Московский Государственный Технический Университет имени Н. Э. Баумана
Факультет «Информатика и системы управления»
Кафедра «Компьютерные системы и сети»

Е.К. Пугачев

УТВЕРЖДАЮ
Зав. кафедрой ИУ6

д.т.н., проф. _____ Сюзев В.В.
"____" _____2013 г.

Исследование методов представления и обработки знаний.

Методические указания

по выполнению лабораторных работ

по дисциплине "Системы искусственного интеллекта".

Часть 2

Москва 2013

Введение

Существует класс задач, решение которых невозможно традиционными формальными методами. Этот класс задач определен как интеллектуальные задачи. К ним относятся: задачи интерпретации, диагностики, контроля, прогнозирования, планирования, проектирования и т.д. Особенностью таких задач является сложность выбора однозначно оптимального решения.

К настоящему времени разработан ряд методов и средств искусственного интеллекта (ИИ), которые можно применить при решении вышеперечисленных задач.

Начало первого этапа развития работ в области искусственного интеллекта приходится на конец 60-х годов. Новая отрасль индустрии - производство интеллектуальных систем сформировалась в 80-е годы.

Можно выделить следующие основные черты интеллектуальных систем:

* наличие человека, который взаимодействует с ней;

* возможность накапливать и получать новые знания;

* способность осуществлять рассуждения, характерные для человека.

Основная проблема при создании систем ИИ состоит в том, что во многих областях человеческой деятельности объекты, с которыми оперируют специалисты, не могут быть сведены к чисто синтаксическим объектам, характерным для математики.

Экспертные системы - это самостоятельное направление в области искусственного интеллекта. Мощность ЭС обусловлена в первую очередь мощностью базы знаний (БЗ) и возможностью ее пополнения, а во вторую очередь - используемыми методами. Опыт показал, что важнее иметь разнообразные специальные знания, а не общие процедуры вывода. Однако, представляемые знания экспертов в основном эвристические, эмпирические и неопределенные.

Одним из важных требований является возможность взаимодействия пользователя с ЭС. При этом предъявляются следующие требования: общение на языке близком к пользователю; линия рассуждения должна быть понятна пользователю; способность ЭС объяснять свои выводы.

При проектировании и разработке ЭС необходимо ответить на ряд следующих вопросов:

1. Какие способы представления знаний лучше всего использовать ?

2. Какой механизм логического вывода лучше подходит для получения наиболее правильного решения ?

3. Как пользователи различной квалификации должны взаимодействовать с ЭС ?

4. Какие сервисные функции должны быть реализованы в ЭС ?

5. Какие инструментальные средства лучше всего использовать при реализации ЭС и т.д.?

Краткие теоретические сведения.

В системах, основанных на знаниях, правила (или эвристики), по которым решаются проблемы в конкретной предметной области, хранятся в базе знаний. Проблемы ставятся перед системой в виде совокупности фактов, описывающих некоторую ситуацию, и система с помощью базы знаний пытается вывести заключение из этих фактов. Эвристики представляют собой правила вывода, которые позволяют находить решения по известным фактам. Обобщенная схема функционирования системы, основанной на знаниях представлена на рис. 1.

Рис. 1. Обобщенная схема функционирования системы, основанной на знаниях

Структура, например, экспертной системы продукционного типа может включать пять модулей (см. рис. 2).

В общем случае знания в такой системе разделяются на три типа:

Декларативные знания (факты). Этот вид знаний представляет собой факты о конкретных ситуациях. Такие факты могут быть описаны заранее и включены в базу знаний на этапе создания ЭС. К декларативным знаниям можно отнести факты, которые собираются в процессе диалога с пользователем непосредственно во время работы ЭС. Структура представления данного вида информации, а также способы ее обработки (считывание, модификация и т.п.) имеет важное значение для организации всех модулей ЭС.

2. **Процедурные знания (правила).** Обычно эти знания собираются заранее путем опроса экспертов в конкретной предметной области и составляют ядро базы знаний. На их основе строится механизм логического вывода. Процедурные знания непосредственно

связаны с декларативными, позволяют обрабатывать имеющиеся в базе знаний факты, а при необходимости генерировать новые факты.

3. **Управляющие знания**. В ЭС должен быть предусмотрен некоторый набор стратегий, чтобы можно было рассматривать альтернативные возможности получения вывода в время работы, т.е. переходить при неудаче от одной стратегии к другой. Управляющие знания определяют, какие из процедурных правил следует использовать для получения вывода. По существу данные знания составляют основу механизма логического вывода.

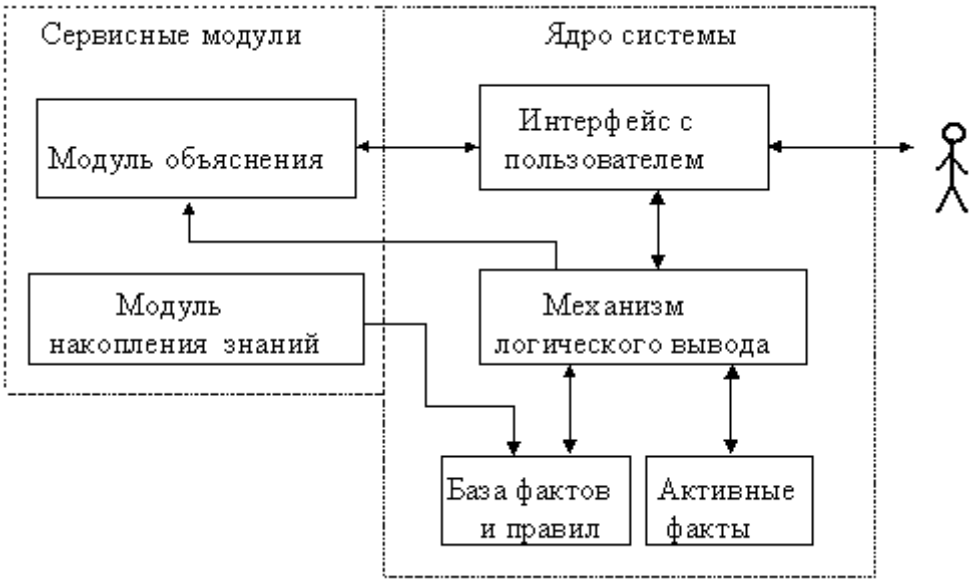


Рис. 2. Структура экспертной системы продукционного типа.

Механизм логического вывода (МЛВ) выполняет следующие функции:

* формирование и обработка активных фактов конкретной ситуации;
* определение порядка выбора и применения фактов и правил.

МЛВ можно представить в виде четырех последовательных процессов:

* выбор активных правил и фактов;
* сопоставление (определяется какие правила выполнять в первую очередь);
* разрешение конфликтов;
* выполнение выбранного означенного правила (действие).

В общем случае выделяют два порядка механизма вывода: прямой и обратный. Управление прямым выводом осуществляется проще, чем управление обратным выводом. Прямой порядок вывода строится от активных фактов к заключению, т.е. по известным фактам отыскивается заключение, которое из этих фактов следует. При обратном порядке вывода заключения просматриваются последовательно до тех пор, пока не будут обнаружены факты конкретной ситуации, подтверждающие какое либо из заключений, т.е. путем подбора подходящих фактов под имеющееся заключение.

В некоторых ЭС вывод основывается на сочетании вышеприведенных подходов - обратного и ограниченного прямого. Такой комбинированный метод получил название циклического.

В ЭС, база знаний которой насчитывает сотни правил возникает необходимость использования некоторой стратегии управления выводом, позволяющей структурировать процесс вывода и минимизировать время поиска решения.

К числу таких стратегий относятся:

* поиск в глубину;
* поиск в ширину:
* разбиение на подзадачи;
* альфа-бета алгоритм и др.

Идея поиска в глубину состоит в том, что при выборе очередной подцели в пространстве состояний предпочтение стремятся отдать той, которая соответствует следующему, более детальному, уровню описания задачи. Такой механизм встроен в систему Пролог. Поиск в глубину наиболее адекватен рекурсивному стилю программирования. Обрабатывая цели, пролог-система сама просматривает альтернативы именно в глубину.

Пример простого пространства состояний, в котором осуществляется поиск в глубину представлен на рис. 3.

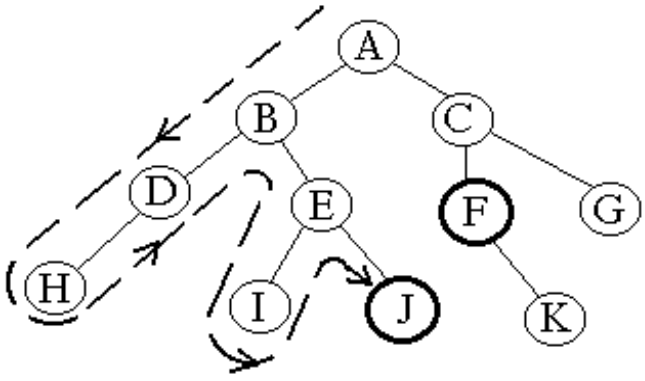


Рис. 3. Поиск в глубину.

В противоположность поиску в глубину стратегия поиска в ширину предусматривает переход в первую очередь к подцели того же уровня. На рис. 4 представлена графическая иллюстрация данной стратегии.

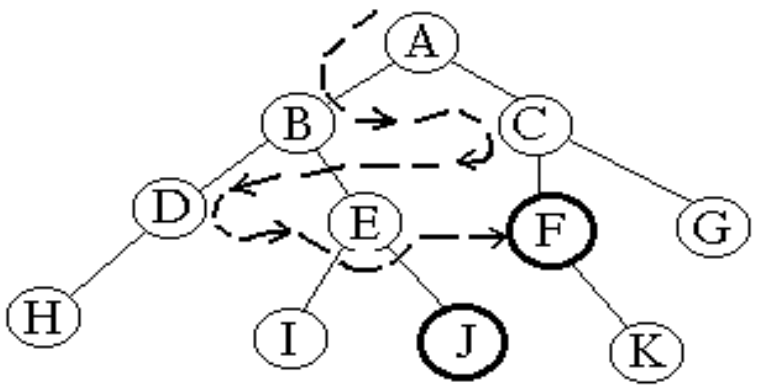


Рис. 4. Поиск в ширину.

Например, при поиске в глубину ЭС, сделав на основе известных симптомов предположение о наличии определенного заболевания, будет продолжать запрашивать уточняющие признаки и симптомы этой болезни до тех пор, пока полностью не отвергнет выдвинутую гипотезу. При поиске в ширину, напротив, система в начале проанализирует все симптомы, находящиеся на одном уровне пространства состояний, даже если они относятся к разным заболеваниям, и лишь затем перейдет к симптомам следующего уровня детальности.

Стратегия разбиение на подзадачи состоит в том, что в исходной задаче выделяют подзадачи, решение которых рассматривается как достижение промежуточных целей на пути к конечной цели. Такая стратегия хорошо зарекомендовала себя в диагностической ЭС определения неисправности в автомобиле. Вначале выявляется отказавшая подсистема (например, система электропитания), затем на следующем уровне уточняется (например, неисправен аккумулятор) и на последнем шаге выдается причина неисправности (например, нет контакта или аккумулятор разряжен).

