ЛАБОРАТОРНА РОБОТА №3-4.

РАЗРОБКА ПРОТОТИПУ ДІАГНОСТИЧНОЇ ЕКСПЕРТНОЇ СИСТЕМИ

Мета роботи:

1. Дослідити предметну область, сформувати для неї поле знань, список фактів, а також правила для роботи з ними.
2. Оволодіти базовими конструкціями мови представлення знань CLIPS, такими як deftemplate, deffacts, defrule, deffunction, defglobal.
3. Освоїти принципи пошуку рішення в експертних системах, заснованих на правилах виду "ЯКЩО-ТО", формування послідовності активації правил при виведенні результату.

Задачі роботи:

1. Описати словесно факти і правила для розроблюваного прототипу предметної області, представити можливу ієрархію понять.
2. Перекласти факти і правила в синтаксис мови CLIPS.
3. Продемонструвати працездатність прототипу на конкретних прикладах.

Варіанти прототипів експертних систем:

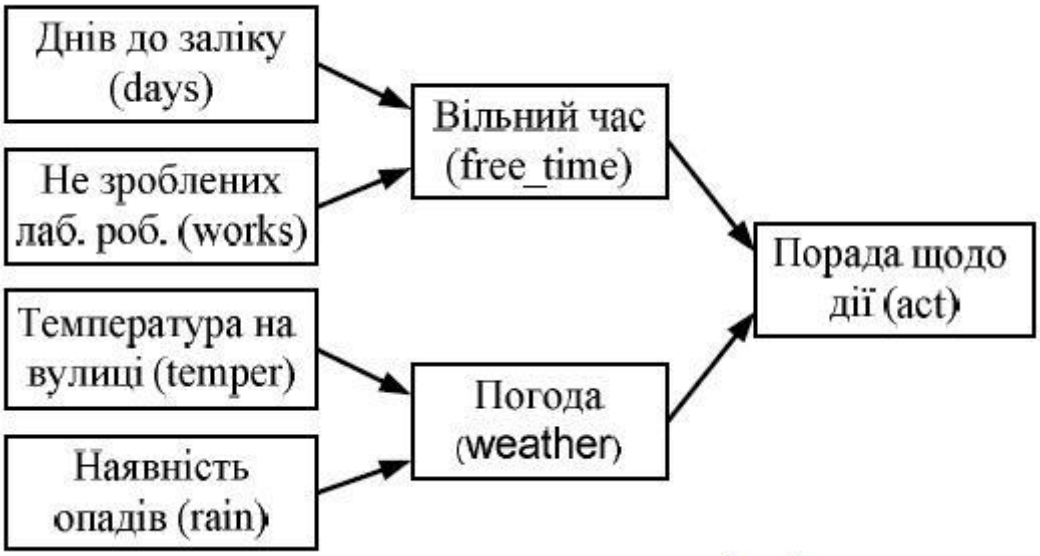
1. Експертна система «Організація відпочинку».
2. Експертна система «Фрукти»
3. Експертна система «Автомобілі»
4. Експертна система «Квіти»
5. Експертна система «Предмети навчання»
6. Експертна система «Професія»
7. Експертна система «Пори року»
8. Експертна система «Птахи»
9. Експертна система «Риба»
10. Експертна система «Жанри музики»
11. Експертна система «Жанри фільмів»
12. Експертна система «Жанри книг»
13. Експертна система «Міста України»
14. Експертна система «Вид спорту»
15. Експертна система «Овочі»
16. Експертна система «Ягоди»
17. Експертна система «Гриби»

21. Експертна система «Дерева»
22. Експертна система «Захворювання»
23. Експертна система «Напої»
24. Експертна система «Країни»
25. Експертна система «Університети»

Приклад оформлення лабораторної роботи ЕС (експертна система) – порадник студенту.

