

ТИТУЛ

Цель работы

Изучение основных возможностей и базовых команд среды CLIPS и практическое закрепление теоретических основ при разработке демонстрационной экспертной системы.

Задачи

1. Изучить базовые команды и конструкции CLIPS: запустить систему CLIPS, активизировать окно просмотра текущего списка фактов, выполнить следующую последовательность действий, фиксируя после каждого шага состояние списка фактов:

- сбросить систему в исходное состояние командой (clear);
- выполнить начальную установку командой (reset);
- ввести 3 любых упорядоченных факта командой (assert);
- повторно выполнить сброс командой (reset);
- установить 3 ранее вводимых упорядоченных факта в качестве исходных фактов, используя конструкцию (deffacts);
- выполнить сброс командой (reset).

Затем активизировать дополнительно окно просмотра агенды и выполнить последовательность действий по созданию правил, фиксируя после каждого шага состояния списка фактов и агенды.

2. Разработать демонстрационную экспертную систему. Для этого необходимо: предложить предметную область демонстрационной экспертной системы; сформировать, пользуясь редактором CLIPS, базу знаний демонстрационной ЭС и сохранить ее в файле rulebase.clp. Общее количество правил в БЗ должно быть не менее 25. Количество значений переменных должно выбираться таким образом, чтобы БЗ отвечала требованию полноты, т.е. содержала правила, соответствующие любым сочетаниям значений переменных в левых частях правил; для активизации ЭС в среде CLIPS использовать пакетный файл Run_Lab_3.bat; протестировать ЭС на различных комбинациях входных значений в пошаговом режиме.

Результаты знакомства со средой CLIPS

На рисунках 1-6 зафиксированы состояния списка фактов (Facts Window), соответствующие указанным действиям. На рисунках 7-9 – состояния списка фактов и агенды.

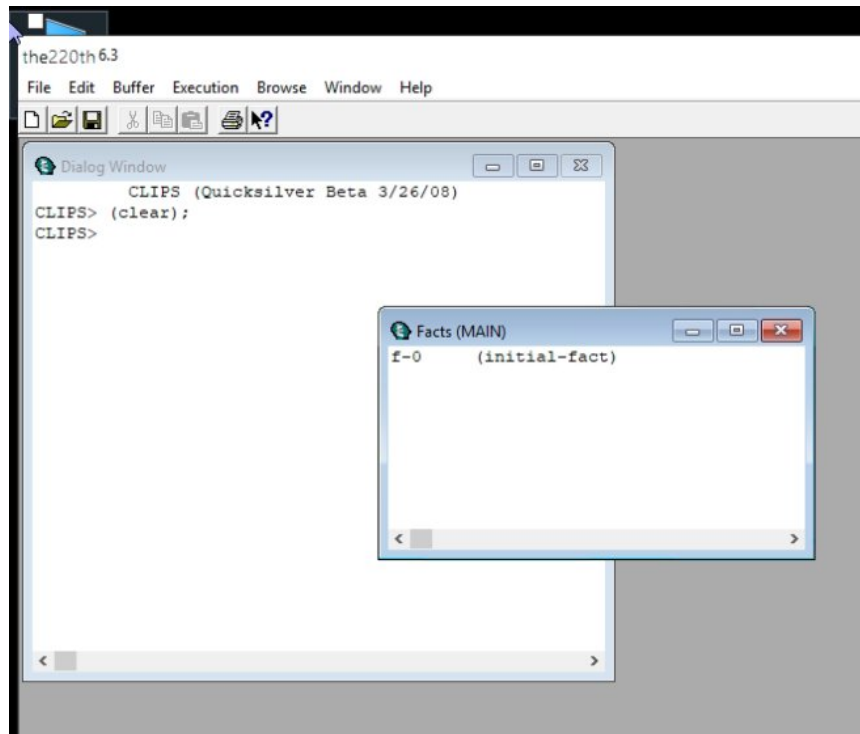


Рисунок 1 – Сброс системы исходное состояние командой (clear)

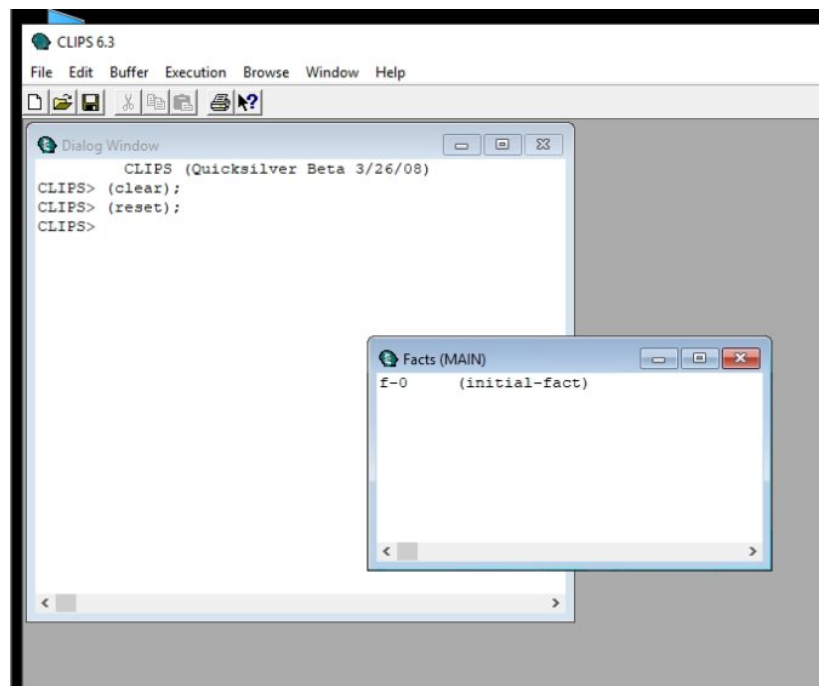


Рисунок 2 – Начальная установка командой (reset)