Альфа-бета алгоритм. Задача сводится к уменьшению пространства состояний путем удаления в нем ветвей, не перспективных для поиска успешного решения. Поэтому просматриваются только те вершины, в которые можно попасть в результате следующего шага, после чего неперспективные направления исключаются из дальнейшего рассмотрения. Например, если цвет объекта, который мы ищем не красный, то его бессмысленно искать среди красных объектов. Такой подход нашел широкое применение в играх, в шахматных системах. Данный подход используется для повышения эффективности поиска решений в продукционных системах.

Интерфейс с пользователем отвечает за обмен информацией между пользователем и экспертной системой.

Экспертная система может быть ориентирована на разные типы пользователей. Но независимо от того, является ли пользователь специалистом или нет, всех их объединяет следующее: языком общения является ограниченный естественный язык, а не формальный язык программирования.

В интерфейсном модуле могут использоваться все известные формы диалога:

* *Директивная* . Словарь данной формы состоит из ключевых слов на естественном языке, сокращений, чисел и мнемокодов.

* *Табличная*. Включает конкретные типы : - *выбор операции для исполнения по меню;* - *заполнение и редактирование шаблона (*шаблон-вектор и шаблон-таблица);

* - *Фразовая.* Использует ограниченно естественный язык.

Фразовая форма используются во многих существующих ЭС, но достаточно на примитивном уровне. Допустимые входные сообщения пользователя ограничены набором понятий, содержащихся в базе знаний.

Перспективным направлением является разработка ЭС ориентированных на пользователя не специалиста, которые могут общаться с системой с помощью полных предложений, включающих в себя любые части речи.

Для уменьшения времени набора фраз могут применяться сокращения, шаблоны фраз, программируемые клавиши ключевых слов и меню.

Существуют следующие основные подходы, используемые для обработки фраз:

- Морфологический анализ - обработка словоформ(отрезок текста между двумя соседними пробелами) вне связи с контекстом.

Выделяют два метода :

-*декларативный* - в словаре находятся все возможные слово формы каждого слова и анализ сводится к поиску словоформы словаре. При использовании данного метода обеспечивают возможность обработки сообщений , состоящих из строчных и прописных букв в произвольной комбинации, причем как латинского, так и русского алфавитов (или других алфавитов);

-*процедурный* - выделяется в текущей словоформе основа, которая затем идентифицируется.

- Синтаксический анализ - используя информацию полученную после морфологического анализа строится синтаксическая структура входного сообщения т.е. осуществляется разбор предложения.
- Семантический анализ - определение смысловых отношений между словоформами (выявляются главные предикаты).Применительно к данной форме можно выделить следующие функции интерфейса:

- осуществлять преобразование сообщения из естественно-языковой формы в форму внутреннего представления и обратное преобразование;

- анализ и синтез сообщений пользователя и системы;

- отслеживать и запоминать пройденный путь диалога.

Обобщенная схема модуля интерфейса с пользователем приведена на рис. 5.

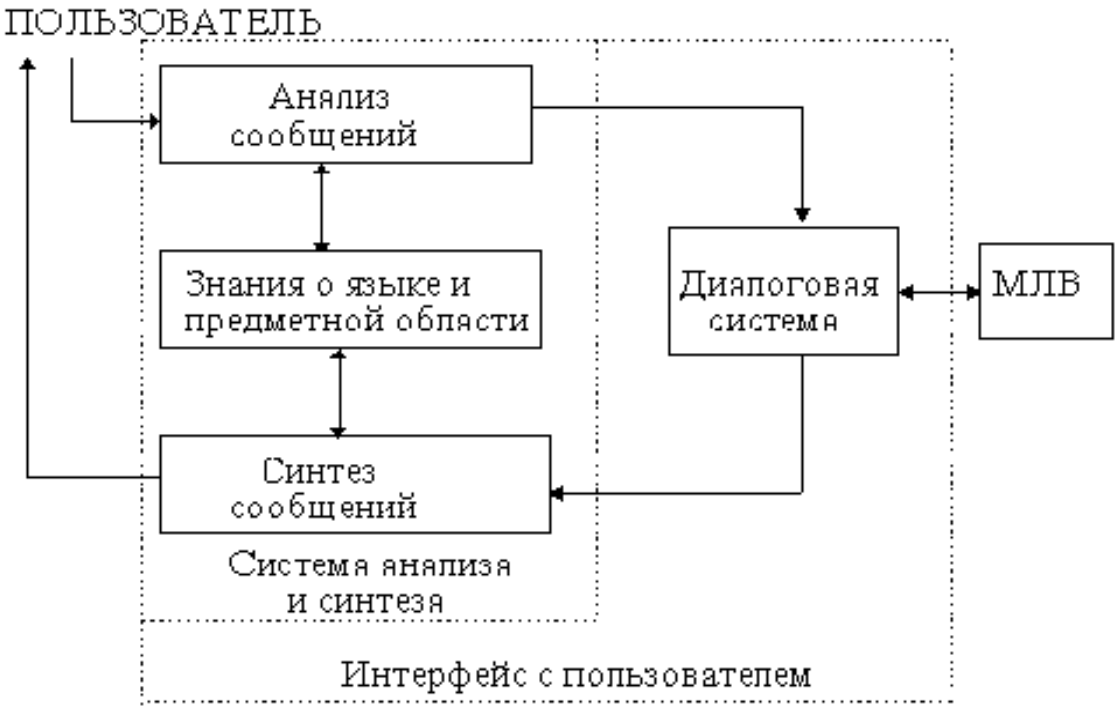


Рис. 5. Обобщенная схема модуля интерфейса с пользователем.

Задача системы анализа и синтеза состоит в обработке отдельных сообщений системы и пользователя. Сложность методов анализа и синтеза зависит как от языка общения, так и от языка, используемого для представления знаний.

Например, на этапе консультации язык общения может быть строго формализован фиксированным набором запросов системы и множеством возможных ответов пользователя. Здесь задача синтеза сводится к генерации подготовленных заранее вопросов, а задача анализа к обработке слов и словосочетаний с помощью морфологического анализа.

Задача синтеза на этапе приобретения знаний и объяснения в существующих системах сводится к использованию шаблонов или заранее заготовленных сообщений.

Модуль объяснения.

Наряду с понятием «модуль объяснений» (МО) используются другие понятия : «блок объяснений», «подсистема объяснений» и др.

Выделяют следующие основные положения, касающиеся функций и назначения модуля объяснений.

Экспертная система должна уметь объяснять свое поведение и свои решения пользователю так же, как это делает эксперт-человек.

Без механизма объяснений пользователь не доверяет полученным результатам и ЭС не будет иметь спроса.

Назначение модуля объяснений - сделать ЭС «прозрачной» для пользователя, т.е. предоставить пользователю возможность понимать логику действий ЭС, дать надежную гарантию правильности полученных результатов.

В настоящее время применительно к МО выделяют пять типов объяснений:

* причинные объяснения (вскрывают причинные взаимосвязи между явлениями);

* объяснения на основе теоретических законов;

* функциональные объяснения (сводятся к установлению функций, выполняемых определенной частью системы);

* структурное объяснение (используют описание структуры, которая обеспечивает выполнение функций и поведение объясняемой системы в целом);

* историческое объяснение (раскрывает условия, причины и законы, которые привели к текущему состоянию системы).

Для достижения наибольшего эффекта от использования МО в структуре ЭС целесообразно использовать все известные типы объяснений. Задача состоит в том, как и в каких случаях использовать тот или иной тип объяснения.

Процессу объяснения можно дать следующее определение: “Объяснить - сделать ясным, ответить на вопросы: “Как ?”, “Почему ?”, “Что следует предпринять далее ?”, “Что будет если ?”, а также обосновать , подтвердить правильность результата.

Все возможные вопросы, возникающие у пользователя в процессе диалога с ЭС, классифицированы на два вида:

* вопросы о действиях ЭС;

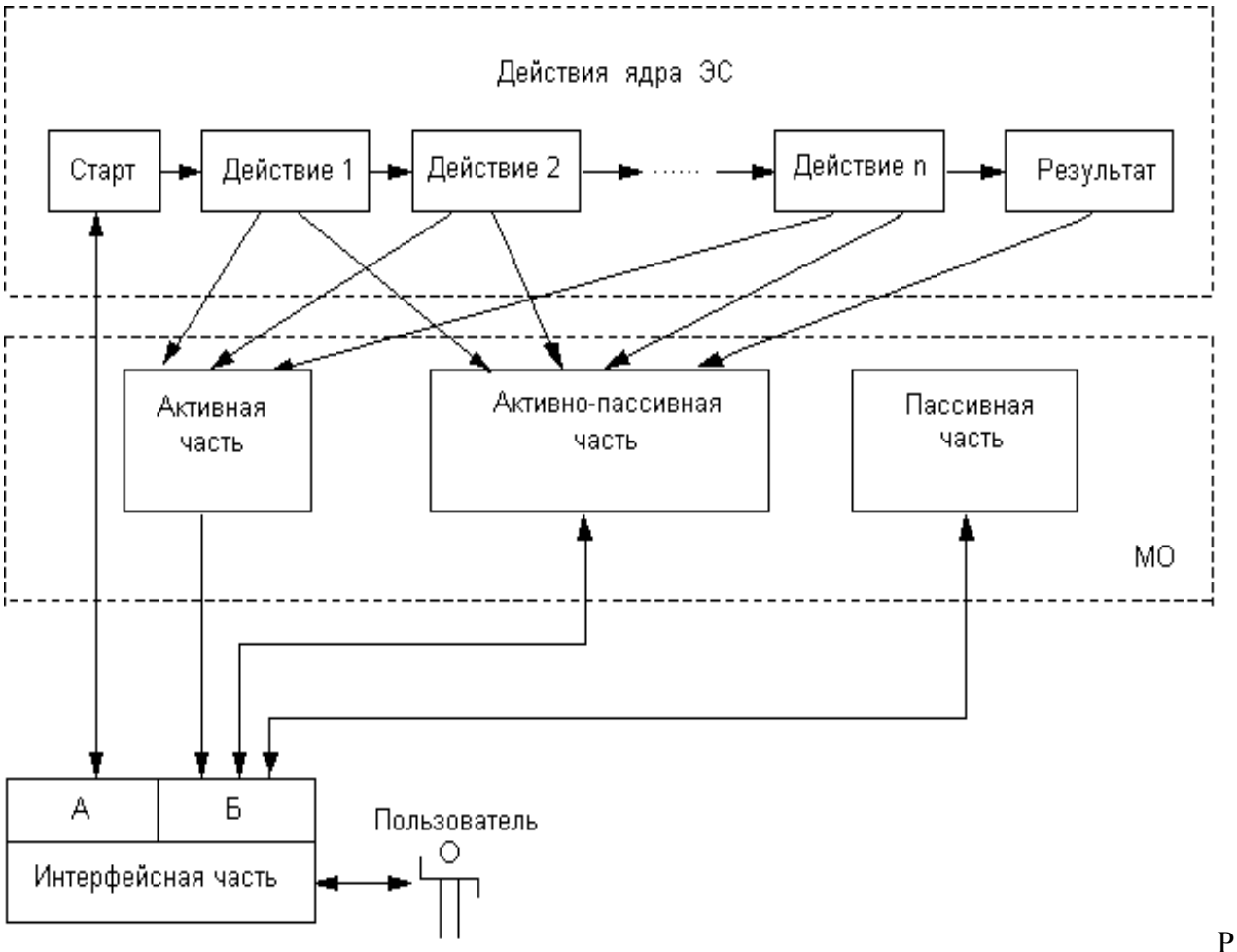
* вопросы касающиеся базы знаний.