ЕС повинна виробляти рекомендації

студентові напередодні заліку й мати чотири вхідні змінні («число днів до заліку», «кількість незроблених лабораторних робіт (в %)», «температура на вулиці» і «наявність опадів»), дві проміжні («вільний час» і «погода») і вихідну змінну («поради щодо дії»). Структура залежності змінних показана на рис., у дужках зазначені можливі імена змінних.




```
(clear)
(defrule data-input
(initial-fact)
=>
(printout t crlf "Введіть число днів до заліку (ціле значення): ")
(bind ?days (read))
(assert (days ?days))
(printout t crlf "Введіть число не зроблених лаб. робіт (в %) ")
(bind ?works (read))
(assert (works ?works))
(printout t crlf "Введіть температуру на вулиці: ")
(bind ?temper (read))
(assert (temper ?temper))
(printout t crlf "Чи є на вулиці опади? (так - 1/ні - 0): ")
(bind ?rain (read))
(assert (rain ?rain)))
(defrule R1
(days ?days)
(works ?works)
(test (= ?works 0))
=>
(printout t crlf crlf "Все вже зроблено." crlf)
(assert (freetime "infinity"))
(assert (freetimecnst 0)))
(defrule R2
```

```
(days ?days)
(works ?works)
(test (and (and(> ?days 5) (<= ?days 7)) (and (<= ?works 50) (>
? works 0) )))

=>
(printout t crlf crlf "Вільного часу багато" crlf)
(assert (freetime "mnogo"))
(assert (freetimecnst 1)))
(defrule R3
(days ?days)
(works ?works)
(test (and (and(> ?days 5) (<= ?days 7)) (and (<= ?works 100) (> ?
works 50) )))
=>
(printout t crlf crlf "Вільного часу не дуже багато" crlf)
(assert (freetime "ne_ochen"))
(assert (freetimecnst 2)))
(defrule R4
(days ?days)
(works ?works)
(test (and (and(> ?days 3) (<= ?days 5)) (and (<= ?works 50) (> ?
works 0))))
=>
(printout t crlf crlf "Вільного часу багато" crlf)
(assert (freetime "mnogo"))
(assert (freetimecnst 1)))
(defrule R5
(days ?days)
(works ?works)
(test (and (and(> ?days 3) (<= ?days 5)) (and (<= ?works 100) (> ?
works 50))))
=>
(printout t crlf crlf " Вільного часу не дуже багато " crlf)
(assert (freetime "ne_ochen"))
(assert (freetimecnst 2)))
(defrule R6
(days ?days)
(works ?works)
(test (and (= ?days 3) (and ( > ?works 0 ) (<= ?works 50)
))) =>
(printout t crlf crlf " Вільного часу не дуже багато " crlf)
(assert (freetime "ne_ochen"))
(assert (freetimecnst 2)))
(defrule R7
(days ?days)

(works ?works)
(test (and (= ?days 3) (and ( > ?works 50 ) (<= ?works 100) )))
=>
(printout t crlf crlf "Вільного часу зовсім не багато. Час
робити" crlf)
```

```
(assert (freetime "pora_delat"))
(assert (freetimecnst 3)))
(defrule R8
(days ?days)
(works ?works)
(test (and (= ?days 2) (and ( > ?works 0 ) (<= ?works 33)
))) =>
(printout t crlf crlf " Вільного часу не дуже багато " crlf)
(assert (freetime "ne_ochen"))
(assert (freetimecnst 2)))
(defrule R9
(days ?days)
(works ?works)
(test (and (= ?days 2) (and ( > ?works 33 ) (<= ?works 66)
))) =>
(printout t crlf crlf " Вільного часу зовсім не багато. Час
робити " crlf)
(assert (freetime "pora_delat"))
(assert (freetimecnst 3)))
(defrule R10
(days ?days)
(works ?works)
(test (and (= ?days 2) (and ( > ?works 66 ) (<= ?works 100)
))) =>
(printout t crlf crlf "Вільного часу нема – не встигаєм" crlf)
(assert (freetime "finish"))
(assert (freetimecnst 4)))
(defrule R11
(days ?days)
(works ?works)
(test (and (= ?days 1) (and ( > ?works 0 ) (<= ?works 25)
))) =>
(printout t crlf crlf "Вільного часу не дуже багато" crlf)
(assert (freetime "ne_ochen"))
(assert (freetimecnst 2)))

(defrule R12
(days ?days)
(works ?works)
(test (and (= ?days 1) (and ( > ?works 25 ) (<= ?works 50)
))) =>
(printout t crlf crlf " Вільного часу зовсім не багато. Час
робити " crlf)
(assert (freetime "pora_delat"))
(assert (freetimecnst 3)))
(defrule R13
(days ?days)
(works ?works)
(test (and (= ?days 1) (and ( > ?works 50 ) (<= ?works 100)
))) =>
(printout t crlf crlf " Вільного часу нема – не встигаєм " crlf)
(assert (freetime "finish"))
```

```
(assert (freetimecnst 4)))
(defrule R14
(days ?days)
(works ?works)
(test (and (= ?days 0) ( > ?works 0 )))
=>
(printout t crlf crlf "Ну колись воно було. А зараз вже не
важливо" crlf)
(assert (freetime "ppc"))
(assert (freetimecnst 5)))
(defrule R15
(temper ?temper)
(rain ?rain)
(test (> ?temper 25))
=>
(printout t crlf crlf "Погода дуже добра " crlf)
(assert (weather "v-good"))
(assert (weathercnst 1)))
(defrule R16
(temper ?temper)
(rain ?rain)
(test (and(and(>= ?temper 5)(< ?temper 25)) (= ?rain 0))
) =>
(printout t crlf crlf "Погода добра" crlf)
(assert (weather "good"))

(assert (weathercnst 2)))
(defrule R17
(temper ?temper)
(rain ?rain)
(test (and(and(>= ?temper 5)(< ?temper 25)) (<> ?rain 0))
) =>
(printout t crlf crlf "Погода погана " crlf)
(assert (weather "bad"))
(assert (weathercnst 3)))
(defrule R18
(temper ?temper)
(rain ?rain)
(test (<= ?temper 5) )
=>
(printout t crlf crlf "Погода дуже погана" crlf)
(assert (weather "v-bad "))
(assert (weathercnst 4)))
(defrule R19
(weathercnst ?weathercnst)
(freetimecnst ?freetimecnst)
(test (and(< ?freetimecnst 3)(= ?weathercnst 1)))
=>
(printout t crlf crlf "Можна йти на прогулянку" crlf)
(assert (act "go")))
(defrule R20
(weathercnst ?weathercnst)
```