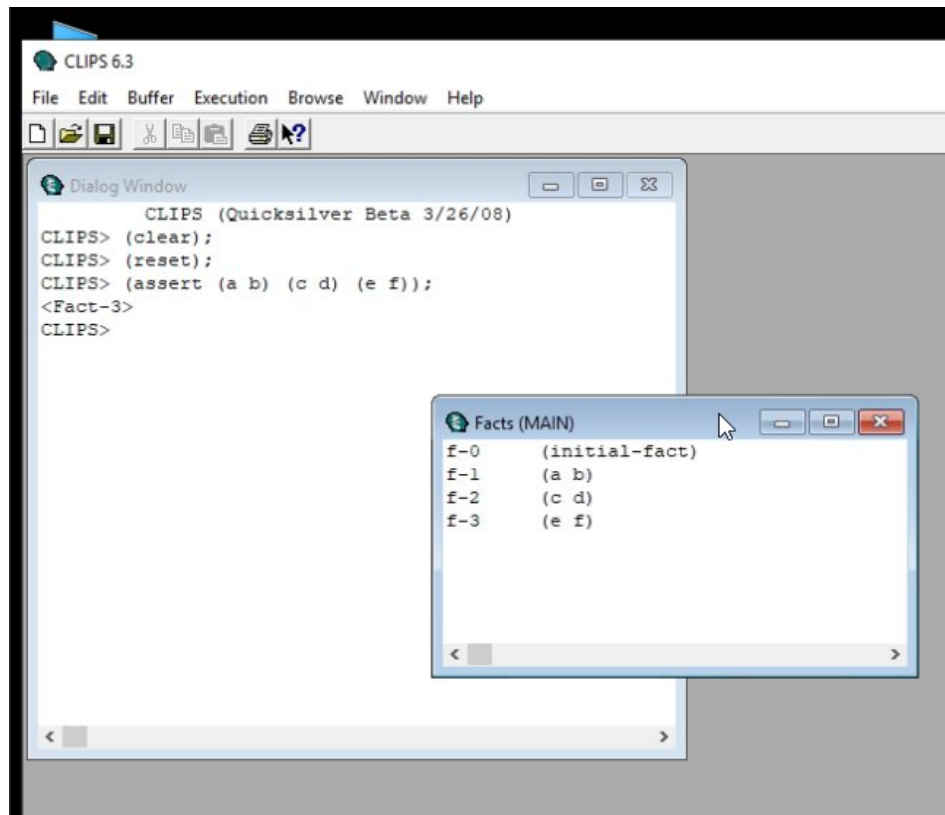


Рисунок 3 – Выполнение команды (assert)

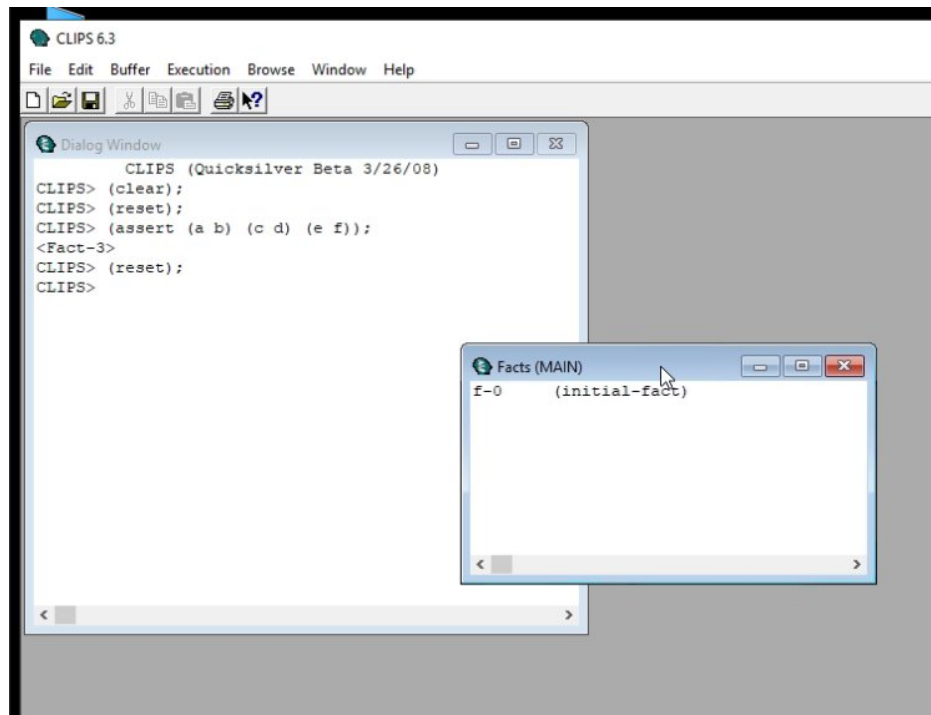


Рисунок 4 – Повторное выполнение сброса командой (reset)

