MEDICAL DIAGNOSTIC EXPERT SYSTEM

A DISSERTATION

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE OF
MASTER OF TECHNOLOGY
IN
COMPUTER SCIENCE AND TECHNOLOGY

BY

MUSBAH MAH'D AQEL

SCHOOL OF COMPUTER AND SYSTEM SCIENCES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI - 110067
INDIA

1989

298

671

MEDICAL DIAGNOSTIC EXPERT SYSTEM

A DISSERTATION

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE OF
MASTER OF TECHNOLOGY
IN
COMPUTER SCIENCE AND TECHNOLOGY

BY
MUSBAH MAH'D AQEL

SCHOOL OF COMPUTER AND SYSTEM SCIENCES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI - 110067
INDIA
1989

DEDICATED TO

MY PARENTS. BROTHERS & SISTERS

AND

MY WIFE & SON

JAWAHARLAL NEHRU UNIVERSITY

NEW DELHI -110067

CERTIFICATE

It is certified that the contents of this dissertation which carries the title MEDICAL DIAGNOSTIC EXPERT SYSTEM has been submitted by MUSBAH AQEL, has not been previously submitted for any other degree of this or any other university

[illegible]

Prof. M.P. MUKHERJEE
(DEAN)
SCSS/JNU

[illegible]

DR. P.C. SAXENA
(Supervisor)
SCSS/JNU

[illegible]

MUSBAH AQEL
(Student)

ACKNOWLEDGEMENT

I am highly indebted to my Supervisor Dr. P.C. Saxena, Associate Professor. School of Computer Sciences, Jawaharlal Nehru University, for his eminent guidance, constant supervision and whelming encouragement throughout my Dissertation work.

I am grateful to all the teaching and non-teaching staff in the School of Computer Sciences, J N U. Department of Computer Science, I.I.T. New Delhi, Department of Medicine in AIIMS for providing me help.

I am also thankful to the Library Staff of J N U, I.I.T.D. and AIIMS, for their Cooperation. Finally my gratitude can not be expressed in words to my wife (Dr.) Aziza Aqel and my son Alaa for their immense patience, cooperation and help during the completion of my Dissertation and My. M.Tech. Study .

PREFACE

According to survey (Handbook of Health Information of India, 1986), the people in India have lowest medical attention in the world. Besides, there is an enormous maldistribution of medical experts, majority of the expert is concentrated in big towns and cities. Moreover, due to poverty, 70% people ill-afford the cost of modern medical care. Clinical / pathological Laboratories are few in numbers and most of them are situated in metropolitan and first grade cities. Many poor people in India only because of the little knowledge of self-made or certificate and diploma holder doctors who are ill-equipped with the techniques of modern medical devices and treatment. Therefore an expert system, has been developed by the present investigator to help non-expert physicians to diagnose all types of fevers according to their respective symptoms. The important feature of this expert system is that it not only diagnoses the disese but also prescribes the treatment with additional advice of prevention, precaution, and comments as required. Another noteworthy characteristic of this system is that there is a room for the inclusion of the symptoms of any new fever at any time to be known in near future.

The first period of AI research was dominated by the belief that a few laws of reasoning coupled with powerful computers would produce expert and superhuman performance. As experience accured, the severely limited power of general-purpose problem-solving strategies ultimately led to the view that they were too week to solve most complex problems. In reaction to perceived limitations in the overly general strategies, many researchers began to work on narrowly defined application problems. By the mid-1970s several expert systems had begun to emerge such as DENDRAL, MYCIN, PROSPECTOR etc. These systems undoubtedly provided a high level of expertise, but at the same time its development required a team of trained people in specific field and knowledge engineering. Designed around a vast amount of knowledge these systems occupied large spaces of memory in the computer. As result these early knowledge based systems were implemented on mainframe computers only. Consequently, it was not feasible for small organisations and individuals to make use of the technology of expert system with full potential because of the high cost of mainframe computers. In order to cope with the problems, recently much attention has been paid by the computer scientists to develop small expert systems instead of large expert systems which can be run on microcomputers. This switchover from large expert systems to small expert systems will be very useful for small organization and individual. This expert system is also a step in this direction which can be run on any IBM PC or compatible computer with a memory of 640k RAM.

C O N T E N T S

PAGE NO

CHAPTER - 1

EXPERT SYSTEM

1.1. INTRODUCTION

Humanity has dreamed of wisdom-spouting machines since ancient times The Greek Oracles were supposed to produce recommendations for action based on wisdom supplied by the gods. Our contemporary counterpart, the computerized expert system, is a near-fulfilment of this age-old dream Though we often think of the expert system as a new technology, its design concepts are as ancient as the human species. Early scholars tried to systematise all human knowledge to unravel the engine of cognition. The Babylonians developed an elaborate set of "if-then' rules describing the empirical associations between observable phenomena in every-day life. many of these constituted an early attempt to develop a system of medical diagnosis.

The subjectivity of empiricism was a favourite theme of Descartes, who believed that empirical knowledge might be deduced with the same degree of certainty as mathematical theorems. It was John Dewey, however, who most influenced today's expert system programmers. Deway saw logic not as an academic but as concrete method of inquiry to be applied to human problems.

He wrote: "The process of arriving at an idea of what is absent on the basis of what is at hand is inference. Every inference, just because it goes beyond the ascertained and known facts, which are either given by observation or by recollection of prior knowledge, inovlves a jump from the known into the unknown". Dewey further stressed the necessity of validating inference and distinguishing between empirical cause-effect relationships and productive ones. "What is important," he contended, "is that eery inference be a tested inference." In these three statements, he effectively described the modern rule-based expert system [Sawyer and Foster, 1986].

Important conceptual ground was laid at Carnegie-Mellon by Alan Newell and Herbert Simon [1963] , who set out to map the processes that underlie human solving. The program they rpoduced was called GPS (General Problem Solving), and it was the first AI program to use something resembling rules. Again, no attempt was made to create a program with a large amount of knowledge about a particular problem domain. Only quite simple tasks were selected. But after 1970s, researches on AI progressed and techniques for handling large amounts of world knowledge were developed. These include perception (vision and speech), natural language understanding, and problem solving in specialized domain such as medical diagnosis, chemical analysis, and engineering design etc. [Rich, 1983]. A very good pictorial model comprising different fields of applications was proposed by N J. Nilson . [1981-82] which he calls the "Oniion Model" as shown in Fig.1.1. The inner ring of the model depicts the basic elements from which the applications shown in the next ring of model are composed. The present work is concerned with one of the constituent of onion model, i.e., the expert system. However, the first expert system DENDRAL [Lindsy et al., 1980] was developed at Standford University in mid 1960s, but by far the most famous precursor to modern rule-based expert system emerged from Standard University in mid 1960s, but by far the most famous precursor to modern rule-based expert system emerged from stanford Heuristic Programming Project (HPP) in 1974 [Buchanan and Shortliffe, 1983]. The MYCIN [Shortliffe, 1976] experiment integrated many of the previously espoused academic theories in such areas as symbolic mathematics [Newell and Simon, 1976], machine heuristics ¼Newell et al., 1963], and knowledge engineering]Feigenbaum, 1980].

As recently as 1980, expert system research was still confined to a few university research laboratories. Today, the United States, Japan, England and the European