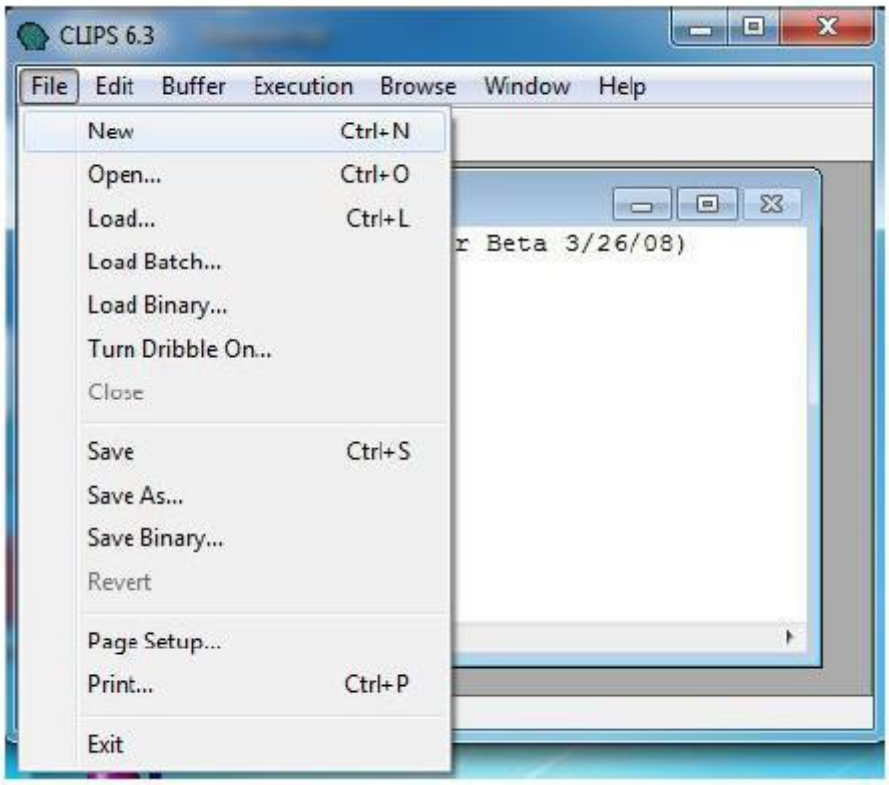
```
(freetimecnst ?freetimecnst)
(test (= ?freetimecnst 5))
=>
(printout t crlf crlf "Щодо погоди не знаю, але вчити вже
пізно" crlf)
(assert (act "nothing ")))
(defrule R21
(weathercnst ?weathercnst)
(freetimecnst ?freetimecnst)
(test (= ?freetimecnst 0))
=>
(printout t crlf crlf "Щодо погоди не знаю – готуюсь до наступної
сесії ..."
crlf)
(assert (act "botan ")))

(defrule R22
(weathercnst ?weathercnst)
(freetimecnst ?freetimecnst)
(test (and(= ?freetimecnst 4)(<> ?weathercnst 5)))
=>
(printout t crlf crlf "Потрібно вчити!" crlf)
(assert (act "learn")))
(defrule R23
(weathercnst ?weathercnst)
(freetimecnst ?freetimecnst)
(test (and(= ?freetimecnst 3)(= ?weathercnst 2)))
=>
(printout t crlf crlf "Краще вчитись" crlf)
(assert (act "learn")))
(defrule R24
(weathercnst ?weathercnst)
(freetimecnst ?freetimecnst)
(test (and(= ?freetimecnst 2)(= ?weathercnst 2)))
=>
(printout t crlf crlf "По желанию" crlf)
(assert (act "auw ")))
(defrule R25
(weathercnst ?weathercnst)
(freetimecnst ?freetimecnst)
(test (and(= ?freetimecnst 1)(= ?weathercnst 2)))
=>
(printout t crlf crlf "По желанию" crlf)
(assert (act "auw ")))
(defrule R26
(weathercnst ?weathercnst)
(freetimecnst ?freetimecnst)
(test (and( or (= ?freetimecnst 2)( = ?freetimecnst 1))(= ?
weathercnst 3)))
=>
(printout t crlf crlf "Краще вчити" crlf)
(assert (act "glearn")))
```