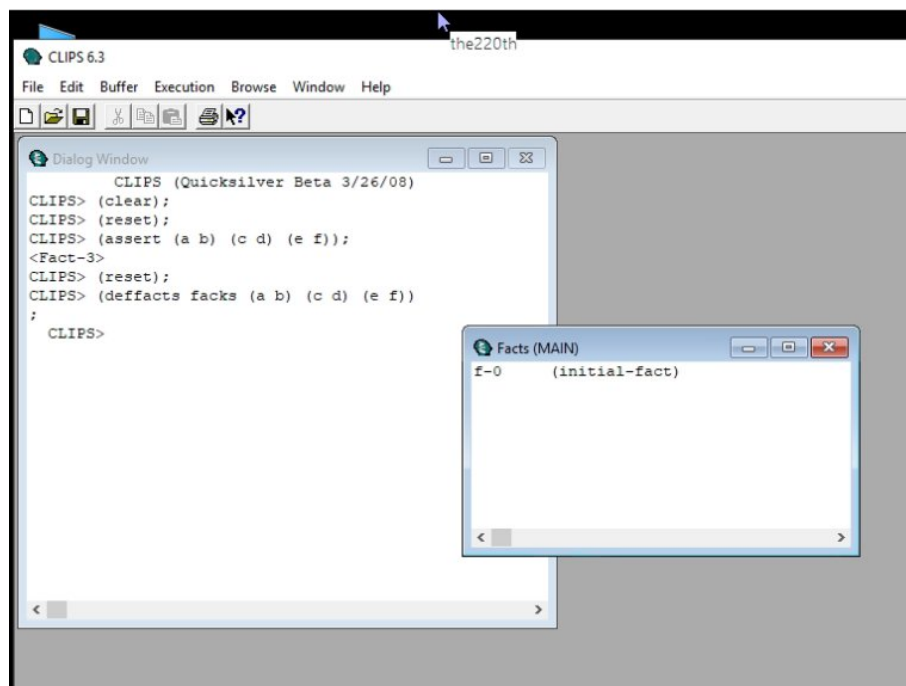


Рисунок 5 – Использование конструкции (deffacts)

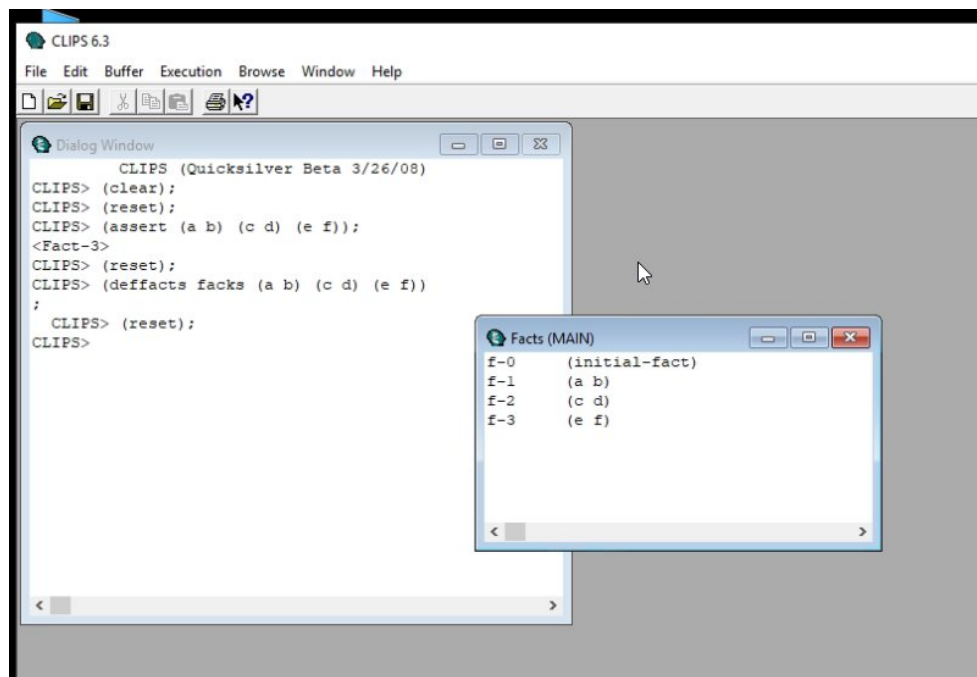


Рисунок 6 – Выполнение сброса командой (reset)