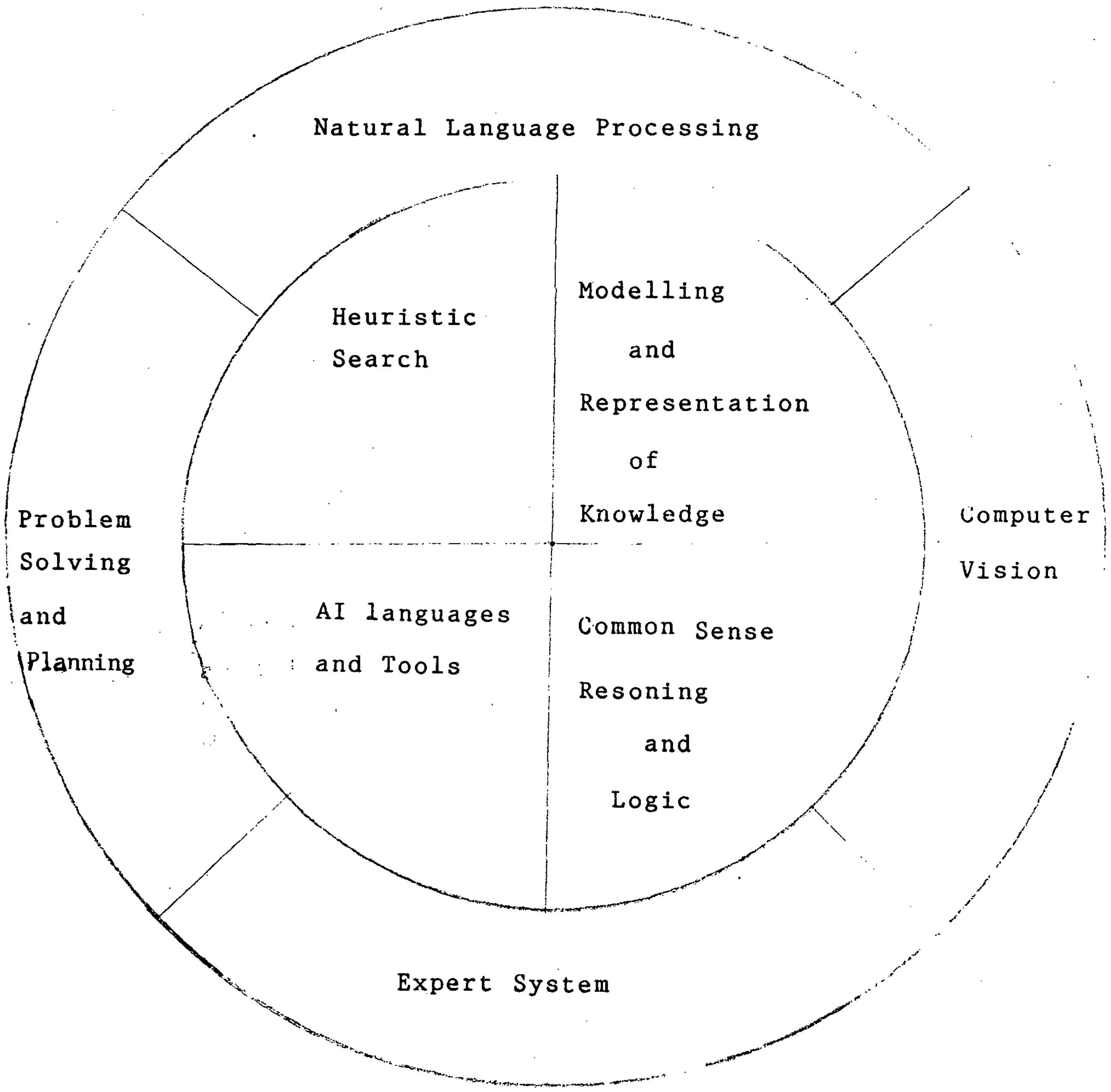
Natural Language Processing
Heuristic Search
Modelling and Representation of Knowledge
Problem Solving and Planning
Computer Vision
AI languages and Tools
Common Sense Resoning and Logic
Expert System

Fig. 1.1. Elements of AI

Economic community are all in the process of launching major research programmes to develop and implement expert systems in near future. Many corporations are assembling AI departments, venture capitalists are rushing to invest in entrpreneurial expert syste companies, and expert systems technology is well on its way to commercial success.

1.2. Expert System:

Much of the current research in artificial intelligence (AI) focuses on computer programmes that help people with complex reasoning tasks. Recent work has indicated that one key to the creation of intelligent systems is the incorporation of large amounts of task-speific knowledge. Thus, researchers have recently created knowledge-based systems which exploit the specialized knowledge of experts to achieve high performance in a specific problem domain [Buchanan and Duda, 1983].

Knowledge in any specially is usually of two sorts; public and private. Public knowledge includes the published definitions, facts, and theories of which text-books and references in the domain of study are typically composed. But expertise ususally involves more than just this public knowledge. Human experts generally possess private knowledge that has not found its way into the published-literature. This private consists largely of rules of thumb that have come to be called heuristics [Lenat,1982]. Heuristics enable the human expert to make educated guesses when necessary to recognize promising approaches to incomplete data Elucidating and reproducing such knowledge is the central tasks in building expert systems.

Various definitions of Expert System can be found in the literature, some of them are given below:

1) Feigenbaum [1982] states: "An 'expert system' is an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solution. The knowledge necessary to perform at such a level, plus the inference procedures used, can be thought of as a model of the expertise of the best practitioners of the field."

2) d'Agapeyeff [1983] has defined expert system as: "Problem solving programmes that solve substantial problems generally conceded as being difficult

and requiring expertise. They are called knowledge-based because their performance depends critically on the use of facts and heuristics used by experts".

3) Buchanan [1982] defines expert system as : "a reasoning program that can be distinguished by other AI programmes in its utility, performance and transparency."

4) Nau [1983] has defined in another way: "Expert system is a problem-solving program whose performance compares with that of a human expert in a specialized problem domain. He points out that the main difference between application programmes and expert system is in that in most expert system themodel of problem-solving in the application domain is explicitly viewed as a separate entity named knowledge-base, rather than being implicitly embedded into the coding of the program. In addition, the knowledge base is manipulated by a separate and clearly identifiable control strategy".

5) The British Computer Society's Committee of the Specialist Group in Expert Systems [Simon, 1984] has defined expert system as : "The embodiment within a computer of a knowledge based component from an

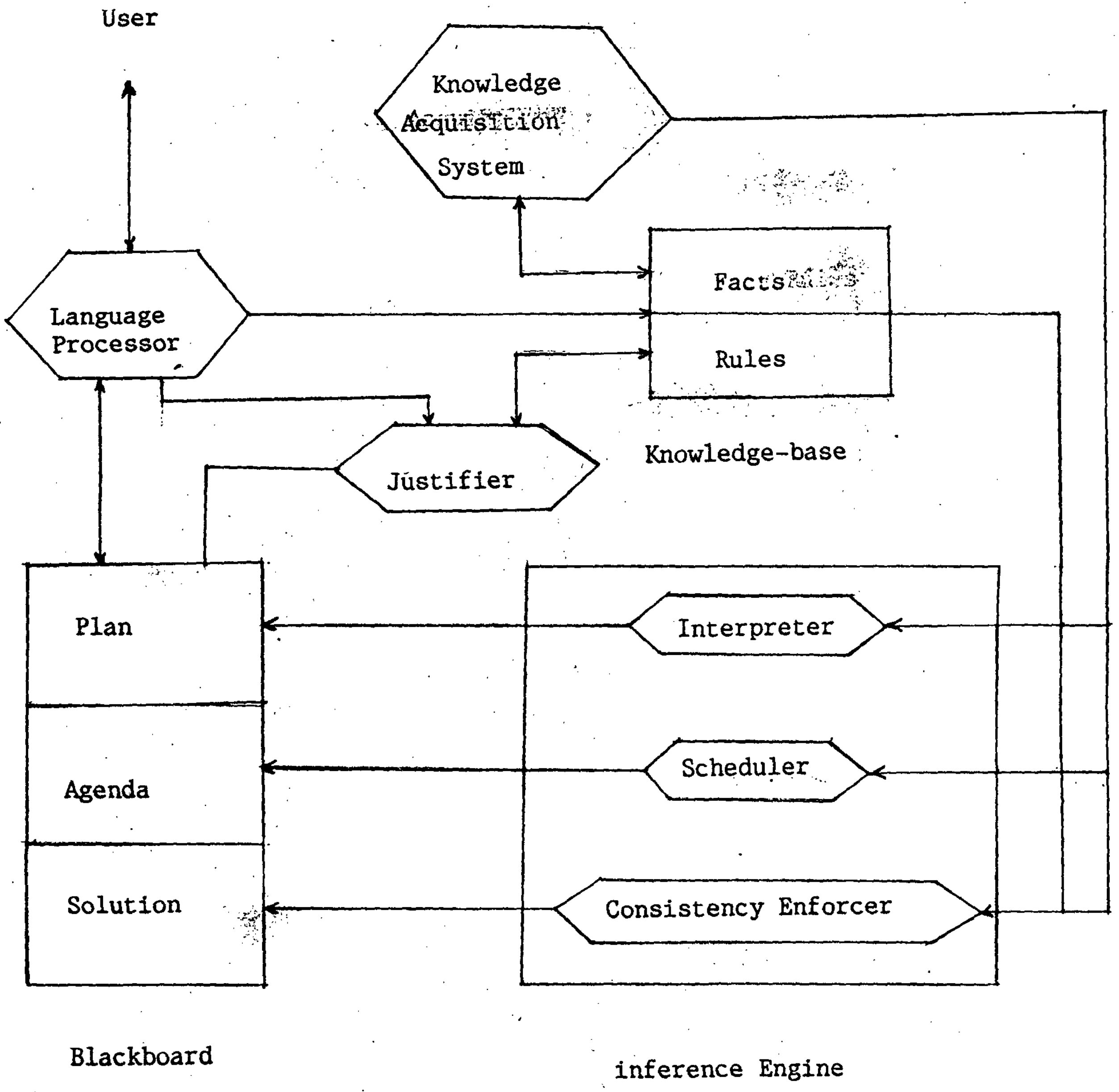

Fig. 1.2. Components of Expert Systems

expert skill in such a form that the machine can offer intelligent advice or take an intelligent characteristic, which many would regard as fundamental, is the capability of the system on demand to justify its own line of reasoning in a manner directly intelligible to the enquirer. the style adopted to attain these characteristic is rule-based programming."

In short, expert system or knowledge-based system is a domain specific application program that is designed to represent and apply human-expertise (of that domain) to solve real problems (of that specific field).

1.3. Components of Expert Systems:

Fig. 1.2 shows an idealized representation of an expert systsem. No existing expert system contains all the components shown, but one or more components occure in every expert system. Each component of this ideal system is described briefly in turn.

The language processor / user interface mediates information exchanges between the expert system and the human user. It is desirable, although not yet common, to have a user-friendly natural language interface to facilitate the use of the system. A natural language interface simulates casual conversation, using everyday, expressions in gramatically correct sentences. Obviously, the more natural, interface, the greater the demands on permanent storage and memory. Hence, systems with extraordinary user sensitivity are largely restricted to mainframe resources accessible from remote work station. Building tools designed to operate within the constraints of PC environment necessarily sacrifice "friendlilness" for the sake of efficiency.