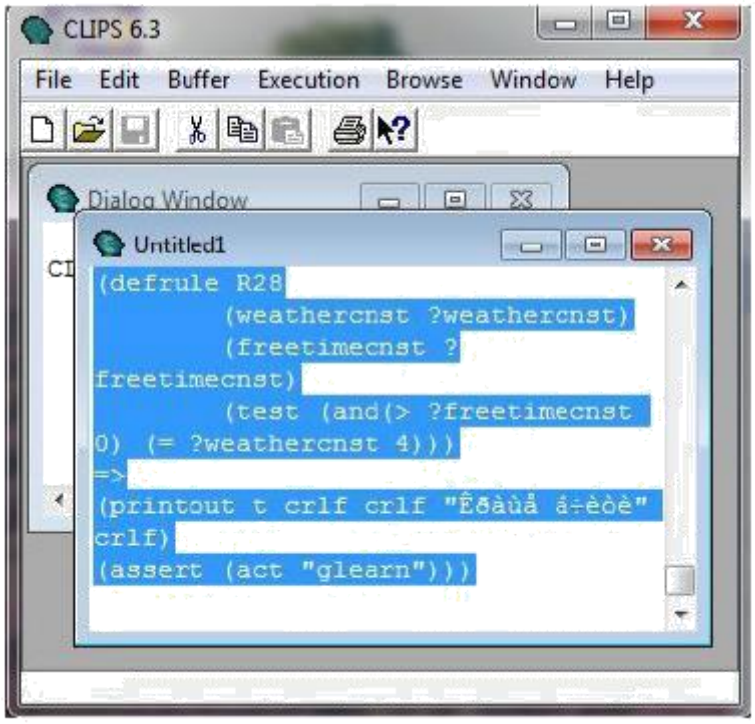
```
(defrule R27
(weathercnst ?weathercnst)
(freetimecnst ?freetimecnst)

(test (and(= ?freetimecnst 3)(= ?weathercnst 3)))
=>
(printout t crlf crlf "Краще вчити" crlf)
(assert (act "glearn")))
(defrule R28
(weathercnst ?weathercnst)
(freetimecnst ?freetimecnst)
(test (and(> ?freetimecnst 0) (= ?weathercnst 4)))
=>
(printout t crlf crlf "Краще вчити" crlf)
(assert (act "glearn")))
```