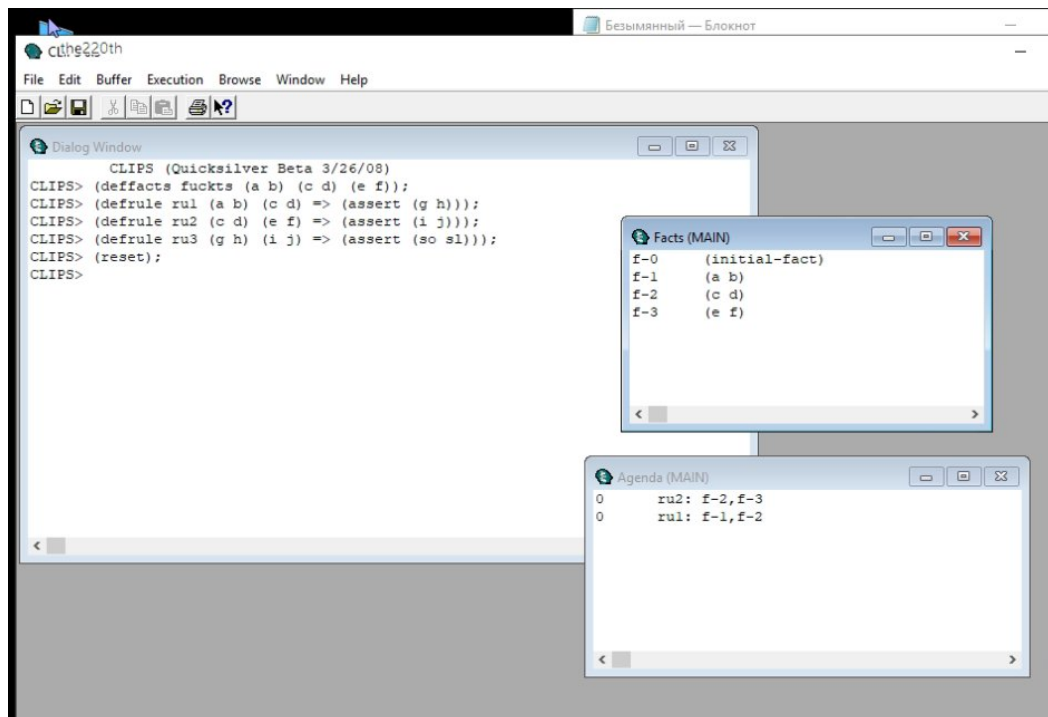


Рисунок 7 – Использование конструкции (defrule)