The <u>knowledge acquisition</u> is a bottleneck in the construction of expert systems. Knowledge acquisition is the transfer and transformation of problem solving expertise from some knowledge source to a program. Potentil sources of knowledge include human experts, text books, data bases, and one's own experiences. Use of the knowledge acquisition module usually requires a partnership between a knowledge engineer and a domain expert in the given field. Ideally the expert should be able to convey his or her knowledge directly to the system. As knoledge acquisition system is a learning module of an expert system, very few existing expert systems incorporate such a module.

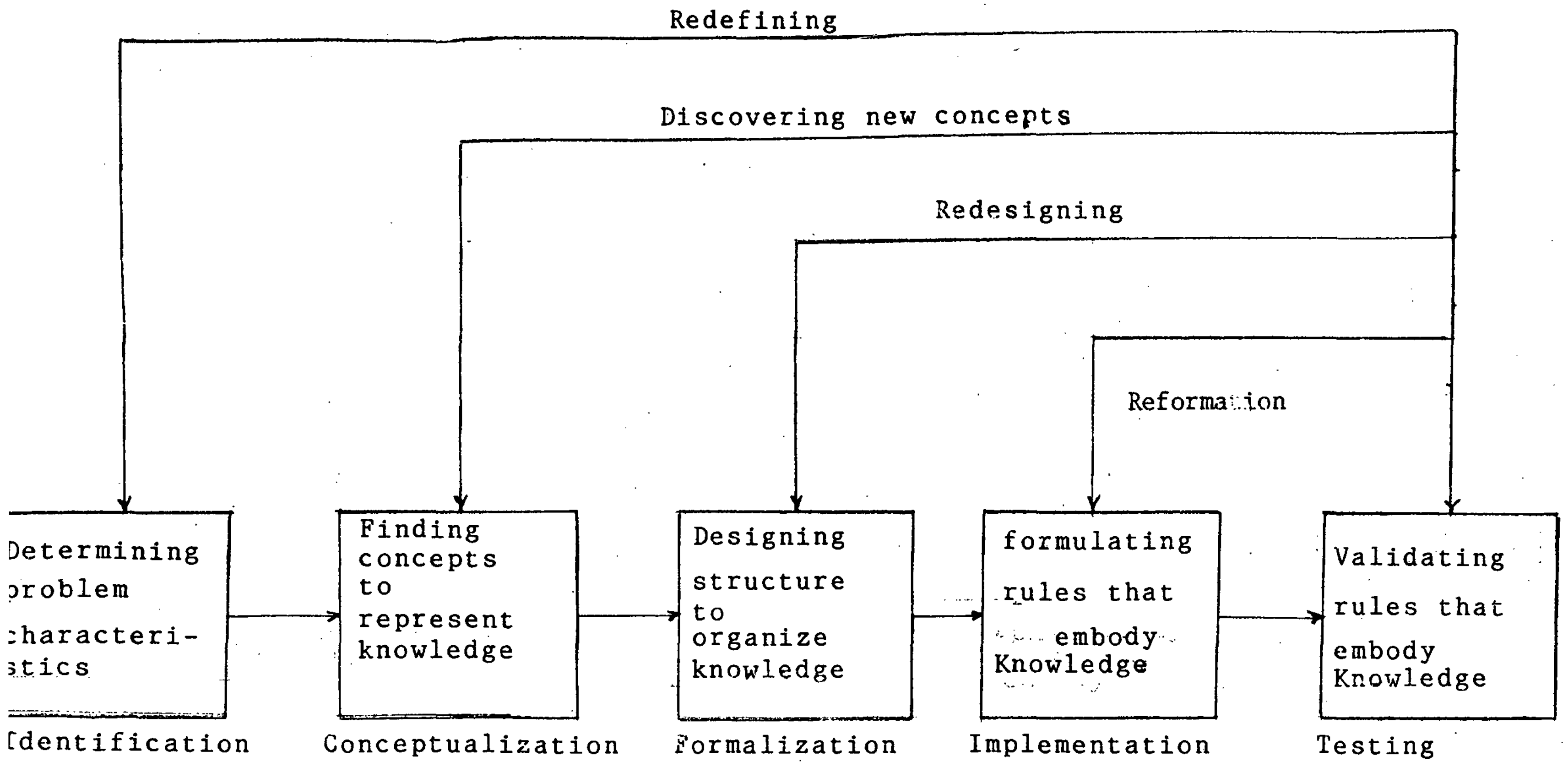
Redefining
Discovering new concepts
Redesigning
Reformation
Determining roblem haracteri- tics
Finding concepts to represent knowledge
Designing structure to organize knowledge
formulating rules that embody Knowledge
Validating rules that embody Knowledge
dentification
Conceptualization
Formalization
Implementation
Testing

Fig.1.3 Stages in the evaluation of an expert system

The <u>knowledge-base</u> contains the facts and rules that embody the expert's knowledge. The fact in the knowledge base describes what is known about the subject matter at the current time. The rule states the situational, conceptual, casual or precedental relationships among the facts. The expert system therefore must have a mechanism for feeding facts and rules into the knowledge base, maintaining repetoire of factual expressions, and displaying the contents for periodic review by the human engineers and operators of the system.

The <u>justifier/explanation system</u> explains the actions of the system to the user. In general, it answeres question about whysome conclusion was reahed or why some alternative was rejected. To do this, the justifier uses a few general types of question-answering plans. These typically require the justifier to trace backward along blackboard solution elements from the questioned conclusion to the intermediate hypotheses or data that support it. Each step backward corresponds to the inference of one knowledge base rule. Thejustifier collects these intermediate inferences and translates them into English for presentation to the user.

The <u>blakboard</u> / <u>dynamic data-base</u> records intermediat hypotheses and decisions that the expert system manipulates. Every expert system uses some type of intermediate decision representation, but only a few explicitly employ a blackboard for the various types of decisions shown in Fig. 1.2. The figure identifies three types of decisions recorded on the blackboard: plan, agenda, and solution elements. Plan elements describe the overall or general attach the system will pursue against the current problem, including current plans, goals, problem states, and contexts. The agenda elements records the potential actions awaiting execution, which generally correspond to knowledge base rules that seem relevant to some decision placed on the blackboard previosly. The solution elements represent the candidate hypotheses and decisions the system has generated thus far, along with the dependencies that relate decisions to one another.

The inference engine performs three tasks simultaneously and therefore contains three elements: interpreter, scheduler, and consistency enforcer. The interpreter executes the chosen agenda item by applying the corresponding knowledge base rule. Generally the interpreter validates the relevance conditions of the rule, binds variables in these conditions to particular solution blackboard elements, and then makes those changes to the blakboard tht the rule prescribes. The scheduler maintains control of the agenda and determines which pending action should be executed next. Schedulers may embody considerable knowledge to provide each agenda item priority according to its relationship to the plan and other extent solution elements. The consistency enforcer attempts to maintain a consistent representation of the emerging adjustment scheme to determine the degree of belief in each potential decision. This scheme attempts to ensure that plausible conclusions are reached and inconsistent ones are avoided.

1.4 The Construction ofExpert Systems:

Workers in expert systems conduct principally empirical research to determine how to solve a problem requiring extensive knowledge and skill. To fashion a solution, they must build a working system by exploiting two types of assets, a methodology and a set of tools.

1.4.1. A Methodology for Building Expert Systems:

The science of designing expert system is largely experimental. Every expert system should be considered an experiment, subject to validation on the basis of empirical results. An expert system therefore evolves gradually. Fig.1.3 shows the major stages in the evolution of an expert system.

During identification, the knowledge engineer and expert work together to identify the problem area and define its scope. They also identify the participants in the development process (additional experts), determine the resources needed (time, computing facilities), and decide upon the goals or objectives of building the expert system.

During conceptualization, the expert and knowledge engineer explicate the key concepts, relations, and information-flow characteristics needed to describe the problem-solving process in the given domain. They also specify subtasks, strategies, and constraints related to the problem-solving activity.

Formalization involves mapping the key concepts and relations into a formal representation suggested by some expert-system-tool or language. The knowledge engineer must select the language and, with the help of the expert, represent the basic concepts and relations within the language framework.

During implementation, the knowledge engineer combines and reorganizes the formalized knowledge to make it compatible with the information flow characteristics of the problem. The resulting set of rules and associated control structure define a prototype program capable of being executed nd tested.

Finally, testing involves evaluating the performance of the prototype program and revising it to conform to standards of excellence defined by experts in the problem domain.

The above stages of expert system development are not clear-cut, well-defined, or even independent. At best they characterise roughly the complex processes involved. If during the testing stage the performance of the system is not satisfactory, the knowledge engineer team advise for reformulating rules and control processes, redesigning knowledge structure, discovering new concepts, and even redefining the problem's scope and goals.