Наибольший интерес вызывает первый вид вопросов , так как он связан непосредственно с объяснением работы ЭС. Поэтому от того , как построено объяснение для этого вида вопросов во многом зависит степень доверия к выводам, сделанным ЭС. Для этого МО должны быть средства, позволяющие отслеживать и запоминать все действия ЭС.

Второй вид вопросов касается знаний, которые имеет в своем распоряжении ЭС, в том числе знаний о самой ЭС.

Во время работы ЭС может возникнуть необходимость выдавать краткие сообщения пользователю о действиях системы независимо от того, задается вопрос или нет.

Согласно вышесказанному МО имеет в своем составе три части: активную, активно-пассивную и пассивную. На рис. 6 приведена структурная схема процесса взаимодействия ядра ЭС и МО.



Р

ис. 6. Структурная схема процесса взаимодействия ядра ЭС и МО.

Если активная часть включена, то МО выдает постоянно краткие сообщения о действиях ЭС. Данная часть реализует формирование сообщений, которые можно отнести к функциональным объяснениям. Такие объяснения строятся по принципу : “Выполняется действие X для того, чтобы выполнить действие У ”. Примером может служить следующее сообщение для пользователя : “ Выполняется поиск в БЗ эталонного значения тестового примера №5 для того, чтобы сравнить его с фактическим значением “.

Возможен упрощенный вариант функционального объяснения, который сводится к выдаче первой части сообщения о выполняемом действии в текущий момент времени.

Важно отметить, что работа МО в активном режиме будет существенно влиять на время получения результатов экспертизы.

Активно-пассивная часть - отслеживает и запоминает все действия ядра ЭС и, если есть запрос от пользователя, отвечает на вопросы о действиях системы.

Данная часть может формировать дополнительно и другие типы объяснений. Это связано с тем, что могут возникнуть вопросы, на которые невозможно ответить используя только функциональное объяснение. К таким вопросам можно отнести : “ Какое текущее состояние логики действия ЭС ? ”, “ Какая структура базы знаний ЭС ? “ , “ Как система пришла к полученному выводу ? “ и т.п.

Чтобы ответить на вопросы такого вида, необходимо использовать структурное и историческое объяснение.

Например, можно описать структуру действий системы, которая охватывает поведение ЭС в целом с помощью графа. Далее можно помечать на этом графе пройденный путь, текущее состояние логики действия ЭС, а при необходимости формировать историческое объяснение.

Пассивная часть начинает работать только в том случае, если есть запрос от пользователя. Данная часть может отвечать на вопросы, которые касаются не только знаний, но и метазнаний. В пассивной части МО могут использоваться все типы объяснений.

Например, в диагностической ЭС можно выделить следующие основные функции МО:

1. Перечисление гипотез по экспертизе диагностируемого устройства, обеспечивающих идентификацию причин неисправности;

2. Стратегия поиска неисправности в диагностируемом устройстве, представленная в форме дерева поиска;

3. Анализ текущего состояния логики действия экспертной системы;

4. Объяснение пройденного ЭС пути в процессе экспертизы диагностируемого устройства;

5. Объяснение причины, сделанного экспертной системой вывода («Почему сделан вывод Nk ?»);

6. Формирование списка имен еще не проверенных гипотез;

7. Демонстрация трассы, соответствующей данному выводу (Формирование объяснения на вопрос «Как ?». Например, «Как ЭС пришла к такому выводу ?»);

8. Формирование комментария пройденного ЭС пути;

9. Функция консультации по экспертизе (Например, сформировать ответ на вопрос: «Что следует предпринять далее ?»).

Модуля накопления знаний.

Модуль накопления (МН) является сервисным модулем, выполняющим различные вспомогательные функции.

Как правило, добавление знаний осуществляется в дискретные интервалы времени в процессе эксплуатации системы. Естественно, что добавление знаний предполагает добавление «новых» знаний. К ним относятся знания, полученное на основе сообщений по особенностям эксплуатации системы.

На начальных этапах эксплуатации системы такие знания отсутствуют. Кроме того, новые знания представляются как результат развития данного научного направления. Постоянное пополнение новыми знаниями делают систему стабильной.

В противном случае, знания, которыми обладает система устаревают, теряется их актуальность и система не способна решать новые задачи.

Для специальных систем отсутствие новых знаний может привести ее к деградации, а комплекс, в состав которого входит ЭС, будет подвергаться большой опасности.

В процессе проектирования ЭС в роли пользователя выступает эксперт, который формирует базу знаний. Суть формирования знаний заключается в их извлечении, структурировании и формализации.

После того, как указанный этап завершен, необходимо наполнить базу знаний. Роль посредника между пользователем-экспертом и ядром ЭС выполняет модуль накопления.

Следует еще выделить функцию актуализации базы знаний, отсутствие которой делает систему неэффективной, а в дальнейшем непригодной к эксплуатации.

В процессе создания ЭС, а также во время ее функционирования возникает необходимость в изменении базы знаний (БЗ). Изменение может быть связано с

выполнением следующих действий: добавление нового правила или фрейма в БЗ; модифицирование имеющегося в БЗ правила или фрейма; удаление из БЗ правила или фрейма и др.

Выбранная модель представления знаний во многом определяет инструментальное средство. Язык программирования Турбо-Пролог 2.0, например, удовлетворяет требованиям продукционной системы. В данном языке имеются собственные средства редактирования.

Сложность заключается в том, что действия связанные с изменением БЗ должен выполнять программист, знающий структуру ЭС. Носителем же знаний является эксперт-человек, который, как правило, не знает структуры ЭС и языка программирования.

Возникает необходимость создания МН , основной задачей которого является обеспечение возможности изменения БЗ пользователем не программистом, а в некоторых случаях пользователем не являющимся экспертом.

С другой стороны, МН позволит сократить время переноса знаний от эксперта в БЗ.

Процесс переноса знаний очень сложный, он включает в себя последовательное выполнение многих функций.

* *Функция извлечение знаний*. Для выполнения этой функции МН должен в диалоге с экспертом выявлять новые правила и фреймы, а также выявлять имеющиеся в БЗ правила и фреймы, для которых необходимо модифицирование или удаление. Форма диалога должна позволять формулировать правила, фреймы на подмножестве знаков естественного языка, что позволит не требовать от эксперта знания языка программирования.

* *Функция структуризации знаний*. При выполнении данной функции извлеченные знания должны быть преобразованы из естественной формы представления, удобной для эксперта, во внутреннюю форму, принятую в ЭС.

* *Функция проверки на существование*. Прежде чем выполнить добавление, модифицирование или удаление из БЗ правила или фрейма, необходимо выяснить, существует ли это правило или фрейм. Данную функцию можно назвать вспомогательной, реализация которой позволит избежать дублирования знаний в базе.

* *Функция добавления*. В случае, если не было обнаружено в БЗ вновь введенного правила или фрейма, эксперту должна предоставляться возможность добавить знания без повторного ввода с клавиатуры. Если эксперт уверен, что данного правила или фрейма не существует в БЗ, ему должна предоставляться возможность перейти непосредственно к функции добавления. Так же должна быть возможность использования данной функции в процессе работы ЭС без специального обращения к МН. Например, когда необходимая

информация в базе знаний не найдена, ЭС запрашивает ее у пользователя. В этом случае МН автоматически предоставляет возможность ввести требуемую информацию.

* *Функция модифицирования*. В случае положительного результата проверки на существование правила или фрейма, пользователю должна быть предоставлена возможность их модифицировать.

* *Функция удаления*. Необходимость данной функции очевидна. Целесообразно удалять те знания, которые либо устарели, либо противоречат вновь введенным знаниям, либо являются неправильными.

* *Функция проверки на непротиворечивость*. Вновь введенное правило или фрейм должны проверяться на непротиворечивость. Данная функция, в случае обнаружения противоречий, должна позволять в диалоге с экспертом исправлять их.

.

Организация механизма логического вывода во фреймовых системах.

В процессе получения вывода возникают задачи: какой фрейм или какой слот считать означенным (конкретизированным). Слот считается конкретизированным, если его значение определено. Если в качестве значения слота выступает другой фрейм более низкого уровня , то конкретизация слота сводится к означиванию этого фрейма.

Для означивания фрейма можно выделить следующие четыре концепции.

* Фрейм считается означенным, если конкретизированы все его слоты.

* Фрейм считается означенным, если конкретизирована какая-либо группа его слотов.

*Фрейм считается означенным, если конкретизирован слот (или какой-либо набор слотов), который является определяющим.

*Фрейм считается означенным, если суммарный вес конкретизированных слотов равен или превышает заранее заданный порог.

Вывод во фреймовых системах можно осуществлять следующими способами: - с помощью присоединительных процедур (демона и служебной процедуры); - с помощью механизма наследования.

Демон - это процедура, связанная со слотом фрейма и автоматически запускаемая при обращении к слоту. Отличие демона от служебной процедуры в том, что он конкретизирует слот без обращения к другим фреймам. Служебная процедура - это процедура, которая позволяет конкретизировать слот на основе означенного фрейма более низкого уровня.

Демон дополняет фрейм, он может являться запросом к пользователю, т.е. непосредственно быть связанным с интерфейсной частью экспертной системы.

Управление выводом с помощью механизма наследования базируется на отношениях «абстрактное - конкретное», это наиболее удобный способ реализации отношения $IS - A$.

В данном способе осуществляется автоматический поиск и определение значений слотов фрейма верхнего уровня и служебных процедур.

Фреймы, описывающие различные объекты, называют шаблонами, а фреймы верхнего уровня, используемые для представления этих шаблонов, называют фреймами класса.

Инструментальным средством, с помощью которого целесообразнее всего строить механизм вывода на основе наследования может быть объектно-ориентированный язык.

Механизм логического вывода в семантических сетях.

Отличительной особенностью семантических сетей является сложность в разграничении базы знаний и механизма логического вывода.

Интерпретация семантических сетей осуществляется с помощью использующих ее процедур. Можно выделить два способа организации процедур:

* способ сопоставления частей сетевой структуры;
* способ перекрестного поиска.

Способ сопоставления основан на построении подсети по определенному запросу и сопоставлении ее с базой данных сети. При сопоставлении с базой данных вершинам переменных подсети присваиваются гипотетические значения.

При перекрестном поиске осуществляется поиск отношений между концептуальными объектами и ответ на вопрос путем обнаружения узла, в котором пересекаются дуги, идущие от двух различных узлов. Другими словами, это обнаружение третьего узла, в котором пересекаются дуги, идущие от двух других узлов.

Этапы проектирования экспертной системы.

1. Выбор предметной области. Формулировка цели. Определение задачи. Разработка алгоритма функционирования экспертной системы.

На первом шаге необходимо провести оценку предметной области на достоверность и полноту накопленных знаний. В результате данной оценки должны быть конкретизированы источники знаний.

Экспертную систему необходимо рассмотреть в качестве семантического объекта информационной семантической системы. Данный подход позволяет сформулировать цель функционирования системы, которая определяется конкретным набором элементов. Необходимо конкретизировать данный набор на самом начальном этапе.