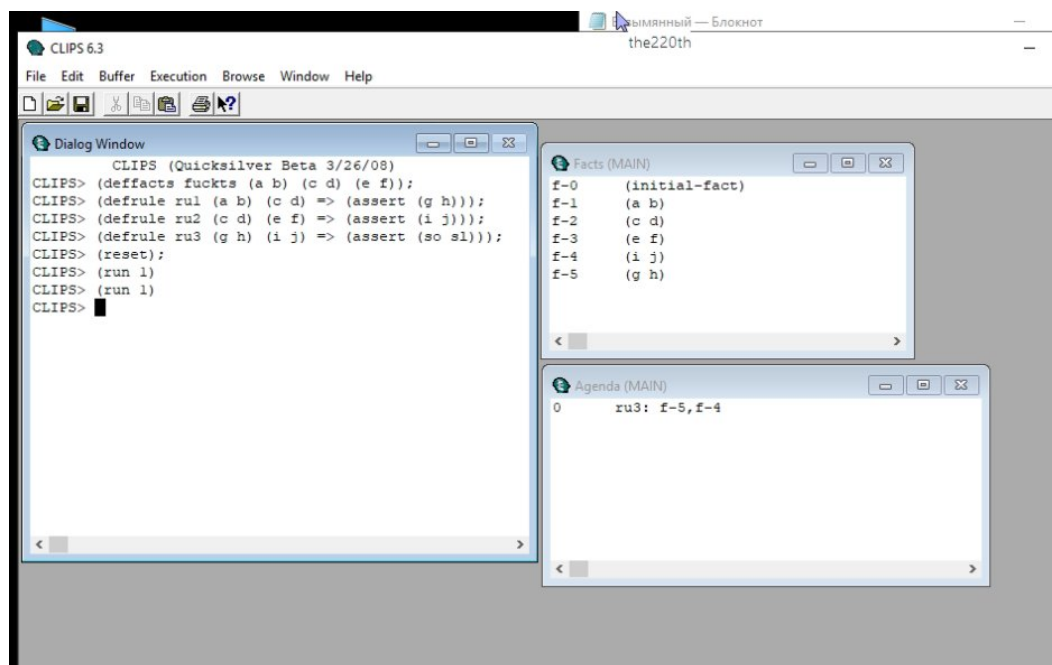


Рисунок 8 – Активизация правил 1

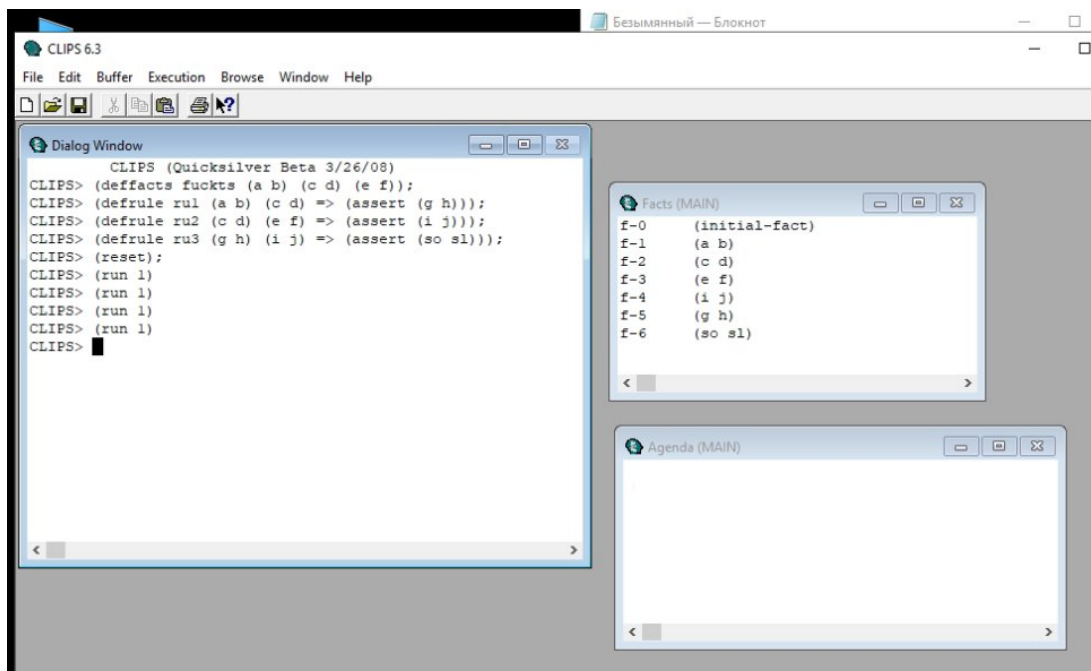


Рисунок 9 – Активизация правил 2

В результате было произведено знакомство с интерактивной средой CLIPS и основными возможностями CLIPS.

Описание выбранной предметной области

ЭС должна определять нужный процессор для пользователя, должна иметь четыре входные переменные (тип работ, теплоотвод, цена, тип установки) две промежуточные переменные (стоимость и энергопотребление) и выходную переменную (название процессора). Диаграмма зависимости переменных представлена на рисунке 10.

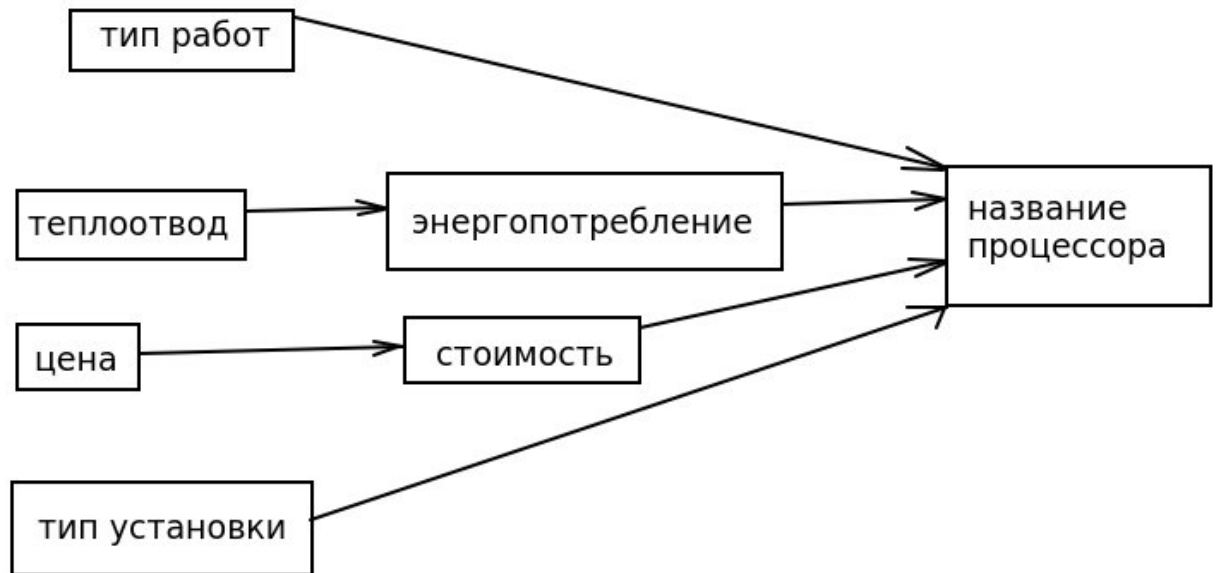
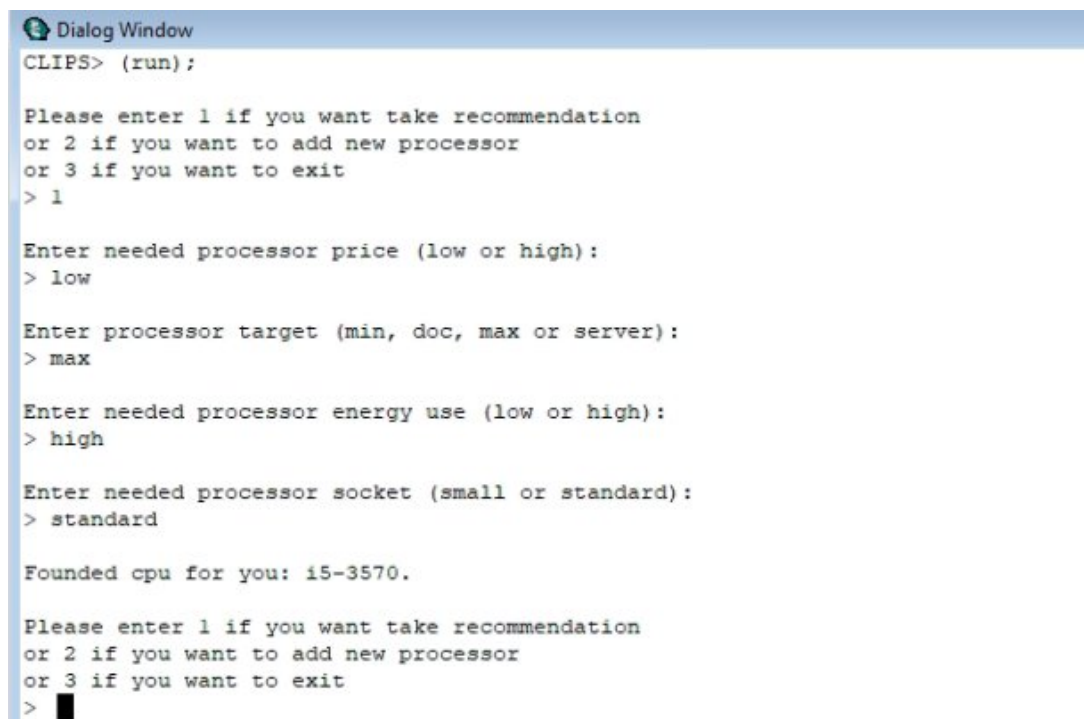