1.4.2. Tools for Building Expert Systems:

Table 1.1. lists the primary tools presently available for knowledge engineering tasks. Except for *OPS*,

Table 1.1. Tools for building expert systems

Tool	Developer	Features
ROSTE	Rand Corporation	Rule-based, Procedure-oriented, General Purpose
OPS5	Carnegie Mellon University	Production-system formation, General purpose
EMYCIN	Stanford University	Rule-based, Diagnosis and explanation
KAS	Standford Research Institute	Rule-baed, Diagnosis and explanation
AGE	Stanford University	LISP-based, Builds various PS architectures
INTERLISP	xerox Corporation	LISP programming environment

Table 1.2. Choosing an appropriate tool for building an expert system

Issues	Maxims
Generally:	Pick a tool with only the generality necessary to solve the problem.
Selection :	Let the problem characteristics determine the tool selected.
Speed ;	When time is critical, choose a tool with built-in explanation/interaction facilities.
Testing :	Test the tool early on by building a very small prototype system.

they operate in the INTERLISP environment. ROSIE is an excellent general-purpose tool for building prototype expert systems. EMYCIN, KS300, and KAS are very useful general tools for diagnostic tasks. KAS provides facilities akin to those of EMYCIN but has not yet been used in the variety of ways EMYCIN has. KS 300 is an industrial system based on the EMYCIN methodology. OPS is the most powerful pure production-system interpreter available. AGE provides the best extant toola for developing a variety of different architetures. INTERLISP provides a LISP programming environment preferred by the most workers in the field.

Despite the wealth of knowledge accumulated about constructing expert systems, choosing an appropriate tool for building a particular system remains a difficult yet crucial task. A tool that in some sense well suits a particular problem area can facilitate the development process, shorten the development time, and lead to a finished product that performs with a high degree of efficiency.

Table 1.2 provides guidelines for choosing an appropriate expert-system-building tool. The predominant consideration involves matching the problem characteristics to necessary tool features and hence particular tools.

1.4.3. Expert System Shells

A number of products are being offered commercially as expert system shells. Unlike a complete expert system, a shell lacks one of the three key ingredients of an expert system : it has no knowledge base (it has the inference engine part and the user interface part only). Instead, an expert system shall provides a sub-system called a knowledge base builder's interface that helps the user build a knowledge base. (Sometimes a knowledge base builder's interface is also provided with complete expert system so that the user can modify or expand the knowledge base).

The idea behind an expert system shell is that the user can produce a true expert system for whatever problem domain he wants by "filling" the shell with the expert knowledge needed for his application. These shells can simplify the task of building an expert system, but there is a price to be paid. Different problem domains may require different kinds of reasoning using different kinds of knowledge, we might also want different kinds of user interfaces for different domains. Thus, merely, filling in the knowledge base may not be enough to build a satisfactory expert system.

The less expensive expert system shells usually support only one kind of inference. It is difficult or impossible to alter either the inference engine or the user interface provided by these shells.

Other expert system shells provide inference engines with wide range of capabilities and provide user interface with great flexibility. These tend to be very expensive.

The alternative to using a shell for our expert system development is to build the inference engine and user interface we need in some Programming Language . Prolog is a good choice for this because it is comes with a built- in inference engine.

1.5. Types of Expert Systems:

Today, there are thousands of expert systems working in different relevant fields. A deail lists of major existing expert systems can be found in [Gevarter, 1985]. On the bais of applications, expert system fall into a few distinct types as shown in Table 1.3.

Table 1.3 Types of expert systems based on their applications

Category	Problem addressed
Interpretation	Inferring situation description from sensor data
Prediction	Inferring system malfunctions from observables
Design	Configuring objects under Constraints
Planning	Designing actions.
Monitoring	Comparing observations to plan vulnerabilities.
Debugging	Prescribing remedies for mal-functions
Repair	Executing a plan to administer a prescribed remedy
Instruction	Diagnosing, debugging, and repairing student behaviour
Control	Interpreting, predicting, repairing, and monitoring system behaviours.

Interpretation Systems infer situation description from observables. This category includes surveillance, speech understanding, image analysis, chemical structure elucidation, signal interpretation, and many kinds of intelligence analysis.

Prediction systems infer likely consequence from given situations. This category includes forecasting, demographic predictions, traffic predictions, crop estimations, and military forecasting.

Diagnosis systems infer system malfunctions from observables. This category includes medical, electronic, mechanical, and software diagnosis, among others.

Design systems develop configurations of objects that satisfy the constraints of the design problems. Such problems include circuit layout, building design, and budgeting.

Planning systems include design actions. These systems specialize in problems of design concerned with objects that perform functions. They include automatic programming as well as robot, project, route, communication, experiment, and military planning problem.

Monitoring systems compare observations of system behaviour to features that seem crucial plan plan outcomes. Thse crucial features correspond to potential flaws in the plan. Many computer-aided monitoring systems exist for nuclear power plant, air traffic, disease, regulatory, and fiscal management tasks.

Debugging systems prescribe remedies for malfunctions. These systems rely on planning, design, and prediction capabilities to create specifications or recommendations for correcting a diagnosed problem. Computer-aided debugging systems exist for computer programming in the form of intelligent knowledge base and text editors, but none qualifies as a expert system.

Repair systems develop and execute plans to administer a remedy for some diagnosed problem. Such systems incorporate debugging, planning, and execution capabilities. Computer-aided systems occur in the doains of automotive, network, avionic, and computer maintenance, as well as others, but expert systems are just entering this field.

Instruction systems diagnose and debug student behaviours. They incorporate diagnosis and debugging subsystems that speifically address the student as the system of interest.

The last type of system to be considered is called control. An expert control system adaptively governs the overall behaviour of a system. Problems addressed by control systems include air traffic control, business management, battle management, and mission control.

1.6. Expert Systems in Medicine:

Computers have been used for medical decision-making for about twenty years, employing programs that carried out well-established procedures. In the main, the program focussed on the diagnostic element in consultation. Once symptoms had been presented, the computer would select one disese from a fixed set, using methods such as pattern recognition through discriminant functions, Bayesian decision theory, and decision-tree techniques. In more complex programmes, sequential diagnosis was carried out. The accuracy of a computer-based diagnostic systems have performed better than medical consultants, and it is likely that automatic diagnostic system will be increasingly common in various medical areas. At the same time it is important to recognize the limitations of computer-based medical systems.

There are now many operating expert systems in medicine. Some prominent medical expert systems are: MYCIN [Shortliffe, 1974], CASNET [Weiss & Kulikowski, 1979], CADUCEUS [Pople et al., 1975], GLAUCOMA [Kulikowski et al., 1973], GUIDON [Clancey, 1983], NEOMYCIN [Clancey & Lesinger, 1981], PUFF [Feigenbaum, 1977], etc.

CHAPTER - II

SYSTEM DESIGN

2.1. Introduction:

The science of designing expert system is largely expreimental. What makes it so is its modus operandi of "leaping from the known into unknown." Every expert system should be considered an experiment, subject to validation on the basis of empirical results[Sawyer and Foster, 1986].

The develoment of expert systems has three special design criteria:

1. The domain.
2. The expert.
3. The user environment.

These considerations interact through three general phases, as illustrated in Fig. 2.1.

The expert system should begin with a thorough analysis of its objectives. Designers must know the nature of the problem to be solved. They must be able to describe the problem, the effects or results expected from the solution. The idea is to eliminate ambiguity and facilitate the development of specific measures that my validate, expand, and redefine the system.

Phase I	Planning
The	.Specify Objectives
Domain	.Define Problems and Sub problems
	.Develop Control and Measures
Phase II	Knowledge Engineering
The	. Select Expert
Expert	.Extract Knowledge
	.Develop Knowledge Base
Phase III	Implementation
The	. Programming
User	. Preliminary Testing
	. Refinement

Fig. 2.1. The Three Phases of Developing an Expert System

Once the objectives have been defined, the types of problem that need to be solved and the manner in which the program should approach them become readily apparent. In this phase as well, the need for specificity is great. Although the knowledge-base should be exhaustive in detail, it should also be limited to the facts and rules required to achieve the stated objectives. For instance, a heart patient's age and weight are more important is designing a therapeutic regimen for heart disease, but the patient's eye colour and hearing activity are not.

The second phase involves acquiring, structuring, and translating the body of expertise required to solve the identified problems. Both domain experts and knowlede engineers are ususally involved in the process. The complexity of the problem often determines the type and number of experts. The planner should find an expert with the right type of knowledge for the stipulated objectives and problems. He should focus on expertise acquired form in-depth experience and repeated observations. Formost expert systems, experienced practitioners are more valuable than academicians. In building a large knowledge-based systems with numerous predefined sub-problems, it is often advisable to gather information at several levels of the organization, from rank-and-file to top management. In building a small knowledge-based systems with a limited problem scope, a single source is preferable.

The third phase of the design is concerned with the transfer of knowledge that involves the user. Input from end-user is required early in the design process to assure that the system will be operationally practical and provide maximum relevance to environment.

2.2. Languages and tools

Once human expertise has been successfully translated into a structured knowledge-base, the expert system programmer can begin to work. The most direct method of implementing a knowledge-base is to code it into the software.

Programming language play a pivotal role in expert system: they act as the interface between the problems to be solved (the application) and the hardware (the implementation). An ideal programming language should be efficiently implementable on available technology but should also be 'human-oriented', providing case of expression of problems and their solutions[S. Gregory, 1987].