При определении ресурсов следует уточнить, какие виды, формы представления и преобразования семантической информации будут использоваться в экспертной системе.

Определение задачи, решаемой ЭС, должно проводиться с функциональной точки зрения. Все множество задач классифицируется по следующим группам : задачи интерпретации; диагностика; контроль; прогнозирование; планирование; проектирование.

На завершающем шаге данного этапа необходимо разработать обобщенный алгоритм функционирования ЭС.

2. Извлечение знаний.

В начале необходимо зафиксировать на естественном языке всю предоставленную информацию из различных источников знаний и уточнить основные понятия (тезаурус). Носителями знаний могут быть : человек-эксперт; документация; магнитный носитель и др.

Далее следует выделить основные и вспомогательные объекты, свойства и отношения для каждого уровня детализации.

Количество уровней детализации определяется целью создания экспертной системы. (Например, если в диагностической ЭС необходимо определять причину неисправности только на уровне основных узлов ЭВМ, то достаточно одного уровня. Далее все множество объектов необходимо разбить на два вида : материальные объекты, которые представляют конструктивную часть ЭВМ и идеальные объекты, представляющие функциональную часть ЭВМ).

Далее для каждого вида объектов выбирается способ представления знаний. (Например, знания о конструктивной части ЭВМ целесообразнее представить с помощью фреймов, где каждый из фреймов содержит информацию об адресе места возможного неисправного объекта. Знания функциональной части можно представить продукциями или семантической сетью).

Далее определяются связи между идеальными и материальными объектами для каждого уровня детализации.

3. Выбор инструментальных средств проектирования.

На данном этапе необходимо сделать заключение по выбору инструментального средства, учитывая при этом результаты предыдущих этапов. В случае выбора специальной

программной среды, автоматизирующие проектирование ЭС, или «оболочки» ЭС дальнейшие этапы разработки заметно упрощаются.

4. Формализация знаний в виде машинных процедур. Построение базы знаний .

Следующим важным шагом проектирования ЭС является разработка структуры базы знаний и определение ее прикладной единицы (базового элемента или структуры). От решения этой задачи зависят: эффективность работы механизма логического вывода; дублирование и избыточность информации базы знаний; осуществление диалога между экспертной системой и пользователем; возможность подключения блока объяснения непосредственно в ходе ведения диалога; возможность создания блока накопления знаний для пользователя не программиста и др.

Кроме вышеуказанного, например, в частном случае, прикладная единица базы знаний должна учитывать связь между функциональной и конструктивной частями диагностируемой ЭВМ. При этом необходимо предусмотреть возможность участия пользователя в процессе получения заключения экспертной системой.

В завершении данного этапа разрабатываются исходные тексты программ, реализующих базу знаний на выбранном инструментальном средстве.

5. *Разработка семантического интерфейса «ЭС - пользователь».*

Разработка интерфейса «ЭС - пользователь», основные функции которого: осуществлять преобразование сообщения из естественно-языковой формы в форму внутреннего представления и обратное преобразование; анализ и синтез сообщений пользователя и ЭС; отслеживание и запоминание пройденного пути диалога.

На выходе данного этапа должны быть разработаны возможные сценарии диалога между ЭС и пользователем в виде блок-схем и текстов программ.

6. Разработка механизма логического вывода.

После того, как знания представлены с помощью выбранных методов, необходимо приступить к разработке механизма логического вывода (МЛВ).

Основная задача МЛВ, например для продукционной модели, это реализация стратегии выбора соответствующего правила, факта . В частности необходимо разработать четыре процесса: выбор активных правил и фактов; сопоставление; разрешение конфликтов; выполнение выбранного означенного правила (действие).

Основной задачей при реализации первого процесса является разработка различных схем порождения новых фактов и формирование правил, направленных на достижение целевых фактов.

После того, как было получено множество активных правил возникает необходимость в их сопоставлении, т.е. определение какие правила максимально охватывают множество активных фактов и какие правила целесообразнее выполнять в первую очередь.

Далее необходимо разработать процесс разрешения конфликтов. Данный процесс в зависимости от способа выбора правила должен обрабатывать все возможные конфликтные ситуации.

В результате должны быть разработаны алгоритмы функционирования ядра экспертной системы и тексты программ.

7. Разработка модуля объяснений.

Назначение данного модуля - сделать ЭС «прозрачной» для пользователя, т.е. предоставить пользователю возможность понимать логику действий ЭС, дать надежную гарантию правильности полученных результатов.

На выходе данного тапа должны быть разработаны исходные тексты программ.

8. Разработка модуля накопления знаний и манипулирования со знаниями.

Основными функциями модуля накопления знаний являются : автоматизация процесса наполнения базы знаний; актуализации базы знаний. На этом этапе разрабатываются алгоритмы функционирования модуля накоплений и экранные формы, позволяющие осуществлять операции манипулирования со знаниями применительно к выбранной структуре базы знаний. В результате должны быть разработаны исходные тексты программ.

Примеры экспертных систем.

В примере 1 приведен текст программы простейшей экспертной системы продукционного типа.

Особенностями данной программы являются:

* структура базы знаний такова: что на ее основе можно разработать модуль накопления ориентированный на пользователя не программиста;

* базу фактов и базу правил можно вынести за пределы программы в отдельный текстовый файл в соответствии с форматом базы данных, а при необходимости подгружать их (или удалять из памяти);

* механизм логического вывода в своем роде универсальный (не требует коррекции при появлении новых фактов или правил);

* реализован обратный вывод с поиском в глубину;

* в интерфейсном модуле реализована фразовая форма с использованием морфологического анализа декларативным методом.

На рис. 7 приведена структура ЭС примера 1, в которой определяется животное и принадлежность его к определенному классу .

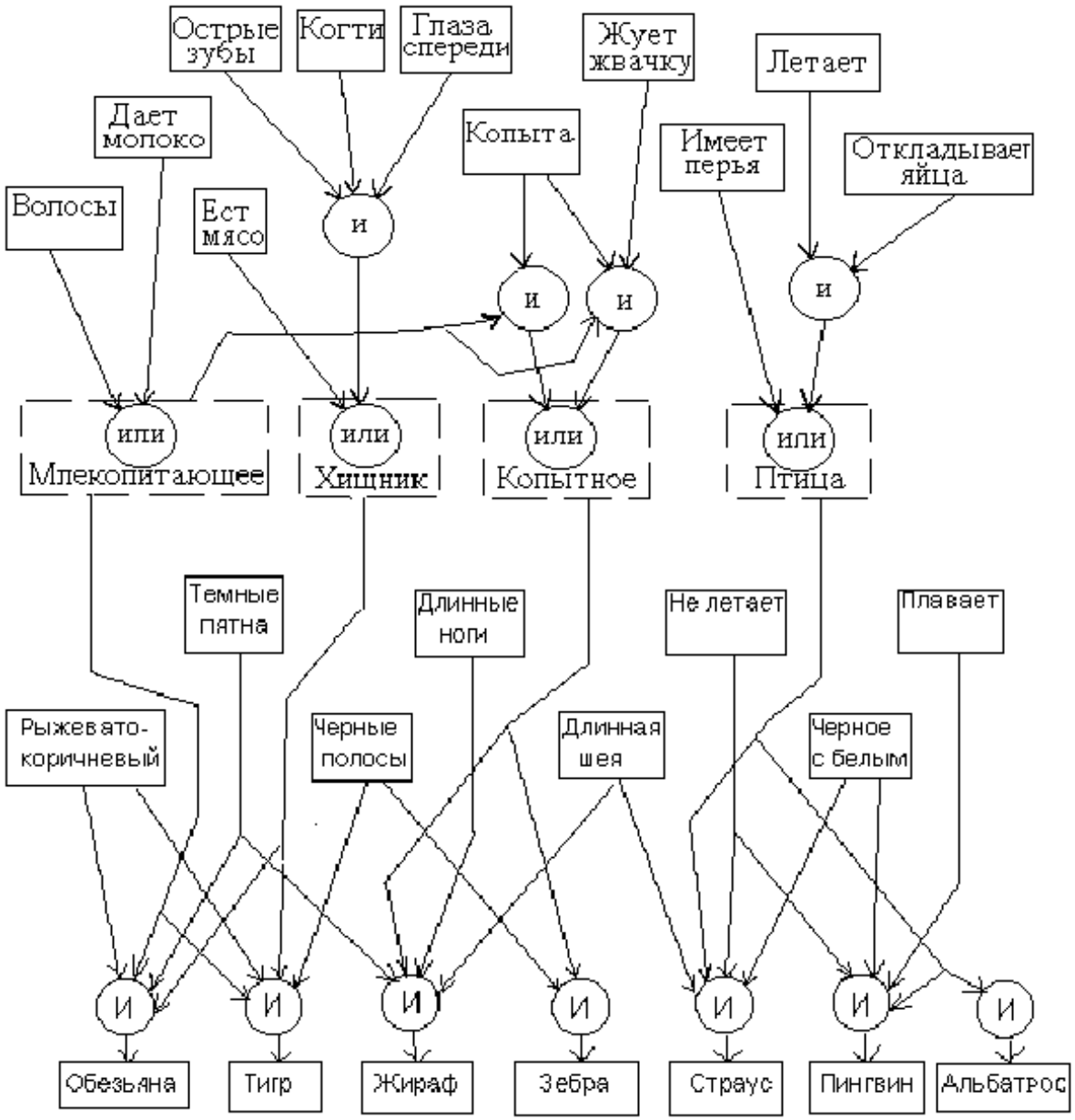


Рис. 7. Структура МЛВ ЭС примера 1.

Обобщенный алгоритм программы примера 1 можно описать так:

* Получение активных фактов в результате диалога;
* Формирование списков групп различных фактов и правил;
* Просмотр заключений и выбор активных правил;
* Выдача результатов работы;
* Возможность уточнения (ввода дополнительных фактов);
* Работа МЛВ до тех пор, пока генерируется новые факты.

В примере 2 приведена ЭС определения вероятности инфаркта.

Особенностями данной системы являются:

* Форма модуля интерфейса табличная и директивная;
* Используется функция модуля объяснения (отслеживается пройденный путь);

Задания ко 2-й части лабораторных работ по СИИ.

Общее требование:

1) Исследовать предметную область;
2) Выбрать модель представления знаний;
3) Представить знания в виде программной модели на языке Турбо-Пролог.
4) Реализовать функцию в соответствии с вариантом задания.

Отчет должен содержать:

- Обоснование выбора модели представления знаний и ее структуру;
- Программную модель базы знаний;
- Обобщенный алгоритм работы программы;
- Текст программы;
- Достоинства и недостатки разработанной программы.

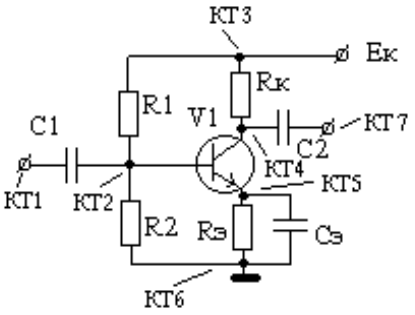
Тема 1. Консультирующая интерактивная экспертная система по определению оптимальной конфигурации ПЭВМ.