Рисунок 10 – Диаграмма зависимости переменных

Пример работы программы

Изначально в базе знаний есть информация о процессорах, но общаясь с пользователем, программа может дополнить базу. Примеры протоколов выполнения представлены на рисунках 11, 12, 13 и 14.



```
Dialog Window
CLIPS> (run);

Please enter 1 if you want take recommendation
or 2 if you want to add new processor
or 3 if you want to exit
> 1

Enter needed processor price (low or high):
> low

Enter processor target (min, doc, max or server):
> max

Enter needed processor energy use (low or high):
> high

Enter needed processor socket (small or standard):
> standard

Founded cpu for you: i5-3570.

Please enter 1 if you want take recommendation
or 2 if you want to add new processor
or 3 if you want to exit
> █
```

Рисунок 11 – Пример протокола выполнения 1

```
Dialog Window
Founded cpu for you: i5-3570.

Please enter 1 if you want take recommendation
or 2 if you want to add new processor
or 3 if you want to exit
> 1

Enter needed processor price (low or high):
> low

Enter processor target (min, doc, max or server):
> max

Enter needed processor energy use (low or high):
> low

Enter needed processor socket (small or standard):
> standard

Not founded cpu for you. =C

Please enter 1 if you want take recommendation
or 2 if you want to add new processor
or 3 if you want to exit
> █
< █
```

Рисунок 12 – Пример протокола выполнения 2

```
Dialog Window

Not founded cpu for you. =C

Please enter 1 if you want take recommendation
or 2 if you want to add new processor
or 3 if you want to exit
> 2

Enter processor name:
> i-super proc228HUK

Enter processor price (rub):
> 1337

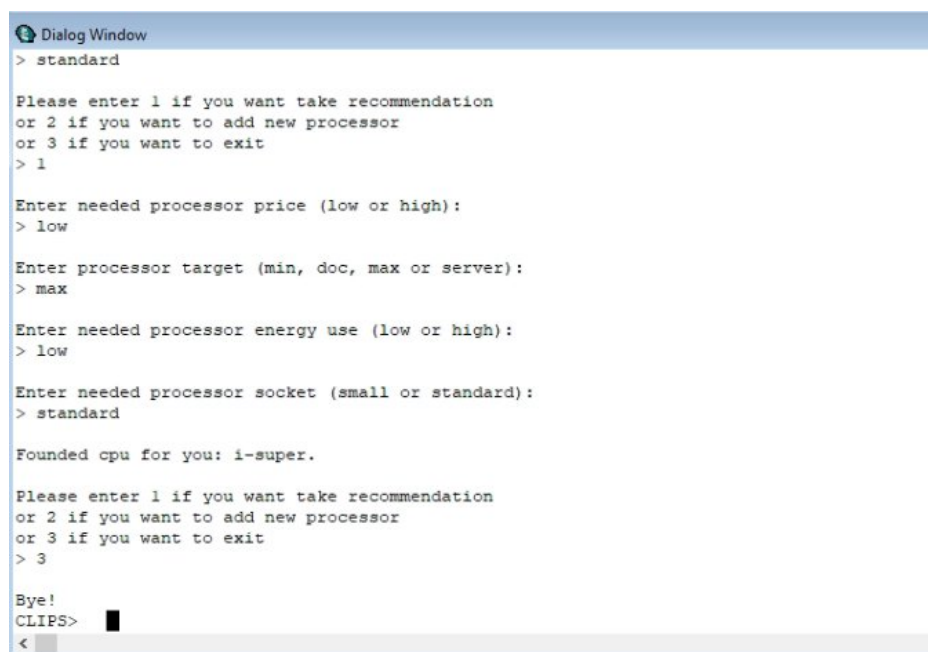
Enter processor target (min, doc, max or server):
> max

Enter processor TDP (watt):
> 9

Enter processor socket (small or standard):
> standard

Please enter 1 if you want take recommendation
or 2 if you want to add new processor
or 3 if you want to exit
> █
< █
```

Рисунок 13 – Пример протокола выполнения 3



```
Dialog Window
> standard

Please enter 1 if you want take recommendation
or 2 if you want to add new processor
or 3 if you want to exit
> 1

Enter needed processor price (low or high):
> low

Enter processor target (min, doc, max or server):
> max

Enter needed processor energy use (low or high):
> low

Enter needed processor socket (small or standard):
> standard

Founded cpu for you: i-super.

Please enter 1 if you want take recommendation
or 2 if you want to add new processor
or 3 if you want to exit
> 3

Bye!
CLIPS> █
< █
```

Рисунок 14 – Пример протокола выполнения 4

Текст программы ЭС представлен в приложении.