The earliest programming languages were much more oriented towards the machine than the human programmer. These languages were simply abstraction of 'von-Neumann' organization of the machines on which they were implemented. Slowly development in new languages have raised the level of programming but these languages still inherit many of von-Neumann type of architecture. All such languages which use the concept of von-neumann style programming are called imperative or procedural languages. FORTRAN, COBOL, BASIC, C, Pascal, Ada, etc. are all examples of procedural languages.

To enhance the programmer's productivity and making the programming more human-oriented, a type of problem-oriented languages, called declarative languages, were developed. In declarative programming language, statements have a declarative interpretation and can be read as a formal description of a problem without recourse to the behaviour of the machine. This means that the program often acts as its own specification. Functional programming and logic programming are the major programming paradigms of declarative languages. Examples of functional languages are Pure Lisp, backus'FP, Hope, Val, and ld. Prolog is the best example of logic programming. However, some ideas of logic programming have been used in early AI languages QA3, PLANNER, MICROPLANNER, and CONNIVER, etc. (Wah & Li, 1986).

In general, three capabilities, namely action, description, and reasoning, are needed for an expert system language. Historically, language strong in one of these capabilities tended to be relatively weak in others. Prolog is a reasoning-oriented language that is limited by its inefficiency of description and action. Lisp retains some features of von-Neumann programming. Some new languages, such as Loglisp and QUTE which amalgamate Prolog and Lisp in natural ways, have been developed. On the other hand, to explore parallelism, the parallel version of Prolog and Lisp, such as parlog, Concurrent Prolog, and Concurrent Lisp, have been proposed. Recent efforts are aimed at automatic programming that will allow the program to be generated from a simple specification of the problem .

Expert system tools have already been discussed in chapter -1.

2.3. Prolog

The name prolog was taken from the phrase "Programming in Logic." The language was originally developed in 1972 by Alain Colmeraur and P. Roussel at the University of Marseilles in France. Since then it is gaining wide popularity throughout the world. It has been used for such varied applications as symbolic calculus, database system, computer aided design, natural language processing, abstract problem solving, and many areas of artificial intelligence. The importance of Prolog has recently been given a considerable boost by its choice by the Japanese for their "Fifth Generation Computer" system.

The suitability of logic for expressing both program and their specifications make it specially useful for program development. Although prolog, as a programming language, can be represented only from programming concepts, its most outstanding features are drawn from its being embedded in logic. The main interest of Prolog resides in its ability to let the programmer express definitions of relations with more emphasis on the definitions than on the ways to compute the relations. In other words, the Prolog approach is rather to describe known facts and relationships about a problems, than to prescribe the sequence of steps taken by a computer to solve problem.

2.4. Rule-Based Expert Systems:

Two types of expert systems can be built with Prolog. The first is known as rule-baed expert system or simply called production system, and the second is frame-based expert system [Townsend, 1987].

A rule-based system has two primary components: the knowledge base and the inference engine as shown in Fig.2.2.

2.4.1. The Knowledge-Base:

The knowledge-base is the data or knowledge used to make decisions. It consists of a set of clauses, where each clause is either a fact about the given information or a rule about how the solution may relate to or be inferred from the given facts. A brief introduction how the knowledge-base is developed using the facts and rules expressed in Prolog is given here just for the sake of illustration.

i) Facts:

Some examples of facts in English and their equivalent in Prolog:

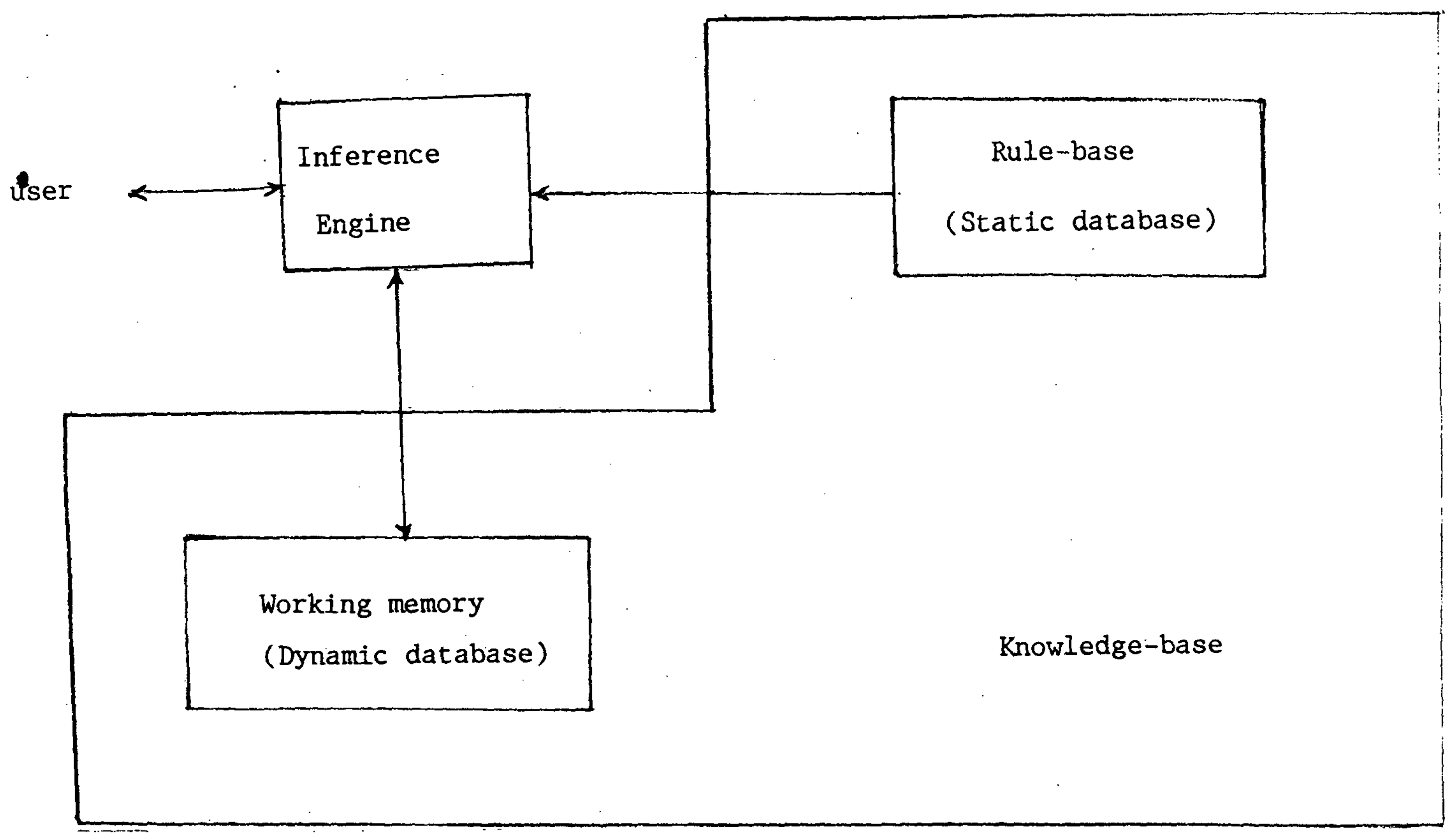
User
Inference Engine
Rule-base (Static database)
Working memory (Dynamic database)
Knowledge-base

Fig.2.2. Rule-based expert system

English	Prolog
1. Diana is female.	female (diana).
2. Albert likes Diana.	likes (albert,diana).
3. Albert and Diana. play badminton.	play (albert, diana, badminton).

II. Rules:

In Prolog, a rule consists of a head (or conclusion) and a body (or antecedent). The body consists of one or more premises. The premises in the antecedent form a conjunction of goals that must be satisfied for the conclusion to be true. If all the premises are true the conclusion is true; if any premise fails, the conclusion fails. As an example:

> If there is a body stiffness or pain the joints AND there is a sensitivity to infections, THEN there is probably a virtamin C deficiency.

This rule could be expressed as the followins Turbo Prolog clause:

Hypothesis (Vit C_deficiency) :-

Symptom (arthritis),

Symptom (infection_sensitivity).

2.4.2. The Inference Engine:

The inference engine has two functions; inferene and control. inference is the basis formal reasoning process. It involves matching and unification [Robinson, 1965; Boyer & Moore, 1972]; it uses facts that are known to derive new facts by means of rules. Such inference operates by modus ponens. This is basis for all formal logic.

The control function determines the order in which the rules are tested and what happens when a rule succeeds or fails. The inference engine takes the facts that it knows are true from the rule-base and working memory and uses these to test the rules in database through the process of unification. When a rule succeeds, the rule is said to fire, and the conclusion of the rule is added to working memory. In some cases more than one rule may unify with the known facts. Then the control function must determine which rule to fire. This is called conflict resolution [Kowalski & Keuhner, 1971]. However, the control function is a cyclic function.