Основными входными фактами (данными) являются:

- Цель, для которой приобретается ПЭВМ;
- Пределы допустимой суммы ($ - $);
- Фирма (страна) изготовитель.

Тема 2. Диагностическая интерактивная экспертная система проверки работоспособности однокаскадного усилителя в статике.

Основными входными фактами (данными) являются величины напряжений в контрольных точках.

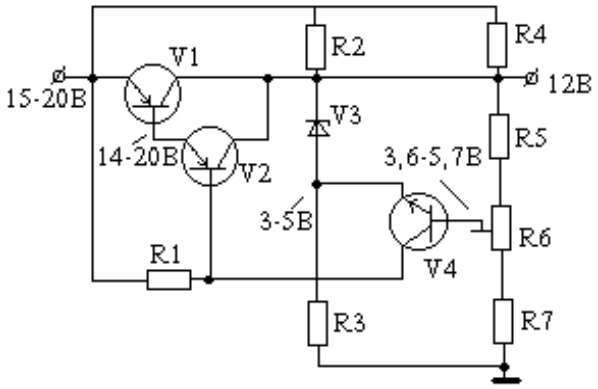


R1 - 8,2 кОм
R2 - 4,7 кОм
Rк - 1,2 кОм
Rэ - 1 кОм
C1 - 220
C2 - 470
Сэ - 6800

КТ1 - +11В
КТ2 - +7,5В
КТ3 - +18В
КТ4 - +15В
КТ5 - +7В
КТ6 - 0В
КТ7 - +11В

Тема 3. Диагностическая интерактивная экспертная система проверки работоспособности модуля стабилизации.

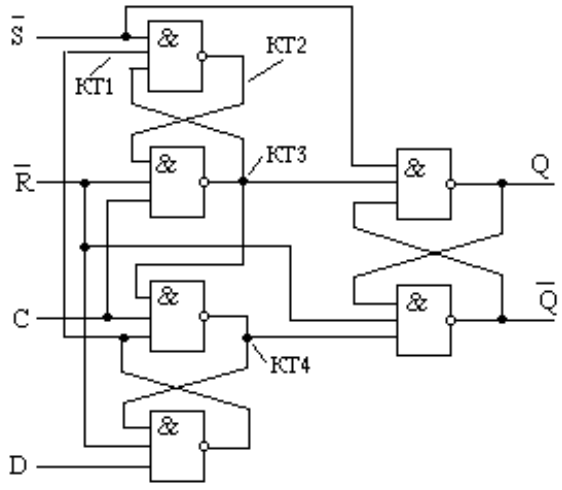
Основными входными фактами (данными) являются величины напряжений в контрольных точках.



R1 - 220
R2 - 100
R3 - 750
R4 - 100
R5 - 180
R6 - 470
R7 - 220

Тема 4. Диагностическая интерактивная экспертная система проверки работоспособности D- триггера (на элементах ТТЛ) в статике.

Основными входными фактами (данными) являются величины напряжений на входах, выходах и в контрольных точках узла.



Тема 5. Консультирующая экспертная система для выбора породы собаки.

Основными входными фактами (данными) являются:

- шерсть (длинная, короткая);
- окрас (черная, белая и т.п.);
- рост;
- хвост (низкопосаженный и т.п.);
- уши (длинные, короткие и т.п.);
- характер;
- вес;
- для каких целей.

Например,

- Английский бульдог (короткая шерсть, рост меньше 22 дюймов, низкопосаженный хвост, хороший характер и т.д.);
- Гончая (короткая шерсть, рост меньше 22 дюймов, длинные уши, хороший характер и т.д.);
- Дог (короткая шерсть, рост меньше 30 дюймов, длинные уши, хороший характер, вес более 100 фунтов и т.д.);
- Американская гончая (короткая шерсть, низкопосаженный хвост, длинные уши, хороший характер и т.д.);
- Коккер-спаниэль (длинная шерсть, рост меньше 22 дюймов, низкопосаженный хвост, длинные уши, хороший характер и т.д.);
- Ирландский сеттер (длинная шерсть, рост меньше 30 дюймов, длинные уши и т.д.);
- Колли (длинная шерсть, рост меньше 22 дюймов, , низкопосаженный хвост, хороший характер и т.д.);
- Сенбернар (длинная шерсть, низкопосаженный хвост, хороший характер, вес более 100 фунтов и т.д.) и т.д.

Тема 6. Медицинская консультирующая экспертная система по выбору лекарственных трав.

- Основными входными фактами (данными) являются симптомы болезни.

Ниже приведены примеры некоторых растений:

(Имя растения, показания к применению, действие на организм)

1. Аир обыкновенный(Acorus Calamus).
   Язва желудка, гастрит с пониженной желудочной секрецией.
   Улучшает аппетит, усиливает отделение желудочного сока.
2. Аскомицеты или сумчатые грибы (Ascomicetis).
   Гипертония , повышенная раздражительность, спазмы сосудов
   Останавливает внутреннее кровотечение седативное действие
   на возбуждение ЦНС(Центральная Нервная Система), снижает кровяное давление.
3. Боярышник отогнуто чашелистиковый(Crategus curvisepala Lindin).
   Спонтанные боли в сердце, начальная гипертония.
   Снижает возбудимость ЦНС, тонизирует мышцу сердца, усиливает кровообращение в коронарных сосудах, снижает А/Д.
4. "Лопух большой(Arctium lappa).
   Отеки почечного и сердечного происхождения, отсутствие пота, обострение геморроя, детский диатез, злокачественные опухоли, бельмо, бородавки, хронические болезни печени.
   Способствует увеличению диуреза, снимает воспалительные процессы, при обработке волос - укрепляет волосяные сумки, помогает при экземе, улучшает функцию почек.

5. Можжевельник обыкновенный(Iunperus communis).
   Отеки почечного происхождения, гипотония, инфекция мочевых путей, кашель, неустойчивый стул.",
   Оказывает мочегонное действие, дезинфекция мочевых путей, отхаркивающее действие, улучшает пищеварение."
6. Ольха черная (Alnus glutinosa),ольха серая (Volrha sehera)",
   Энтероколиты, ожоги, дерматиты, кровотечения из носа, поносы, геморрой.
   Противовоспалительные и бактерицидные, особенно при поверхностных гнойных процессах."
7. Одуванчик лекарственный (Taraxacum officinalis Wigg).
   Нарушение аппетита, запоры, холициститы, перевозбуждение ЦНС.
   Возбуждает аппетит; как желчегонное средство, как успокаивающее средство. делает жидким стул.
8. Пастушья сумка(Capsella bursa- pastoris).
   Гипертиреоз, при капиллярном кровотечении.
   Повышает свертываемость крови, понижает давление крови, принимают при поносе, болезнях печени.
9. Сирень обыкновенная (Syrtuga Vulgaris).
   "Невралгии, воспалительные процессы, язва желудка, кашель, повязки на гнойные раны, отеки почечного происхождения.",
   "Снимает боль при местных воспалениях, улучшает состояние при общих простудах, воспалительных процессах, улучшает почечные отеки, мочегонное.
10. Хрен обыкновенный.
    Отсутствие аппетита ,авитаминоз, понижение желудочной секреции, ухудшение пищеварения(неустойчивый стул),вирусный гепатит,ангины.
    "Повышает аппетит, улучшает пищеварение, усиливает секрецию пищеварительных желез, желчегонное средство, как отвлекающее.
11. "Полынь горькая(Artemisia absintinum).
    "Пониженный аппетит.
    Улучшает аппетит, пищеварение ,желчегонное действие, спазмолитическое действие, аппликации при рентгеновских ожогах, экземах, бронхиальной астме.
12. Грецкий орех(Iuglands regia).
    Гиповитаминоз С, поносы, неустойчивый стул, желудочное геморроидальное кровотечение, кожный туберкулез, другие кожные заболевания.
    Вяжущее действие на раздраженную слизистую оболочку, кровоостанавливающее действие на десны, геморрой, язву желудка.

Тема 7. Диагностическая медицинская экспертная система.

Основными входными фактами (данными) являются ответы пациента на вопросы, задаваемые экспертной системой.

Пациент должен выразить степень согласия на ниже перечисленные утверждения.

- 'В общем я нервный';
- 'Я очень беспокоюсь о своей работе';
- 'Я часто ощущаю нервное напряжение';
- 'Моя повседневная деятельность вызывает большое напряжение';
- 'Общаясь с людьми, я часто ощущаю нервное напряжение';
- 'К концу дня я совершенно истощен физически и психически';

Степень согласия может быть выражена одним из следующих четырех вариантов:

1. "ДА, СОГЛАСЕН"
2. "СКОРЕЕ, СОГЛАСЕН"
3. "СКОРЕЕ, НЕ СОГЛАСЕН"
4. "НЕТ, НЕ СОГЛАСЕН"

Каждый вариант ответа имеет свой вес в соответствии с его порядковым номером, т.е. равен ему.

Система может выдать два решения:

1: Повышенное психоэмоциональное напряжение.

2: Психоэмоциональное напряжение в норме.

Решение выдается на основе среднего веса следующим образом:

- если пациент мужчина и (A1+A2+A3+A4+A5+A6)/6<=2, то 1 вариант решения;
- если пациент женщина и (A1+A2+A3+A4+A5+A6)/6<=1.83, то 1 вариант решения.

Во всех остальных случаях выдается второй вариант решения.

Варианты задания:

Вариант	Тема	Модуль, функции экспертной системы
1	1	Механизм логического вывода с поиском в глубину
2	1	Модуль объяснения.
3	1	Модуль накопления знаний.
4	1	Функцию объяснения пройденного пути.
5	2	Механизм логического вывода.
6	2	Модуль накопления знаний.
7	2	Модуль объяснения
8	2	Режим «А что будет, если ?»
9	3	Механизм логического вывода.
10	3	Модуль накопления знаний.
11	3	Модуль объяснения
12	3	Стратегия поиска в форме дерева
13	4	Механизм логического вывода.
14	4	Модуль накопления знаний.
15	4	Модуль объяснения
16	4	Что следует предпринять далее
17	5	Механизм логического вывода.
18	5	Модуль накопления знаний.
19	5	Модуль объяснения
20	5	Функция извлечения и структуризации
21	6	Механизм логического вывода.
22	6	Модуль накопления знаний.
23	6	Модуль объяснения
24	6	Проверка на непротиворечивость
25	7	Механизм обратного вывода .
26	7	Механизм прямого вывода
27	7	Модуль накопления
28	7	Объяснение пройденного пути

Пример 1. Экспертная система определения животного .