Вывод

В ходе выполнения лабораторной работы были изучены азы работы в среде разработки экспертной системы CLIPS, познакомились со способами представления знаний, с основными типами данных, с командами для работы с фактами, со способами задания условий по обработке фактов и со способами организации правил.

Использованные источники

1. Рассел С, Норвиг П. Искусственный интеллект: современный подход, 2-е изд., М. «Вильямс», 2006.
2. Лекция “CLIPS - среда разработки ЭС” по предмету “Введение в искусственный интеллект”.

Приложение

Исходный текст программы:

; Шаблон процессора

```
(deftemplate cpu_struct "CPU struct"
```

; Название процессора

```
(slot name (type STRING))
```

; Тип работ: min-для микрозадач (тонкий клиент), doc - работа с документами, max - любая работа, server - для сервера

```
(slot work (type STRING) (allowed-values "min" "doc" "max" "server"))
```

; Энергопотребление

```
(slot W (type STRING) (allowed-values "low" "high"))
```

; Стоимость

```
(slot price (type STRING) (allowed-values "low" "high"))
```

; Тип установки

```
(slot size (type STRING) (allowed-values "small" "standard"))
```

```
);
```

; Изначальная "база знаний"

```
(deffacts initial
```

```
(cpu_struct (name "allwinner H5 cortex-A53") (work "min") (W "low") (price "low") (size "small"));
```

```
(cpu_struct (name "Z-01") (work "min") (W "low") (price "high") (size "small"));
```

```
(cpu_struct (name "sempron M140") (work "min") (W "high") (price "low") (size "small"));
```

```
;(cpu_struct (name "") (work "min") (W "high") (price "high") (size "small"));
```

```
(cpu_struct (name "celeron j1800") (work "min") (W "low") (price "low") (size "standard"));
```

```

;(cpu_struct (name "") (work "min") (W "low") (price "high") (size "standard"));

(cpu_struct (name "celeron 440") (work "min") (W "high") (price "low") (size "standard"));

(cpu_struct (name "celeron 400") (work "min") (W "high") (price "high") (size "standard"));


(cpu_struct (name "core 2 Duo E8500") (work "doc") (W "high") (price "low") (size "standard"));

(cpu_struct (name "e2-3800") (work "doc") (W "low") (price "low") (size "standard"));

(cpu_struct (name "Pentium 4 3.0") (work "doc") (W "high") (price "high") (size "standard"));

;(cpu_struct (name "") (work "doc") (W "low") (price "high") (size "standard"));

(cpu_struct (name "celeron 847") (work "doc") (W "low") (price "low") (size "small"));

(cpu_struct (name "i5-4300U") (work "doc") (W "high") (price "high") (size "small"));

(cpu_struct (name "pentium Gold 5405U") (work "doc") (W "low") (price "high") (size "small"));

(cpu_struct (name "i3-2350M") (work "doc") (W "high") (price "low") (size "small"));


(cpu_struct (name "i5-3570") (work "max") (W "high") (price "low") (size "standard"));

(cpu_struct (name "i5-9400F") (work "max") (W "high") (price "high") (size "standard"));

;(cpu_struct (name "") (work "max") (W "low") (price "low") (size "standard"));

;(cpu_struct (name "") (work "max") (W "low") (price "high") (size "standard"));

;(cpu_struct (name "") (work "max") (W "high") (price "low") (size "small"));

(cpu_struct (name "i9-8950HK") (work "max") (W "high") (price "high") (size "small"));

(cpu_struct (name "Broadcom BCM2711 ARM Cortex-A72") (work "max") (W "low") (price "low") (size "small"));

(cpu_struct (name "Apple M1") (work "max") (W "low") (price "high") (size "small"));


(cpu_struct (name "neoverse E1") (work "server") (W "low") (price "low") (size "small"));

(cpu_struct (name "atom C2518") (work "server") (W "low") (price "high") (size "small"));

(cpu_struct (name "pentium D-1507") (work "server") (W "high") (price "low") (size "small"));

(cpu_struct (name "atom c2758") (work "server") (W "high") (price "high") (size "small"));

```

```
; (cpu_struct (name "") (work "server") (W "low") (price "low") (size "standard"));
(cpu_struct (name "xeon E3-1105C") (work "server") (W "low") (price "high") (size "standard"));
(cpu_struct (name "xeon E5-2640") (work "server") (W "high") (price "low") (size "standard"));
(cpu_struct (name "epyc 7351P") (work "server") (W "high") (price "high") (size "standard"));
);
```

```
; Начальный цикл
(defrule data-input
(initial-fact)
=>
(assert (user_want 0));
);
```

```
(defrule R_circle_start
(user_want 0)
?f_addr <- (user_want 0)
=>
(printout t crlf "Please enter 1 if you want take recommendation" crlf
              "or 2 if you want to add new processor" crlf
              "or 3 if you want to exit" crlf "> ");
(bind ?user_want (read));
(assert (user_want ?user_want));
(retract ?f_addr);
);
```

```
; Если нужно выйти из программы
(defrule R_bye
```



```
(user_want 3)
```

```
=>
```

```
(printout t crlf "Bye! " crlf);
```

```
);
```