Once a rule fires and the conclusion is added to the working memory, the unification process starts again, and the cycle is repeated.

i) Matching:

The term match has been used without proper definition in such phrases as "the condition part matches knowledge-base." There are many different types of matches that must be accomodated in a rule-based language. The simplest type is a match to an exact pattern of literals [Reitman & wilcox, 1978]. That is, a pattern in a clause in the left-hand side of a rule is identical to an element of knowledge-base. The following example-1 will illustrate this point:

```
                    /*Example-1 */
domains
            disease, indication = symbol
predicates
            symptom (disease, indication)
clauses
          symptom (chicken_pox, high_fever).
          symptom (chicken_pox, chills).
          symptom (flu,chills).
          symptom (cold, mild_body_ache).
          symptom (flu,severe_body_ache).
symptom (cold, runny_nose).
symptom (flu, runny_nose).
symptom (flu,moderate_cough).
```

The clauses in this example are a simple collection of facts. As the program starts execution, the following prompt appears in the Dialog window:

```
Goal:
```

Turbo Prolog is asking for a goal. A goal is essentially a question. Specify the following goal:

```
Goal :symptom (cold, runny_nose)
```

Turbo Prolog will then respond with True and prompt for another goal. It means that Turbo Prolog matches the goal with a fact in the database. Enter the following goal:

```
Goal : Sympton (cold, headache)
```

Turbo Prolog responds with False. This response does not mean that a headache is not a cold symptom, but simply that Prolog could not find a match in the clauses for the specified goal. In Prolog, False indicates a failure to find a match using the current information.

The use of variables is another source of flexibility in matching. Variables allow a portiion of the pattern to remain unspecified but be referenced for additional testing. Suppose a patient has a runny nose and we wish to get a list of all the diseases in the database for which this is a symptom. Specify the following goal:

```
Goal : Symptom (Disese, runny_nose)
```

This goal includes the variable Disease. Turbo Prolog will respond

```
Disease= cold
Disease= flu
2 Solutions
```

Prolog found two matches for the specified symptoms.

Now let us examine what happens when a particular goal is entered. As illustrated in Fig.2,3 , Turbo Prolog starts at the first clause that matches the specified predicate, which in this case is the first clause in the program. It then tries to match the object argument in the specified goal (runny_nose) with the corresponding argument in the clause predicate. This match will fail. Prolog then tries the next clause, continuing until it finds this match:

Symptom (cold, runny_nose).

The match succeeds, and Prolog displays the values of the variables for the successful match. Execution does not stop, however. Turbo Prolog continues its search until it has tested all predicates for a match with the specified goal. In this case the goal can be solved in two ways by successful matching the variable 'Disease' to 'cold' and 'flu'.

During a match, the variable portion of the condition is <u>bound</u> (replaced by a constant from a database), and that binding is used consistently throughout the fact in both the condition and action part. In this example, Turbo Prolog first binds variable 'Disease' to 'cold' and then to 'flu'.

The solution in the above example illustrates one of the most important principles of Prolog execution: <u>backtracking</u> [Clocksin & Mellish, 1984]. The goal proceeds from left to right. If any condition in the chain fails, Prolog backwards to the previous condition, tries to prove it again with another variable binding, and then moves forward again to see if the failed condition will succeed with the new binding. Prolog moves relentlessly forward through the conditions trying every available binding in an attempt to get the goal to succeed in as many ways as possible.

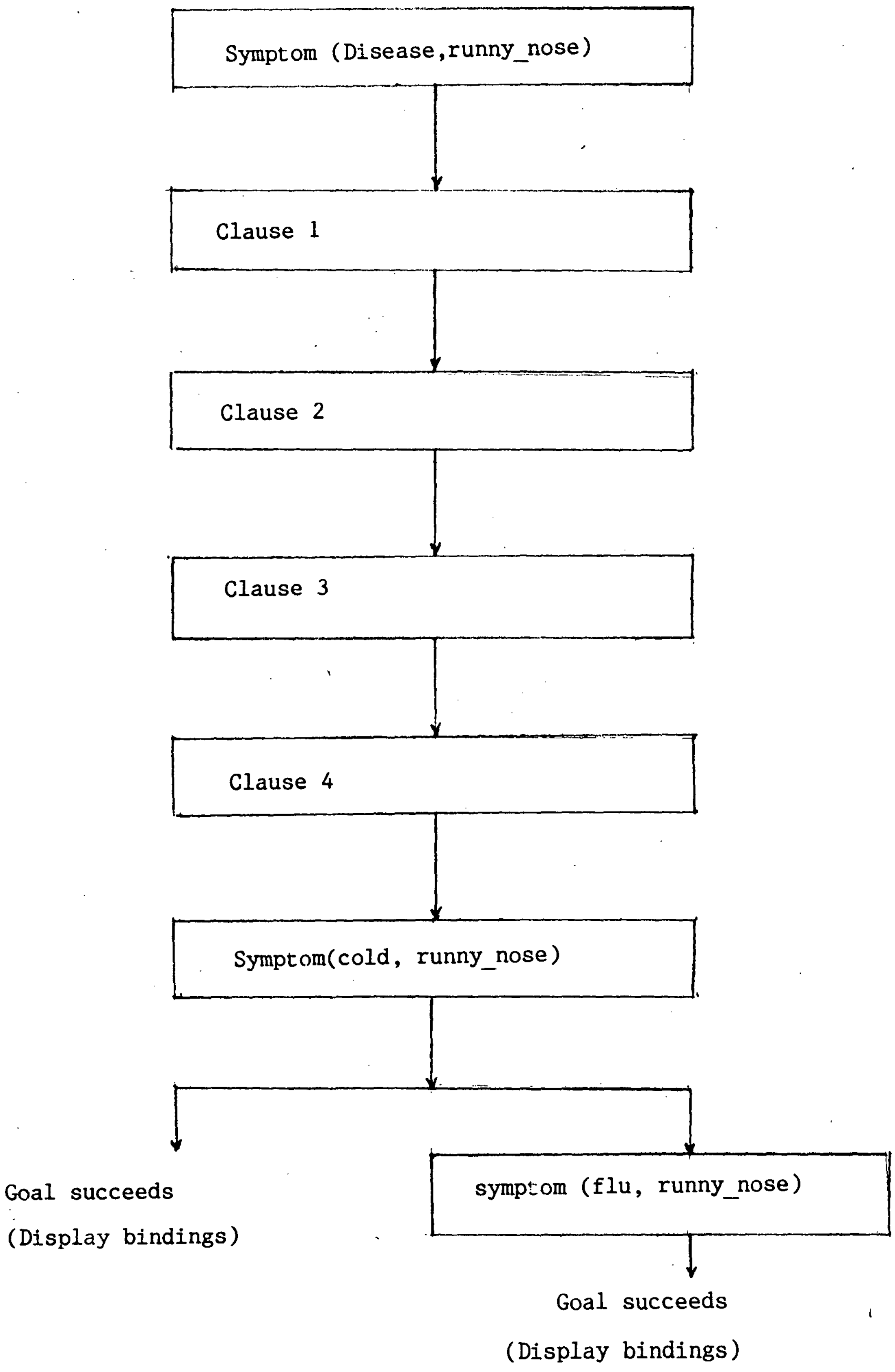

Fig. 2.3. Execution flow of Prolog program

(ii) BACK TRACKING

The solution in the previous example illustrates one of the most important principles of Prolog execution: backtrading [Clocksin & Mellish 1984]. The solution of the compound goal proceeds from left to right. If any condition in the chain fails, Prolog backtracks to the previous condition, tries to prove it again with an order variable binding, and when moves forward again to see it the failed condition will succeed with the new binding. Prolog moves relentlessly forward through the conditions trying away available binding in attempt to get the goal to succeed in many ways as possible.

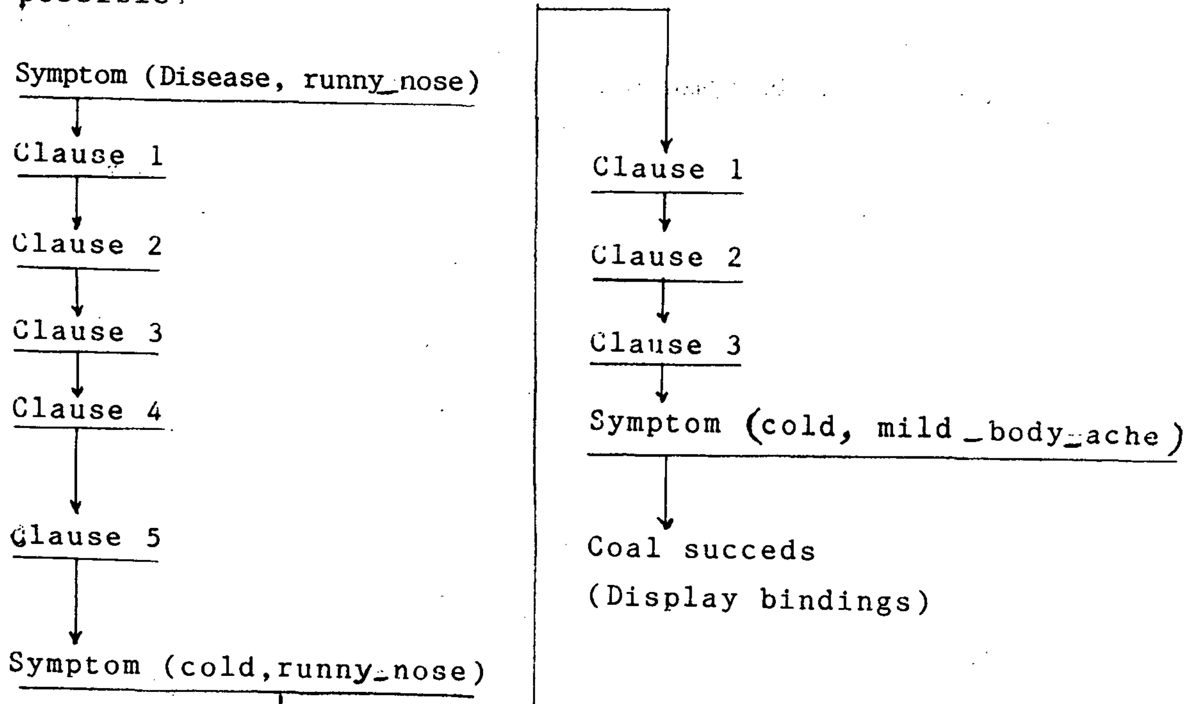

Fig.2.4. Execution with a compound goal

Goal : Symptom (Disese, mild_body_ache)and
Sympton (Disease, runny_nose)

Backtracking is the mechanism that instructs Prolog where to go to look for solution to the program. This process given Prolog the ability to search through all known facts and rules for a solution.