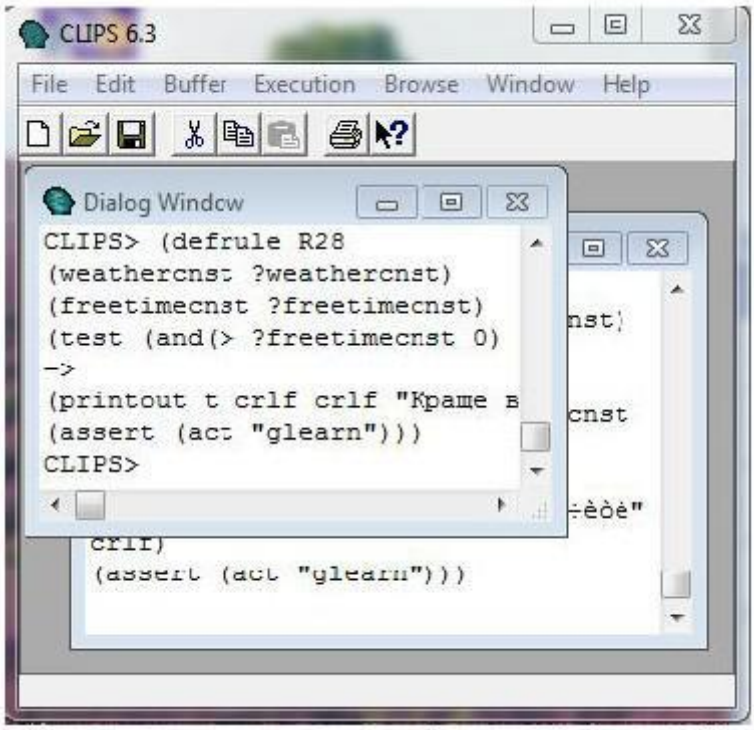
Демонстрація працездатності цієї ЕС на деякому прикладі виглядає так. Спочатку створюємо новий файл



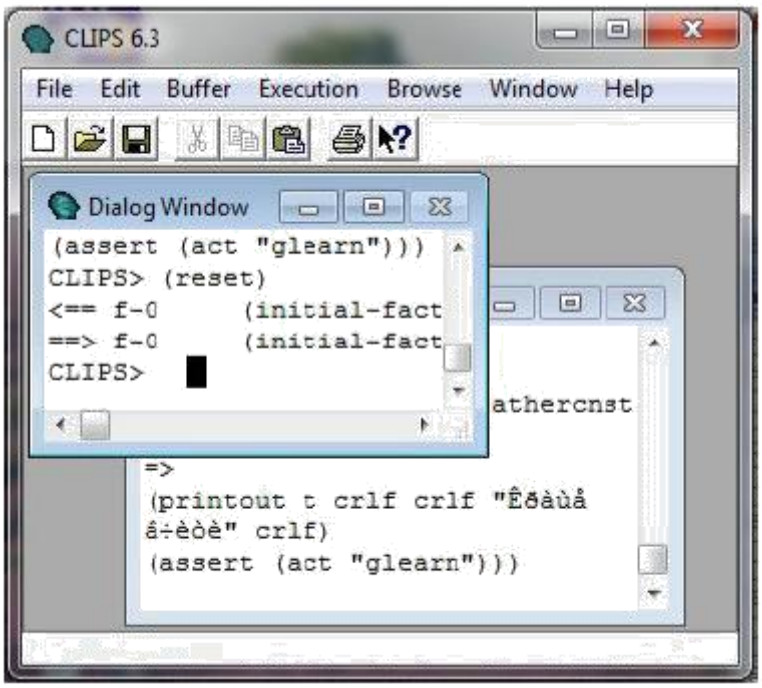
Потім набираємо лістинг програми у вікні вводу, виділяємо його



