный вывод   #####"),nl,
  write(" Если : ",L," То : ",Q), nl,
  write("\t Вывод  действительно справедлив."),nl,nl,nl,
  write("\t \t \t     Нажмите любую клавишу...").
доказать(Uslov,Vivod,0,L,Q).

/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - */
                  /*   Обратный вывод    */
обратный_вывод(Temp,Test):- Temp=Test.
обратный_вывод(Temp,Test):-  правило(N,L,Q),
поиск_в_БП(Temp,Temp,N,L,Q),
       write(" Вывод выполняется !!! ( ",  L," )"),nl, readchar(_),
  cc(N,Temp,Spisok1,L,L,Prn),nl,
  обратный_вывод(Spisok1,Temp).

/*              Объединение двух списков                    */
con([],L,L).
con([X|L1],L2,[X|L3]):- con(L1,L2,L3).
```

```
/*            Поиск эл-ов в списке выводов                */
find(_,_,0,_,_):- readchar(D), clearwindow, меню.
find(Res,In,N,L,[]):-retract(правило(N,_,_)).
find(Res,In,N,L,Q):- Q=[HQ|TQ], In=[H|T], H=HQ,
            write(" Если : ",L," То :",Q), nl,
            con(In,L,Res),
           find(Res,T,N,L,QT).

find(Res,In,N,L,Q):- In=[H|T],     find(Res,T,N,L,Q).

get(In,Test,Res,K):- K=0.
get(In,Test,Res,K):- правило(N,L,Q),
            find(Res,In,N,L,Q),
            get(Res,Res,Res,N).

/* Temp - входящий список (выводы); Res - выходной список !!!        */
vivod(Temp,Res):- get(Temp,Temp,Res,1),
          write("\n Find the result ==> ",Res),
          readchar(E).
/*        ------------------------------------------------          */

 меню :- makewindow(2,26,31," Меню пользователя ",5,28,11,23,1,255,"г¬L-
=¦"),
        clearwindow, nl,nl,
        write("~~~~~~~~~~~~~~~~~~~~"), nl,
        write(" 1- Просмотр       "), nl,
        write(" 2- Добавить       "), nl,
        write(" 3- Удалить        "), nl,
        write(" 4- Прямой вывод   "), nl,
        write(" 5- Обратный вывод "), nl,
        write(" 6- Выход          "), nl,
        write("~~~~~~~~~~~~~~~~~~~~"), nl, readchar(C),
        gotowindow(1), clearwindow,  пункт(C), clearwindow, nl, меню .

 /*                Ппросмотр базы правил                          */
пункт('1'):-  правило(N,L,Q), просмотр_списка(N,L,Q).
пункт('1'):- clearwindow, меню.
  /*                Добавление правил                            */
пункт('2'):-  write("       Введите номер правила : "),
      readint(N),
      write("  Введите количество вводимых условий: "), readint(K),
      ввод_условий(N,L,K),nl,
      write("       Введите количество выводов :"), readint(M),
      ввод_выводов(N,Q,M), openappend(f,"sps.pro"),
```

```
        assertz(правило(N,L,Q)), save("sps.pro"), closefile(f),
        clearwindow,  меню.

    /*                  Удаление правил                  */
    пункт('3'):- write(" Введите номер удаляемого правила :  "), readint(N),
        consult("sps.pro"), openmodify(f,"sps.pro"), retract(правило(N,L,Q)),
        save("sps.pro"), closefile(f), clearwindow, меню .

    /*              Прямой вывод                         */
    пункт('4'):- clearwindow, nl, write(" \t Введите количество вводимых
условий :  "),
        readint(K), nl, nl, список_выводов(Temp,K),  Vivod_Pr=Temp,
clearwindow,
        nl,nl,nl,write("   ------------  Вывод : ",Vivod_Pr), nl,
        write("   выполняется тогда, когда выполняется  : "), nl,
        прямой_вывод_(Temp,[z,x],Prn,1), nl,nl,
        readchar(_), consult("sps.pro").

    /*              Обратный вывод                       */
    пункт('5'):- write("\t\t  Выберите один из вариантов обратного вывода :
"),
        nl, nl, write(" a) Справедлив ли вывод ? "), nl,
            write(" b) Когда справедлив вывод ? "), readchar(CH),
            item(CH), clearwindow.
    /*                  Выход                            */
    пункт('6'):- makewindow(2,0,15,"-",0,0,25,80,1,255,"-"), exit.

    item('a'):-  clearwindow, nl,
    write("**********************************************************
**"),
    write("*  Вы получите ответ только тогда, когда вывод имеется в БД .
*"),
    write("*  Если в БД вывод не содержиться, то система проигнорирует
ваш запрос !!!   *"),
    write("**********************************************"),
nl, nl,
    write(" Введите кол-во вводимых выводов для доказательства: "),
        readint(K), nl, nl,
        список_выводов(Vivod,K),
        write(" Введите кол-во условий при которых вывод - "),
        write( Vivod ," , должен выполняться : "),
        readint(KL), nl, nl, список_условий(Uslov,KL), правило(N,L,Q),
        доказать(Uslov,Vivod,N,L,Q),
        readchar(_).
```

```
item('b'):-  clearwindow, nl,
             write(" Введите кол-во вводимых выводов: "), readint(K), nl, nl,
      список_выводов(Temp,K), Vivod_Obr=Temp, clearwindow, nl,nl,nl,
      vivod(Temp,Res),      write(" Цепочка решений : ",Res),
  %    обратный_вывод(Temp,[]),
      readchar(_), меню.

GOAL
consult("sps.pro"),
makewindow(1,91,62,"",0,0,25,80,1,255,"г¬L-=¦"), nl,
write("                    Экспертная система          "), nl,
write("=========================================="),
nl, nl,  date(A,B,C), write("       Сегодня : ",C,'/',B,'/',A),
nl, time(H,M,S,MS), write("       Текущее время : ",H,'ч',' ',M,'м',' ',S,"сек"),
cursor(15,5),
cursor(22,2), write("                    Нажмите любую клавишу..."),
readchar(_), clearwindow, меню, fail.
```

Методичні вказівки щодо виконання розрахунково-графічної роботи з навчальної дисципліни «Дослідження та аналіз комп'ютерних систем штучного інтелекту» для студентів денної форми навчання зі спеціальності 123 – «Комп'ютерна інженерія» освітнього ступеня «Магістр»

Укладачі: к.т.н., доцент П. П. Костенко, ст. викл. А. Л. Юдіна

Відповідальний за випуск зав. каф. КІС А. В. Луговой

Підп. до др. _____. Формат 60х84 1/16. Папір тип. Друк ризографія.
Ум. друк. арк. ____. Наклад _____ прим. Зам. №_____. Безкоштовно.

Видавничий відділ Кременчуцького національного університету
імені Михайла Остроградського
вул. Першотравнева 20, м. Кременчук, 39600