These are the four basic principles of backtracking given:-

* Subgoals must be satisfied in order, from top to bottom.

* Predicate clauses are tested in the order they appear in the program, from top to bottom.

* When a subgoal matches the head of a rule, the body of the rule than constitutes anew set of subgoals to be satisfied.

* A goal has been satisfied when a matching fact is sound for each of the extremities (Leaves) of the goal tree.

Turbo Prolog provides two tools that allow to control the backtracking mechanism, the Fail predicate, which is used to force backtracking and are acut (Signified by !), which is used to prevent backtracking.

iii) Unification:

The process by which Prolog tries to match a term against the facts or the heads of the other rule in an effort to prove a goal is called unification. Unification is a pattern matching process.

Consider the following example-2 program:

```
                    /'* Example-2*/

domains
            disease, indication, name = symbol
predicates
            hypothesis (name, disease )
            hypothesis (name, indication)
clauses
            symptom (charlie, fever).
            symptom (charlie, rash).
            symptom (charlie, headache).
            symptom (charlie, runny_nose).

hypothesis (Patient, measles):-
            symptom (Patient, fever),
            symptom (Patient, cough),
            symptom (Patient, runny_nose),
            symptom (Patient, runny_nose),
            symptom (Patient, rash).
hypothesis (Patient.german_ measles):-
            Symptom (Patient, Headache),
            Symptom (Patient, headache),
Symptom (Patient,runny_nose),
Symptom (Patient, rash).
```

A quick look at or database tells that charlie is sick. Charlie's systems (fever, rash, headache, and runny nose) have been entered as facts in the database. There are also two rules in the database that can be used to make inferences from these known facts. When the program prompts for a goal, enter the folllwing:

```
Goal : hypothesis (Patient, Disease)
```

The program will then display

```
Patient = Charlie, Disease = german_measles
1 solution.
```

Let us look what happened when the goal is specified. Prolog moves immediately to the first rule whose head matches the goal or the first matching facts, whichever it encounters first:

```
hypothesis (Patient, measles)
```

Prolog marks this place and takes the conclusion as a goal and tries to prove it. Moving to the first premise, Prolog then tries to prove

```
Symptom (Patient, fever)
```

Prolog starts from the head of the first rule or fact matching this predicate. This is the first clause in the program:

```
Symptom (Charlie, fever)
```

The match occurs, and patient is bound to charlie. this value

of variable is passed to the original rule.
Prolog then moves to the next premise in the rule, holding the binding to Charlie that has alrady occured:

Symptom (Charlie, cough)

This match will fail, forcing the conclusion hypothesis (Charlie, measles) to fail. Prolog then backtracks and tries to find another match on the original goal. The goal hypothesis (Patient, Disease) will unify next with

hypothesis (Patient, german-measles)

Again the first premise of the conclusion is tested, and Patient is again bound to Charlie. The four premises of the german measles conclusion will all succeed with this binding, causing the original specified goal with hypothesis (Charlie, german-measles) to succeed. The variable 'Patient' and 'Disease' in the original goal are now bound and displayed.

A term is said to unify with another term if

* Both terms appear in Predicates that have the same number of arguments (the same arity), and both terms appear in the same position in their predicates.

* Both terms appear as arguments of the same type-a symbol type can only unify with a symbol type and so on.

* All subterms unify with each other.

  Here are the basic rules for unification:

* Available that is free will unify with any term that satisfies the preceding condition. After unification, the variable is bound to the value of the term.

* A constant can unify with itself or any free variable If the constant is unified with a variable, the variable will be bound to the value ofthe constant.

* A free variable will unify with any other free variable. After unifying the two variables will act as one. If one of the variables becomes bound, the other will be bound to the same value.

Predicates unify with each other if

* They have the same, relation name
* They have the same number of arguments
* All arguments pairs unify with each other.

IV. <u>Execution Control:</u>

Although prolog is not a procedural language, we can provide some control over its excution. Prolog follows these general rules ;

* All clauses for the same Predicate must be grouped together in the program. If we fail to group clauses,

Turbo Prolog will give us an error message. Predicate groups can be entered in any order.

* Within a specific predicate group, Prolog begins testing for a match (unification) at the first fact on rule head that matches the specified goal. Subsequent clauses of the same predicate that also unify are tested in the order in which they appear in the Program.

* If the head of a rule unifies with the goal, the antecedent becomes a new subgoal and must be satisfied from left to right.

Execution is from top to bottom and from left to right. Prolog uses depth-first testing, that is, the testing of any particular rule will Proceed as far as it can until all subgoals fail, and then the next rule is tested to see if it unifies with the specified goal.

The definition of the Predicates and the order of the clauses impose a heuristic on the Program. This heuristic defines the way Prolog works through the problem space to the eventual solution. The order of the clauses controls more than the efficiency of program.

V. Forward and Backward Chaining:

The expert system processes symbolic representations of reality by means of heuristic rules [Lenat, 1982-83], usually by the technique of backward chaining. In backward chaining the engine begins with a terminal conclusion, the object goal. The program searches through the rule list for any rule that has the object goal as its conclusion. The engine then tests each part of the rule's premises and the process begins a new.

Forward chaining systems progress from the given information to a goal. Starting from what is initially known, the current state of the knowledge is used to make a chain of inferences until either a goal is reached a solution is shown to be unattainable, or the search is terminated after exceeding some cut off in use of resources.

2.5 The Two Prolog Data Bases

A prolog program is a collection of facts and rules about a particular knowledge domain. The program really is a database, and Prolog is a very Powerful query language for this database, permitting us to select facts from the database through unification. (a matching algorithm).

The program, however, is a static database, that is, the database does not change over time. When we compile the Program, we essentially 'freeze' the knowledge in the database.

To solve the medical diagnostic system problem, Prolog permits us to add a dynamic data base to our program. Built predicates permit us to add facts to or remove facts from this dynamic database during program execution This is done by creating one or more database predicates and defining these as a data base predicates in a data base section of the program. These database predicates are then updated using one of Turbo Prolog's built in database predicates.

To store information in a dynamic database,we must create one or more database predicates. Facts can then be stored in these predicates during program execution using the build in asserta (fact) and assertz (fact) or by reading facts from a disk file using the built in Consult (filename) predicate

To create a data base we must add a database section to our program and define the predicates in this section. This system must follow the domains section and precede the predicates section.

An expert system, such as the medical diagnostic system, normally use two database predicates. One is for facts proven true, and the other is for facts proven false.

xpositive (symptom)	- Store facts proven true
xnegative (symptom)	- Store facts proven false.

The database predicates can be used like any other predicate. They are considered true if a match is found in the database and falseif no match is found. New facts are stored in database predicates using the asserta (fact)and assertz (fact) built in predicates. Facts are removed from the predicate using retract (fact).For example.

retract (xpositive (fever))
removes the fact fever from the database xpositive(fact).

Variable do not have to be bound before the predicate is invoked. For example,

retract (xPositive (-))

removes the fact fever from the database matching the specified condition.

Here are some general rules for storing facts in a data base.:-

1. The asseta (fact) predicate stores a fact at the beginning of the data base. The assertz (fact) predicate stores a fact at the end of the data base which of these we use depends upon where we wish to put the fact in the database.

2. All variables in fact must be bound before the database predicate (asserta (fact) or assertz (fact) - is invoked) .

3. We can use any expression of a fact as fact in the build in database predicates, and we can use multiple arguments. For example, to store the patient name with the symptom in medical diagnostic system, we could redefine the data base predicate and use

Asserta(xpositive (name, symptom))

in the medical diagnostic system, the dynamic data base must be cleared before the next patient's data is entered. This is done with clear-facts predicate. as follows:

```
Clear_facts:-
retract (xpositive(-)),fail.
Clear_facts:-
retract(xnegative (-)),fail.
Clear_facts.
```

This code includes two failure loops. the first clause.

```
Clear-fact:-
retract (xPositive (-)), fail.
```

Removes one item from the data base and fails at the fail predicate. This forces backtracking to the retract predicate. which removes another item from the database. Eventually the database is empty, and Prolog backtracks to the second clause.

```
Clear-facts:-
retract (xnegative(-)), fail.
```

The second clause loops in the same way, clearing the second database predicate. Like the first clause the retract predicate eventually fails here forcing backtracking to the third clause which always succeeds.