Domains

```
i = integer  s = string  c = char
list=integer* %  список целых чисел
```

Database-f % база фактов интерфейсного модуля

```
w(i,i,s)  have(i,s)  give(i,s)  feed(i,s) move(i,s)  dop(i,s)
```

Database-rul % база заключений

```
rule(i,list,list,list,list,list,s) % факты заключений
```

Database

```
b(s,s) quit  nnn(i) % для вспомогательных целей
% предикаты для формирования активных фактов
   have_act(i)  give_act(i)  feed_act(i) move_act(i)  dop_act(i)
buff(s)    % буфер выделенных слов или фраз
vbuff(s)   % для формирование одного слова или одной фразы
fact_rule(i,s)  % факты полученных заключений
tim_rule(i,s)  % для формирования списка заключений
split(i,list)  % для формирования списков по номерам
split_rule_old(list) % список предыдущих сработанных правил
```

Predicates

```
start
  % предикаты для описания фактов интерфейсного модуля
readmy(s) anedit(i,s,s) interface
asbuff(s,s)  % для множества выделенных слов или фраз
prob(s,s) % для определения лишних пробелов
predprob(s,s) % для удаления пробела перед разделителем
wp(i,i) % для обработки вопросника
   %для обработки строк
d1(s) pr1(s,s) pr12(i) pr2(s) v(c,c)  sumb(s)

formact asform(s) % для формирования активных фактов
% предикаты для организации механизма вывода
  repeat no_iterac
  analiz(i,list,list,list,list,list,list,s)
  sortsp(i)  con(list,list) choo(i,i) form(i)
  cross(list,list) % для проверки на пересечение списков
  cross_rule(list,list) % для проверки на пересечение списков
  in_split(i,list) % для проверки вхождения в список
  compin(i,List,List)  compout(List,List)
  zak  outzakl  % вывод заключений
```

Goal start.

Clauses

```
start:-makewindow(1,1,7," ЭС определения животных (Выход - Esc... )",0,0,25,80),
     makewindow(2,27,47,"Перечислите возможные ответы",5,10,15,59),interface.


interface:-  retractall(_),   % удаляем старые факты, если они есть
             assert(split_rule_old([])), % начальные сработанные правила
             wp(1,1),  % опрос и формирование основных входных фактов
             not(quit), % если не Esc, то работаем дальше

                /* Обработка основных фактов*/
             formact,   % формирование активных фактов
             repeat,    % вызов механизма вывода
             zak, % вывод результатов работы

                /* Обработка дополнительных фактов*/

           wp(2,1),  % опрос и формирование дополнительных фактов
             not(quit), % если не Esc, то работаем дальше
             formact,   % формирование активных фактов
             repeat,    % вызов механизма вывода
             zak, % вывод результатов работы
             fail.

interface:-  quit,cursor(5,10), write("Сеанс работы завершен ..."),
             readchar(_),removewindow,removewindow,exit.
interface:- interface.


/* ************************************************************
 *************** Интерфейс с пользователем ********************
 *********** морфологический анализ декларативным методом ******/

% Цикл опроса и формирование активных фактов
% Вопрос выводим как заголовок меню
wp(K,N):-W(K,N,W),makewindow(3,54,27,W,10,14,4,50), d1(S),not(quit),N1=N+1,
       removewindow,wp(K,N1),!.
wp(K,N):-quit,removewindow,!.
wp(K,N).

% ввод строки
% S-обработанная строка (буквы после обработки только строчные)

d1(S):- readmy(Sv), pr1(Sv,S), pr2(S),!.

readmy(S):- editmsg("",Sout,"F10 - ввод вопроса  Esc - отказ","","",0,"",U),
            anedit(U,S,Sout).
anedit(U,S,Sout):-U=0,S=Sout,!. % если нажата F10
anedit(U,S,Sout):-U=1,S="", assert(quit),!. % если нажата Esc
```

```
% обработка входной строки
pr1(Sv,S1):- str_len(Sv,L),retractall(b(_,_)),
             assert(b(Sv,"")),pr12(L),b(_,S1),!.
pr12(L):- b(S,H),L<>0,frontchar(S,C,So),v(C2,C),
        str_char(Sc,C2),concat(H,Sc,H1),
        retractall(b(_,_)),L1=L-1,assert(b(So,H1)),pr12(L1).
pr12(L):-L=0,!.
pr12(L):-pr12(L).

% Для преобразования входной строки из русских букв
v('й','Й').v('ш','Ш').v('в','В').v('д','Д').v('м','М').v('ц','Ц').
v('щ','Щ').v('а','А').v('ж','Ж').v('и','И').v('у','У').v('з','З').
v('п','П').v('э','Э').v('т','Т').v('к','К').v('х','Х').v('р','Р').
v('я','Я').v('ь','Ь').v('е','Е').v('ъ','Ъ').v('о','О').v('ч','Ч').
v('б','Б').v('н','Н').v('ф','Ф').v('л','Л').v('с','С').v('ю','Ю').
v('г','Г').v('ы','Ы').
v(C,C2):-C=C2,!.

% формируем базу фактов выделенных слов или фраз
pr2(S):-S<>"",frontstr(1,S,S1,S2),asbuff(S1,S2),pr2(S2).
pr2(S).

% формируем в vbuff фразу или слово и отправляем в buff
% при этом лишние пробелы пропускаем

asbuff(S1,S2):-not(vbuff(_)),assert(vbuff("")),fail.
asbuff(S1,S2):-not(sumb(S1)),vbuff(S),prob(S1,S2),concat(S,S1,Sz),
              retractall(vbuff(_)),assert(vbuff(Sz)),fail.
asbuff(S1,S2):-not(sumb(S1)),S2<>"",!.
asbuff(S1,S2):-vbuff(Sn), predprob(Sn,S),
              retractall(vbuff(_)),assert(buff(S)),!.
% если разделитель, а предыдущий пробел, то удаляем этот пробел
predprob(Sn,S):- str_len(Sn,L),L1=L-1,frontstr(L1,Sn,Ss,Sz),Sz=" ",S=Ss,!.
predprob(Sn,S):- S=Sn,!.

% узнаем это лишний пробел или нет
prob(S1,S2):-S1<>" ",!.
prob(S1,S2):-S1=" ",vbuff(S),S<>"",str_len(S,L),L1=L-1,
                    frontstr(L1,S,Sn,Sz),Sz<>" ",!.
prob(S1,S2):-S2="".
sumb(",").sumb(".").sumb(";").    % разделители слов или фраз

% Вопросник
% основная группа вопросов
W(1,1,"Что имеет ?").
W(1,2,"Что дает ?").
W(1,3,"Чем питается ?").
W(1,4,"Как перемещается ?").

% дополнительная группа вопросов
```

```
W(2,1,"Что дополнительно можно добавить ?").


/* ************************************************************
 ************** Формирование  активных фактов ***************
 ************************************************************/

formact:-buff(A), asform(A),fail.
      % создание сортированных списков номеров фактов
formact:-sortsp(1),sortsp(2),sortsp(3),sortsp(4),sortsp(5),sortsp(6),!.
formact.

          % получаем номера активных фактов для разных групп
asform(A):-have(N,A),not(have_act(N)),assert(have_act(N)),fail.
asform(A):-give(N,A),not(give_act(N)),assert(give_act(N)),fail.
asform(A):-feed(N,A),not(feed_act(N)),assert(feed_act(N)),fail.
asform(A):-move(N,A),not(move_act(N)),assert(move_act(N)),fail.
asform(A):-dop(N,A),not(dop_act(N)),assert(dop_act(N)),fail.
asform(A).


% создаем отсортированные списки
% в группе не более 100 фактов или сработанных правил

sortsp(N):-not(split(N,_)),assert(split(N,[])),fail.
sortsp(N):-not(nnn(_)),assert(nnn(100)),fail.

sortsp(N):-nnn(M),M>0, choo(N,M), fail.
sortsp(N):-nnn(M),M>0,not(choo(N,M)),M1=M-1,retractall(nnn(_)),
            assert(nnn(M1)),fail.
sortsp(N):-nnn(M),M=0,retractall(nnn(_)),!.
sortsp(N):-sortsp(N).


choo(1,M):-have_act(M),retractall(have_act(M)),form(1).

choo(2,M):-give_act(M),retractall(give_act(M)),form(2).

choo(3,M):-feed_act(M),retractall(feed_act(M)),form(3).

choo(4,M):-move_act(M),retractall(move_act(M)),form(4).

choo(5,M):-dop_act(M),retractall(dop_act(M)),form(5). % дополнительные факты

choo(6,M):-fact_rule(M,_),retractall(tim_rule(M,_)),form(6). % сработанное правило

form(N):- nnn(M), split(N,B),not(in_split(M,B)),con([M|B],S),
              retractall(split(N,_)),assert(split(N,S)),
              M1=M-1,retractall(nnn(_)),assert(nnn(M1)),!.
```

```
con(A,S):-S=A,!.  % для сбора в список

% проверяем вхождение элемента в список
  in_split(M,[A|B]):-M=A,!.
  in_split(M,[A|B]):-not(B=[]),in_split(M,B).


/*****************************************************
 ***** База правил  (процедурные знания) ************
 *****************************************************/

/* основное универсальное метаправило
   (перебираем все заключения и анализируем,
    получаем заключения сработанных правил fact_rule)*/


repeat:-rule(N,Split_have,Split_give,Split_feed,Split_move,Split_dop,Split_rule,Zakl),
        analiz(N,Split_have,Split_give,Split_feed,Split_move,Split_dop,Split_rule,Zakl),
        fail.
repeat:-no_iterac,!.% если не нужно делать еще итерацию, то выходим
repeat:-repeat.

no_iterac:-split(6,A),split_rule_old(B),
        %  write(A,"-",B),nl,readchar(_),
          A=B,!.
no_iterac:-split(6,A),
          retractall(split_rule_old(_)),
             assert(split_rule_old(A)),fail.

% анализируем на срабатывание заключений
analiz(N,S_h,S_g,S_f,S_m,S_d,S_r,Z):-
     split(1,S1), cross(S_h,S1),
     split(2,S2), cross(S_g,S2),
     split(3,S3), cross(S_f,S3),
     split(4,S4), cross(S_m,S4),
     split(5,S5), cross(S_d,S5),
     split(6,S6), cross_rule(S_r,S6),
     not(fact_rule(N,_)),
     assert(fact_rule(N,Z)),
     assert(tim_rule(N,Z)), % для формирования списка заключений
     sortsp(6). % сортируем список сработанных правил


  % анализ на совпадение или пересечение списков
cross(Lr,Lf):-Lf=Lr,!. % если просто совпали, то выходим
cross(Lr,Lf):-Lr=[],!. % если в правиле пустое множество, то не анализируем
cross([Ra|Rb],Lf):- compin(Ra,Lf,Lfn),compout([Ra|Rb],Lfn),!.

  % поиск начала вхождения одного списка в другой (возвращаем хвост Lfn)
compin(Ra,[Fa|Fb],Lfn):- Ra=Fa,Lfn=[Fa|Fb],!.
```

```
compin(Ra,[Fa|Fb],Lfn):- Ra<>Fa,not(Fb=[]),compin(Ra,Fb,Lfn).
compin(Ra,[Fa|Fb],Lfn):- Fb=[],Lfn=[].

compout([Ra|Rb],[Fa|Fb]):- Ra=Fa,not(Rb=[]),compout(Rb,Fb).
compout([Ra|Rb],[Fa|Fb]):- Ra=Fa,Rb=[],!. % то входит

  % анализ только на совпадение
cross_rule(Lr,Lf):-Lf=Lr,!.
cross_rule(Lr,Lf):-Lr=[],!.


% Вывод сформированных заключений

zak:-fact_rule(_,_), % если есть заключения, то выводим
    makewindow(4,32,47," Такие вот выводы ! ",3,4,17,70),
    outzakl,readchar(_),removewindow,!.
zak:-not(fact_rule(_,_)), % если нет заключений
    makewindow(4,32,47," Просим прощения ! ",3,4,17,70),
    cursor(5,10),write("Трудно что-нибудь вывести по данным фактам!"),
    readchar(_),removewindow,!.
zak.


outzakl:-fact_rule(N,Z),write("Это ",Z,".  (Сработало правило ",N,")"),
        nl,fail.
outzakl.


% База фактов интерфейса с пользователем (декларативные знания)
have(1,"волосы").
have(2,"копыта").
have(3,"зубы").   have(3,"острые зубы"). have(3,"зубы острые").
have(4,"когти").  have(4,"большие когти").
have(5,"глаза спереди"). have(5,"вперед смотрящие глаза").
have(6,"лапы").
have(7,"перья").

give(1,"молоко").
give(2,"яйца").

feed(1,"мясо").
feed(2,"жует жвачку"). feed(2,"трава"). feed(2,"растения").
feed(3,"рыба").
feed(4,"овощи").
feed(5,"фрукты").
feed(6,"чем угодно").feed(6,"всем").feed(6,"что попадется").

move(1,"летает"). move(1,"хорошо летает"). move(1,"летает хорошо").
move(2,"плавает").
move(3,"ходит").
```

```
move(4,"бегает"). move(4,"быстро бегает"). move(4,"бегает быстро").

dop(1,"рыжевато-коричневый").
dop(2,"темные пятна"). dop(2,"пятна темные").
dop(3,"черные полосы"). dop(3,"полосы черные").
dop(4,"длинные ноги"). dop(4,"ноги длинные").
dop(5,"длинная шея").  dop(5,"шея длинная").
dop(6,"не летает").
dop(7,"черное с белым").  dop(7,"белое с чнрным").
dop(8,"плавает").
dop(9,"хорошо летает").  dop(9,"летает хорошо").


/**************** База фактов заключений ***************************
 rule(N правила, списки номеров фактов(имеет,дает,питается,перемещается,
     дополнительные) и срабатываемых правил(заререзвированы),заключение) */
% внутри правила реализовано по 'И' , группа правил реализует 'ИЛИ'

rule(1,[1],[],[],[],[],[],"Млекопитающее").
rule(1,[],[1],[],[],[],[],"Млекопитающее").
rule(1,[1],[1],[],[],[],[],"Млекопитающее").

rule(2,[],[],[1],[],[],[],"Хищник").
rule(2,[],[],[3],[],[],[],"Хищник").
rule(2,[3,4,5],[],[],[],[],[],"Хищник").
rule(2,[3,4,5],[1],[],[],[],[],"Хищник").
rule(2,[3,4,5],[3],[],[],[],[],"Хищник").

rule(3,[],[],[2],[],[],[1],"Копытное").
rule(3,[2],[],[],[],[],[1],"Копытное").

rule(4,[],[2],[],[1],[],[],"Птица").
rule(4,[7],[],[],[],[],[],"Птица").

rule(5,[],[],[],[],[],[1,2],[2,1],"Обезьяна").
rule(6,[],[],[],[],[],[1,3],[2,1],"Тигр").
rule(7,[],[],[],[],[],[2,4,5],[3,1],"Жираф").
rule(8,[],[],[],[],[],[3],[3,1],"Зебра").
rule(9,[],[],[],[],[],[5,6,7],[4],"Страус").
rule(10,[],[],[],[],[],[6,7,8],[4],"Пингвин").
rule(11,[],[],[],[],[],[9],[4],"Альбатрос").
```