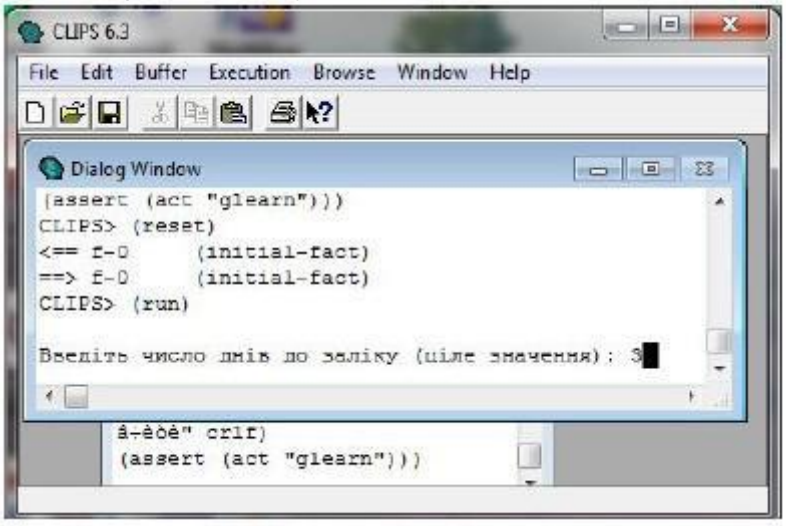
і користуючись Batch Selection у меню Buffer передаєм у вікно діалогу.



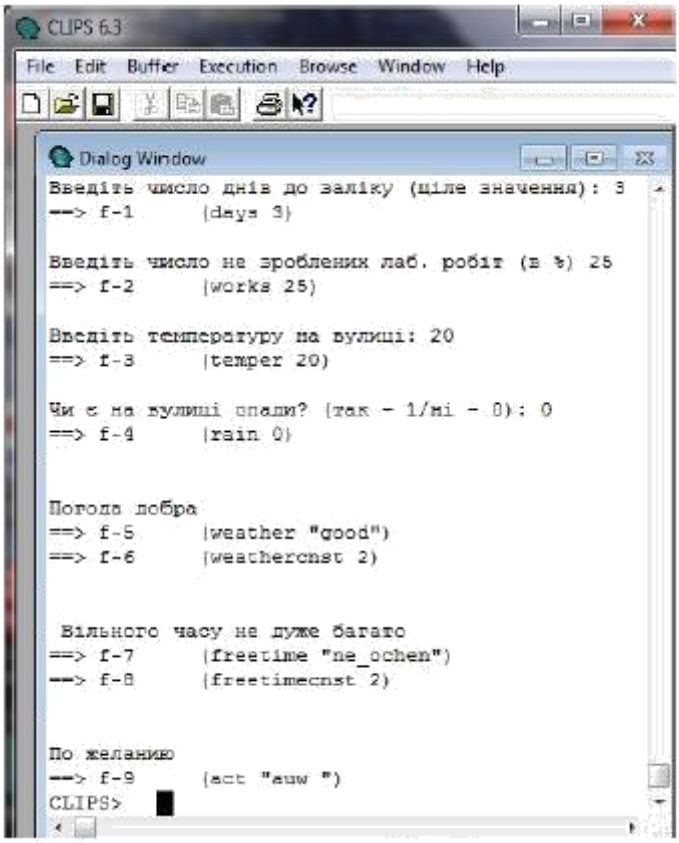
Оскільки додавання правил пройшло успішно, запустимо логічний вивід.



Для цього в діалоговому вікні послідовно виконують дві команди: (reset), а потім (run) . І починається діалог ЕС із користувачем. Нехай на запит експертної системи «Введіть число днів до заліку (ціле значення):» користувач повідомляє – 3.



Реакцією на це згідно із правилом data-input буде: спочатку зв'язування із змінною days значення 3, а потім функцією assert додано перший факт f-1 (days 3)



Згідно до відповіді на наступний запит «Введіть число не зроблених лаб. робіт в %» 25, із змінною works буде зв'язано значення 25 та додано факт f-2 (works 25)

Так само, згідно із відповідями на наступні
запити: «Введіть температуру на вулиці:» - 20;
«Чи є на вулиці опади? (так - 1/ні - 0):» - 0;
спочатку будуть додані факти f-3 (temper 20)
f-4 (rain 0)
Потім видано висновок системи про поточний стан погоди:
«Погода добра», додано факти
f-5 (weather "good")
f-6 (weathercnst 2)
Потім видано заключення про поточний стан засвоєння предмету:
«Вільного часу не дуже багато», додано факти
f-7 (freetime "ne_ochen")
f-8 (freetimecnst 2)
І, наприкінці, видано рекомендації ЕС щодо дій студента:
«По бажанню» та додано факт:
f-9 (act "auw")