; Если пользователь решил дополнить базу

```
(defrule R_if1_2
```

```
(user_want 2)
```

```
=>
```

```
(printout t crlf "Enter processor name: " crlf "> ");
```

```
(bind ?buff1 (str-cat (read)));
```

```
(assert (new_proc_name ?buff1));
```

```
(printout t crlf "Enter processor price (rub): " crlf "> ");
```

```
(bind ?buff2 (read));
```

```
(assert (new_proc_price_rub ?buff2));
```

```
(printout t crlf "Enter processor target (min, doc, max or server): " crlf "> ");
```

```
(bind ?buff3 (str-cat (read)));
```

```
(assert (new_proc_work ?buff3));
```

```
(printout t crlf "Enter processor TDP (watt): " crlf "> ");
```

```
(bind ?buff4 (read));
```

```
(assert (new_proc_W_W ?buff4));
```

```
(printout t crlf "Enter processor socket (small or standard): " crlf "> ");
```

```
(bind ?buff5 (str-cat (read)));
```

```
(assert (new_proc_size ?buff5));  
);
```

; Перевод числовой цены в категорию стоимости

```
(defrule R_if2_1  
(user_want 2)  
(new_proc_price_rub ?x)  
(test (< ?x 15000))  
?f_addr <- (new_proc_price_rub ?x)  
=>  
(assert (new_proc_price "low"));  
(retract ?f_addr);  
);
```

; Перевод числовой цены в категорию стоимости

```
(defrule R_if2_2  
(user_want 2)  
(new_proc_price_rub ?x)  
(test (>= ?x 15000))  
?f_addr <- (new_proc_price_rub ?x)  
=>  
(assert (new_proc_price "high"));  
(retract ?f_addr);  
);
```

; Перевод числового теплоотвода в категорию энергопотребления

```
(defrule R_if3_1
```

```

(user_want 2)
(new_proc_W_W ?x)
(test (< ?x 40))
?f_addr <- (new_proc_W_W ?x)
=>
(assert (new_proc_W "low"));
(retract ?f_addr);
);

```

; Перевод числового теплоотвода в категорию энергопотребления

```

(defrule R_if3_2
(user_want 2)
(new_proc_W_W ?x)
(test (>= ?x 40))
?f_addr <- (new_proc_W_W ?x)
=>
(assert (new_proc_W "high"));
(retract ?f_addr);
);

```

; Занесение нового процессора в базу

```

(defrule R_new_proc
(user_want 2)
(new_proc_name ?x_name)
(new_proc_price ?x_price)
(new_proc_work ?x_work)
(new_proc_W ?x_W)

```

```

(new_proc_size ?x_size)
?f_addr_0 <- (user_want 2)
?f_addr_1 <- (new_proc_name ?x_name)
?f_addr_2 <- (new_proc_price ?x_price)
?f_addr_3 <- (new_proc_work ?x_work)
?f_addr_4 <- (new_proc_W ?x_W)
?f_addr_5 <- (new_proc_size ?x_size)
=>
(assert (cpu_struct (name ?x_name) (work ?x_work) (W ?x_W) (price ?x_price) (size ?x_size)));
(retract ?f_addr_1 ?f_addr_2 ?f_addr_3 ?f_addr_4 ?f_addr_5 ?f_addr_0);
(assert (user_want 0));
);

```

; Если пользователь хочет получить рекомендацию

```

(defrule R_if1_1
(user_want 1)
=>

```

```

(printout t crlf "Enter needed processor price (low or high): " crlf "> ");
(bind ?buff2 (str-cat (read)));
(assert (user_proc_price ?buff2));

```

```

(printout t crlf "Enter processor target (min, doc, max or server): " crlf "> ");
(bind ?buff3 (str-cat (read)));
(assert (user_proc_work ?buff3));

```

```
(printout t crlf "Enter needed processor energy use (low or high): " crlf "> ");
(bind ?buff4 (str-cat (read)));
(assert (user_proc_W ?buff4));
```

```
(printout t crlf "Enter needed processor socket (small or standard): " crlf "> ");
(bind ?buff5 (str-cat (read)));
(assert (user_proc_size ?buff5));
);
```

; Рекомендация пользователю

```
(defrule R_predict
(user_want 1)
(user_proc_price ?x_price)
(user_proc_work ?x_work)
(user_proc_W ?x_W)
(user_proc_size ?x_size)
(cpu_struct (name ?x_name) (work ?x_work) (W ?x_W) (price ?x_price) (size ?x_size))
?f_addr <- (user_want 1)
=>
(printout t crlf "Founded cpu for you: " ?x_name ". " crlf);
(retract ?f_addr);
(assert (user_want 0));
);
```

; Если запросу пользователя не найти процессор из "базы знаний"

```
(defrule R_predict_not
(user_want 1)
```

```

(user_proc_price ?x_price)
(user_proc_work ?x_work)
(user_proc_W ?x_W)
(user_proc_size ?x_size)
(not(exists(cpu_struct (name ?x_name) (work ?x_work) (W ?x_W) (price ?x_price) (size ?x_size))))
?f_addr <- (user_want 1)
=>
(printout t crlf "Not founded cpu for you. =C" crlf);
(retract ?f_addr);
(assert (user_want 0));
);

```