Пример 2. Экспертная система определения вероятности инфаркта.

```
/*************** ЭКСПЕРТНАЯ СИСТЕМА **********************/

CONSTANTS
    cz0=63 % исходный цвет закраски пунктов меню
```

```
col=18 % цвет для пометки выбранных пунктов
c1=63  % цвет фона и символов окон 14 и 15
c2=118 % цвет обрамления и названия окон 14 и 15
```

DOMAINS

```
% вводим с целью упрощения дальнейшего описания
 i = integer  s = string c = char  r = real
```

DATABASE

```
wpw(s,s,s,s,s,s,s,s,s) % для хранения вариантов ответов и вероятности инфаркта

ar(i,i)  %для хранения кода нажатой клавиши и номера выбранного пункта меню

mn(i,i)  % вспомогательная база для вывода меню

tw(i)    % для хранения текущего номера вопроса или кода меню

aw(i,s)  % вспомогательная база интерфейсной части

rom(i,i) % для отдельного хранения номера вопроса и ответа

ind(i)   % для индикации: 1 - работа в главном режиме; 2 - в режиме опроса

q(i)     % вспомогательная база режима опроса
```

PREDICATES

```
start      % предикат целевого утверждения
menu       % для работы с меню
wop(i,i,i)
k(i)       % факты кодов клавиш стрелок

read_kod(i)  char_kod(i,i) % для чтения с клавиатуры
o(i,i,s,i,i)  d(i,i,i,i)   % для описания фактов меню

bor        % для основного цикла программы
an(i,i)    % для анализа и обработки нажатых клавиш
uz(i,i,s,i) % для описания фактов-вопросов

rw(i) ro(i,i) pw(i) po(i,i) % для разметки или пометки пунктов меню

lop(i)     % для организации считывания цифр в режиме опроса
let(i,s)
cur(i,i,i)
asrom(i,i)  % для инициализации базы с предикатом rom
lb(i,i)    % для защиты по количеству вводимых цифр
preobr     % для описания основных правил непосредственного
vero(r,s)  % определения баллов и вероятности инфаркта
dbl(i,s)   % для работы с базой rom(i,i)
```

```
    qduk       % для работы с базой q(i)
    lb1(i,i)   % факты для ограничения диапазона ввода цифр
    wiw        % для вывода информации в окно пройденного пути
    wiwex      % для вывода основного результата работы системы
    sob        % для вывода служебного сообщения

GOAL
  assert(ind(1)),assert(wpw("","","","","","","","","","")),
  makewindow(3,c1,c2," П Р О Й Д Е Н Н Ы Й   П У Т Ь ",2,40,21,37),
  makewindow(4,c1,c2," О К Н О   Д И А Л О Г А ",2,3,21,37),
  start.

CLAUSES

start:-ind(Q),Q=1,
  makewindow(1,82,111," МЕДИЦИНСКАЯ ЭКСПЕРТНАЯ СИСТЕМА ",0,0,25,80),
  makewindow(2,62,c2," ГЛАВНОЕ МЕНЮ ",5,10,8,50),
  retractall(mn(_,_)),assert(mn(10,10)), menu,
  retractall(mn(_,_)),po(10,1),
  retractall(tw(_)),assert(tw(10)),
  retractall(ar(_,_)),assert(ar(80,1)),
  bor,!.

% считывание из базы пунктов меню
menu:- mn(N1,N2),X2=N1+1,N1<N2+1,retractall(mn(_,_)),
       assert(mn(X2,N2)),uz(N1,_,_,Y), wop(N1,1,Y),fail.
menu:- mn(N1,N2),N1<N2+1,menu,!.
menu:- mn(N1,N2),N1=N2+1,!.

% непосредственный вывод пунктов меню
wop(N,X,Y):-  o(N,X,A,D1,D2),cursor(D1,D2),
            write(A),str_len(A,L),field_attr(D1,D2,L,cz0),
            X2=X+1,X<Y,wop(N,X2,Y),!.
wop(N,X,Y):-X=Y,!.

% Инициализация вспомогательной базы для хранения вариантов ответов
asrom(N,M):-N<=M,assert(rom(N,0)),N1=N+1,asrom(N1,M).
asrom(N,M).

% Основные правила анализа выбранных пунктов меню и
%         чтения кода клавиш

bor:- ar(K,R),K<>100,tw(W),W>4,W<>6,W<8,read_kod(K1), an(K1,R),fail.
bor:- ar(K,R),K=100,tw(W),W>4,W<>6,W<8, an(K,R),fail.
bor:- ar(K,R),K<>27,tw(W),W<7,W<>5,an(K,R),fail.
bor:- ar(K,R),K<>13,tw(W),W=10,read_kod(K1),an(K1,R),fail.

% выход на исполнение пунктов главного меню
bor:- ar(13,R),tw(10),R=1,an(300,1),fail,!. % режим опроса
bor:- ar(13,R),tw(10),R=2,an(500,1),fail,!. % сведения о системе
```

```
bor:- ar(13,R),tw(10),R=3,exit.            % выход из системы

% Если код 27, то выходим в главное меню
bor:- ar(K,R),K=27,retractall(ind(_)),assert(ind(1)),
      retractall(ar(_,_)),assert(ar(80,1)), an(K,R),  fail,!.

% Если вопросы закончились (их 7, т.е. W=8), то выводим результаты экспертизы
bor:- ind(Y),Y=2,tw(W),W=8, wiwex,
      retractall(ind(_)),assert(ind(1)),
      retractall(ar(_,_)),assert(ar(80,1)),
      an(27,1),fail,!.

bor:- ind(Y),Y=2,q(X),X=1,retractall(ar(_,_)),assert(ar(80,1)),an(27,1),fail,!.

bor:-!,bor.

% Коды клавиш стрелок вверх и вниз
k(72).k(80).

% Анализ нажатия клавиши Esc, если нажата, то
%  возвращаемся в главное меню с установкой исходных параметров
an(K,R):-K=27,ind(Q),Q=1,retractall(q(_)),
  shiftwindow(1),shiftwindow(2),
  retractall(mn(_,_)),assert(mn(10,10)),  menu,
  retractall(mn(_,_)),po(10,1),
  retractall(tw(_)),assert(tw(10)),
  retractall(ar(_,_)),assert(ar(80,1)),!.

% если режим опроса,то устанавливаем параметры и в итоге активизируем окно 4
an(K,R):-K=300, retractall(ind(_)),assert(ind(2)),
       retractall(tw(_)),assert(tw(1)),
  retractall(rom(_,_)),asrom(1,7),
  retractall(ar(_,_)),assert(ar(80,1)),
  shiftwindow(3),wiw,shiftwindow(4),clearwindow,sob,!.

an(K,R):-K=500,
  makewindow(5,30,112,"КРАТКИЕ СВЕДЕНИЯ О СИСТЕМЕ ( Выход - Esc ... )
",3,5,12,70),nl,nl,
  Sf="prim2_es.hlp",existfile(Sf),file_str(Sf,S),display(S), removewindow,
  retractall(ind(_)),assert(ind(2)),retractall(ar(_,_)),assert(ar(80,2)),!.


an(K,R):-ind(Q),Q=0,removewindow,removewindow,removewindow,removewindow,
        removewindow,removewindow,removewindow,removewindow,exit.

an(K,R):-ind(O),O=2,
        K=100,tw(W),W>4,W<8,W<>6,
        shiftwindow(4),clearwindow,sob,
        uz(W,Y,P2,_),cursor(Y,2),write(P2),
        retractall(mn(_,_)),assert(mn(W,W)),
```

```
        menu,retractall(mn(_,_)),
        field_attr(Y,16,1,240),
        po(W,1),retractall(ar(_,_)),assert(ar(80,1)),!.

an(K,R):-ind(O),O=2, K=13,tw(W),W>4,W<8,W<>6,
        retractall(rom(W,_)),
        Ws=W+1,Rs=R-1,assert(rom(W,Rs)),
        retractall(tw(_)),assert(tw(Ws)),
        shiftwindow(3),wiw,gotowindow(4),
        retractall(ar(_,_)),assert(ar(80,1)),!.

an(K,R):-k(K),tw(W),W>4,W<>6,W<8,uz(W,W2,S,_),
    str_len(S,Lw),
    o(W,R,A,Y,X),d(W,R,K,Rs),
    o(W,Rs,As,Ys,Xs),d(W,Rs,Ks,_),str_len(A,L),str_len(As,Ls),
    field_attr(Y,X,L,c1),field_attr(Ys,Xs,Ls,col),
    retractall(ar(_,_)),assert(ar(K,Rs)),!.

an(K,R):-k(K),tw(W),W>8,
    o(W,R,A,Y,X),d(W,R,K,Rs),
    o(W,Rs,As,Ys,Xs),d(W,Rs,Ks,_),str_len(A,L),str_len(As,Ls),
    field_attr(Y,X,L,c1),field_attr(Ys,Xs,Ls,col),
    retractall(ar(_,_)),assert(ar(K,Rs)),!.

an(K,R):-K=13,tw(W),W=10,R=1,retractall(ind(_)),assert(ind(2)),
         retractall(ar(_,_)),assert(ar(K,R)),!.

an(K,R):-K=13,tw(W),W=10,R=2,
        retractall(ind(_)),assert(ind(3)),
        retractall(ar(_,_)),assert(ar(K,R)),!.

an(K,R):-K=13,tw(W),W=10,R=3,retractall(ar(_,_)),assert(ar(K,R)),!.

an(K,R):- ind(O),O=2, q(U),U=1,retractall(q(_)),
         retractall(ind(_)),assert(ind(1)),
         retractall(ar(_,_)),assert(ar(K,R)),!.

an(K,R):-ind(O),tw(W),O=2,W<7,W<>5,
        shiftwindow(4),clearwindow,sob,
        uz(W,Y,P2,_),cursor(Y,2),
        write(P2), field_attr(Y,16,1,240),
        cur(W,Y1,X1),cursor(Y1,X1),
        let(W,Lt),qduk,
        dbl(W,Lt), retractall(tw(_)),
        W1=W+1,assert(tw(W1)),
        shiftwindow(3),wiw,gotowindow(4),
        retractall(ar(_,_)),assert(ar(100,1)),!.

an(K,R):-ind(O),O=2, tw(W),W<7,W<>5,
        retractall(ar(_,_)),assert(ar(27,1)),!.
```

```
an(K,R):-ind(O),O=3,K=13,tw(W),W>4,W<8,W<>6,
        retractall(rom(W,_)),
        Rs=R-1,assert(rom(W,Rs)),
        retractall(ar(_,_)),assert(ar(80,R)),
        fail, !.

an(K,R):-ind(O),O=3,K=27,tw(W),W>4,W<8,W<>6,
        retractall(tw(_)), assert(tw(20)),
        retractall(ar(_,_)),assert(ar(80,W)),
        shiftwindow(2), !.

an(K,R):-ind(O),O=2,K=27,tw(W),W>4,W<8,W<>6,
        retractall(q(_)),assert(q(1)),!.

an(K,R).

sob:-cursor(18,0),S="Esc - выход",write(S),
    str_len(S,L),field_attr(18,0,L,112),!.
sob.

qduk:-not(q(_)),!.
qduk:-q(V),V<>3,!.

% преобразуем и добавляем ответ в базу данных
dbl(R,Lt):-str_int(Lt,N),retractall(rom(R,_)),assert(rom(R,N)),!.
dbl(R,Lt).

% основное правило для подсчета количества баллов по каждому вопросу
% и обновление результирующей базы
preobr:- rom(1,X1),rom(2,X2),rom(3,X3),
        rom(4,X4),rom(5,X5),rom(6,X6),rom(7,X7),
        M=4.2*X1+3.6*X2+4.5*X3+222*X4+68*X5-78*X6+224*X7,
        str_int(S1,X1),str_int(S2,X2),str_int(S3,X3),
        str_int(S4,X4),str_int(S5,X5),str_int(S6,X6),str_int(S7,X7),
        str_real(S8,M), vero(M,S9),
        wpw(H1,H2,H3,H4,H5,H6,H7,H8,H9),
        retractall(wpw(_,_,_,_,_,_,_,_,_)),
        assert(wpw(S1,S2,S3,S4,S5,S6,S7,S8,S9)),!.
preobr.


cur(1,9,16).cur(2,10,16).cur(3,10,16).cur(4,10,16).cur(6,11,16).

let(W,X):-  retractall(aw(_,_)),assert(aw(1,"")),
          rom(W,U),lop(W),aw(E,X),!.


% для защиты по количеству вводимых цифр при ответе на вопросы
lb(1,2).lb(2,2).lb(3,3).lb(4,1).lb(6,2).
```

```
% факты, ограничивающие диапазон ввода цифр для всех вопросов,
% кроме 5-го и 7-го
lb1(1,100). % для количества лет, т.е. по 1-му вопросу и т.д.
lb1(2,X):-rom(1,X),!.
lb1(3,301).
lb1(4,10).
lb1(6,169).

% основное правило описывающее интерфейсную часть
% диалогового окна (анализ и обработка нажатых клавиш)
% и защита от случайного ввода

  % анализ клавиш цифр (коды 48 - 57) и формирование строки ответа
  % позволяет ввести только цифры

lop(W):-ind(G),G=2,aw(X,Q),read_kod(K),retractall(aw(_,_)),
     assert(aw(K,Q)),K>=48,K<=57,char_int(C,K),
    str_len(Q,L),lb(W,M),L<M,
    write(C),str_char(S,C),concat(Q,S,S1),
    retractall(aw(_,_)),assert(aw(K,S1)),fail.

  % обработка клавиши Esc (код 27) - выход из цикла
lop(W):-ind(G),G=2,aw(X,Q),X=27,retractall(q(_)),assert(q(1)),!.

  % обработка клавиши Enter (код 13) - выход из цикла
lop(W):-ind(G),G=2,aw(X,Q),X=13,str_len(Q,L),lb(W,M),L<=M,str_int(Q,M1),!.

  % обработка клавиши Backspase (код 8)
lop(W):-aw(X,Q),X<>13,X=8,str_len(Q,L),L>0,
        cur(W,E1,E2), cursor(E1,E2),write("       "),
        retractall(aw(_,_)),assert(aw(1,"")),
        cursor(E1,E2), rom(W,U), fail.
  % зацикливаем правило
lop(W):-lop(W).


% правила для изменения цвета пунктов меню на экране
ro(W,R):-o(W,R,A,Y,X),str_len(A,L),field_attr(Y,X,L,58),!.
po(W,R):-o(W,R,A,Y,X),str_len(A,L),field_attr(Y,X,L,col),!.
po(W,R).

rw(R):- uz(R,W,S,_), str_len(S,L),field_attr(W,2,1,62),!.
pw(R):- uz(R,W,S,_), str_len(S,L),field_attr(W,2,1,240),!.

% вывод информации о пройденном пути
wiw:- clearwindow,preobr, wpw(H1,H2,H3,H4,H5,H6,H7,H8,H9),
      cursor(2,0),
        write("\n            ВОЗРАСТ : ",H1,"\n            КУРИТ(Л) : ",H2),
        write("\n     ВЕРХНЕЕ ДАВЛЕНИЕ : ",H3,"\n КОЛИЧЕСТВО ИНФАРКТОВ : ",H4),
```

```
        write("\n     НАСЛЕДСТВЕННОСТЬ : ",H5,"\n              СПОРТ : ",H6),
        write("\n       БОЛИ В СЕРДЦЕ : ",H7), !.


% Вывод результатов экспертизы
wiwex:-  preobr, wpw(H1,H2,H3,H4,H5,H6,H7,H8,H9),
   makewindow(10,0,0,"",17,22,3,38),
   makewindow(11,32,31,"ВЫВОДЫ ЭКСПЕРТНОЙ СИСТЕМЫ ",16,20,3,38),
   write("     Вероятность инфаркта - ",H9,"%"),readchar(_),
   removewindow,removewindow, !.

% Основные правила, определяющие вероятность инфаркта
% по набранным баллам
vero(M,S):-M<570,S="0",!.
vero(M,S):-M>=570,M<=595,S="5",!.
vero(M,S):-M>595,M<=810,S="15",!.
vero(M,S):-M>810,M<=860,S="35",!.
vero(M,S):-M>860,M<=920,S="55",!.
vero(M,S):-M>920,M<=1100,S="70",!.
vero(M,S):-M>1100,M<=1800,S="90",!.
vero(M,S):-M>1800,S="100",!.
vero(M,S):-S="",!.

% База вопросов
% Имеет формат:
% uz(N вопроса; координата по У; вопрос; кол. пунктов,если вопрос по меню)

uz(1,3,"          1\n\n\n\n     Сколько  вам  лет ?",1).
uz(2,3,"          2\n\n\n\n    Сколько  лет вы курите\n       или  курили  ?",1).
uz(3,3,"          3\n\n\n\n    Какое  у  вас  верхнее\n    артериальное давление ?",1).
uz(4,3,"          4\n\n\n\n    Сколько  было  у  Вас\n       инфарктов ?",1).
uz(5,3,"          5\n\n\n\n    Имелись  ли  сердечно-\n    сосудистые заболевания\n       у
родителей ?",3).
uz(6,3,"          6\n\n\n\n    Сколько часов в неделю\n    Вы занимаетесь спортом\n       (
физкультурой ) ?",1).
uz(7,3,"          7\n\n\n\n    Есть ли  боли в сердце\n    при физической нагрузке ?",2).
uz(10,1,"",3).

% Описывают переход курсора при нажатии клавиш стрелок
% Факты для работы меню по 5-му вопросу
  d(5,1,80,2).d(5,1,72,3).d(5,2,80,3).d(5,2,72,1).d(5,3,80,1).d(5,3,72,2).

% Факты для работы меню по 7-му вопросу
  d(7,1,80,2).d(7,1,72,2).d(7,2,80,1).d(7,2,72,1).

% Факты для работы основного меню
  d(10,1,80,2).d(10,1,72,3).
  d(10,2,80,3).d(10,2,72,1).
  d(10,3,80,1).d(10,3,72,2).
```

```
% Нижеперечисленные факты меню имеют следующий формат:
% o(код меню, номер пункта меню, имя пункта, координата по Y и X)

% Факты-пункты меню по 5-му вопросу
o(5,1,"НЕ ИМЕЛИСЬ",11,8).
o(5,2,"У ОДНОГО РОДИТЕЛЯ",12,8).
o(5,3,"У ОБОИХ РОДИТЕЛЕЙ",13,8).

% Факты-пункты меню по 7-му вопросу
o(7,1,"НЕТ",10,15).
o(7,2,"ДА",11,15).

% Факты-пункты главного меню
o(10,1,"РЕЖИМ ОПРОСА",1,5).
o(10,2,"КРАТКИЕ СВЕДЕНИЯ О СИСТЕМЕ",2,5).
o(10,3,"ВЫХОД",3,5).

% Правила, позволяющие читать простой и расширенный коды
  read_kod(Kod):- readchar(C), char_int(C,A),char_kod(A,Kod),!.
  char_kod(A,Kod):- A<>0,Kod=A,!.
  char_kod(0,Kod):- readchar(C),char_int(C,Kod),!.
```