To modify or update a fact in a database, we must delete it and then add it again with our changes incorporated. There is no built-in Predicate for editing, a fact in a data base.

2.6 Probabilistic Reasoning:

probabilitstic reasoning is common in the sciences. Only a small portion of natural science can be termed exact areas such as pure mathematics and sub-field of physics. But most of the world's knowledge is uncertain. As an example consider the problem of diagnosing people's illness from their clinical records. There is certainly some randomness in our description of the world since medical science does not completely understand how the body works. And, in addition, we must design a program that can function even if it does not have access to all the data medical science could conceivably provide, since some clinical tests are expensive and dangerous. Particularly with such incomplete data to work with, we will have to use probabilistic reasoning. Medical expert systems therefore must have a mean of handling the variable degree of confidence in a given factual expression. These expressions of relative confidence are often based on statistics, probability, or subjective opinion.

The majority of Al researchers have however not, hitherto, found standard probabilistic techniques very appealing for use in rule-based expert systems. Among themany alternative numerical schemes for quantifying uncertainty that have been developed are the Certainty Factors (CF) use in MYCIN [Shortliffe & Buchanan, 1975] and its descendants, Fuzzy Set Theory [Zadeh, 1984], the Quasi-probabilistic scheme of PROSPECTOR [Duda et a;. 1976], and the Belief Function of Demster-Shaffer Theory [Shafer, 1976]. There have also been attempts to develop non-numerical schemes, including Paul Cohen's Theory of Endorsements [Cohen,1 985], Doyle's Theory of Reasoned Assumptions [Doyle, 1983] etc.

Conditional probability provides useful reslts in the areas of medical decision making. The medical diagnostic problem can be viewed as the assignment of probabilities to specific diagnoses after analyzing all relevant data. If the sum of the relevant data (or evidencea) is represented by e, and d_i is the ith diagnosis (or "disease") under consideration, then $P(d_1/e)$ is the conditional probability that the patient has disese i in the light of evidence e. Diagnostic programmes have traditionally sought to find a set of evidence that allows $P(d_i/e)$ to exceed some threshold, say 0.95, for one of the possible diagnoses Under these conditions the second-ranked diagnosis is sufficiently less likely (-0.05) that the user is content to accpet disease i as the diagnosis requiring therapeutic attention.

Bayes' Theorem is useful in these applications because it allows $P(d_i/e)$ to be calculated from the component conditional nprobabilities:

$$P(d_i/e) = \frac{P(d_i)\ P(e/d_i)}{\sum P(d_j)\ P(e/d_j)}$$

In this representation of theorem, d_i is one of the n disjoint diagnoses, $P(d_i)$ is simply the priori probability that the patient has diesese i before any evidence has been gathered , and $P(e/d_i)$ is the probability that a patient will have the coplex symptoms and signs represented by e, given that he or she has disease d_i.

Diagnostic programs that mimic the process of analyzing evidences incrementally often use a modified version of Bayes' Theorem which can be stated as follows:

Let e_1 be the set of all observations to date, and s_1 be some new piece of data. Further let e be the new set of observations once s_1 has been added to e_1. Then :

$$P(d_i/e) = \frac{P(s_1/d_1/d_i\&e_1)\ P(d_i/e_1)}{\sum P(s_1/d_j\ \&e_1)\ P(d_j/e_1)}$$

The successful program that use Bayes' Theorem in this form requires huge amounts of statistical data, not only $P(s_k/d_j)$ for each of the pieces of data, s_k, in e, but also the interrelationships of the s_k within each disease d_j.

The conditional probability statements are generally referred as decision rules or decision criteria in the diagnostic context. For example, the expression $P(d_i/s_k) = x$ can be read as a statement that there is a 100x% chance that a patient observed to have symptom s_k has disease d_i. Stated in rule form, it would be

IF: The patient has sign or symptom s_k

THEN: Conclude that he has disease d_i with probability x

The value of x for such rules may not be obvious, but an expert may be able to offer an estimate of this number based in clinical experience and general knowledge[Shortliffe * Buchanan, 1975].

The estimation of prior probabilities for many of the propositions being dealt with in tedious. Also, it is difficult to asisgn the prior probabilities in advance to some particular propositions for a certain application area. Also, it seems impossible at the moment to guarantee that all the prior probabilities could be chosen consistently, especially for a large knowledge-base. But it has been found [Cheeseman, 1985; Quinlan, 1983] that supplying the prior probabilities can lead to more precise results, if it can indeed be known in advance, for example, in application areas such as poker.

Several problems can arise when a single probability value represents the state of belief for an event. As pointed out in [Quinlan, 1983], the first problem is that nothing can be known from the single value about its precision. The second is that the single value combines the evidence for and against a proposition withoutindicating how much there is of each. it is easy to see that the belief in a particular proposition is, often, not necessarily the complement of the disbelief in the same proposition. The third is the difficulty in dealing with inconsistent information as put in [Quinlan, 1983].

It is only in systems using a two-valued approach where the values are probabilities that there is a firm basis for detecting general inconsistency.

The does not mean that the systems using the single-value approaches can not detect inconsistency. It is just more difficult, more computationally expensive perhaps in systems employing single-value approaches, say Blake's system, than in those using two-valued approaches [Liu & Gammerman, 1987].

Quantification of the real world knowledge has been the subjet matter of criticism for the researchers working in the field of expert systems. In particular, White [1984] describes the principal deficiencies of the earlier systems like PROSPECTOR. Quinlan's INFERNO [1983] is a recent probabilistic inference system that solves some of the problems of earlier systems. INFERNO system can deals with unquantified (logical) propositions and relations among them. However an improvement to deal with quantified probabilistic conclusion in INFERNO system has been made by Liu and Gammerman [1987].

CHAPTER III

SYSTEM IMPLEMENTATION

3.1. Introduction:

As it has been discussed in the previous chapters that early attempts in the field of expert systems were made on generalized problem solving, but it was found that these systems did not offer sufficient power for high performance. Then there was a move towards task-specific problem solving, called knowledge-based systems [Davis and Lenat, 1980]. Rather than non-specific problem solving power, knowledge-based systems have emphasised both the accumulation of large amount of knowledge in a single domain, and the development of domain specific techniques in order to develop a high level of expertise. But the development of such large expert system demands a team of people trained in knowledge engineering and specific field of application. Moreover, this type of system requires huge amount of memory and can be implemented only on mainframe computers. From commercial point of view, these systems are not cost effective for small organization and individual.

Keeping in view the above facts small expert systems have gained wide popularity among the researchers of this discipline in recent years. Small expert systems are designed to run on personal computers. They are designed to help individuals deal with small but nasty problems. In some cases, these problem require knowledge engineering techniques because they are not easily amenable to traditional solutions. In other cases, the problems are simply ones that users understand and want to solve. Users maynot understand the traditional programming techniques that would be required to solve these problems, but they will be able to turn to knowledge-based approaches simply because of the user-friendly interface and the rapid prototyping features that are available.

Probably the most promising strategy for an individual exploring applications is to purchase and experiment with one of the small knowledge based system tools that are now in the market place. There are many reasonably small problems for which knowledge based systems offer cost-effective solutions. And in the end there is no substitute for the experience gained from building a prototype system. With a small investment in software and training and a properly focused project, onecan develop a useful prototype. This approach will allow one to investigate the field and develop interest within his company without making a major commitment [Harmon & King, 1985].

Our expert system is a small expert system developed for the diagnosis of all types of fevers known on the earth at present keeping in view all the advantages discussed above. This system incorporates the feature of modularity, easily implemented on any IBM PC or compatible computers (with a memory of 640K RAM) running under MS-DOS or PC-DOS. The program has been written in Turbo Prolog, and therefore the system is very strong in reasoning but lacks the facility of explanation, knowledge acquisition, and natural language interfacing. The knowledge is expressed in form of production rules. For inference mechanism backward chaining, depth-first scanning, and two level of certainties have been exploited.

3.2. Problem Formulation;

Let d_i ($i = 1, 2,$ ------------) be a universal set of all diseases and f_i will be a subset of d_i.

Further let S_i^* be sets of symptoms of all patients of each disease and assume that S_i is a subset of S_i^*

which comprises all elements of f_i that are already known. Also assume that a patient's symptoms are r. Diagnosis can be said to be an inference of S_i^*, judging from the information of S_i. Expert doctors acquire knowledge necessary to make this inference from S_i, which are sets of known symptoms of the past d_i.

We can now expresss our problem in rule form as :

If S_1, S_2, ---------------THEN f_i

Where S_1, S_2, ------------- are elements of set S_i and f_i is the ith element of f_i.

The author created the following two categories of knowledge to use as inferences:

A. Provisional-Rule (P-Rule):

This rule is based on the ways of clarifying patients' conditions through clinical settings. The main emphasis here is on physical examinations because more information about patient is hard to obtain immediately. According to this type of rule, if all symptoms listed in the premise part of hypothesis are fulfilled, the patient is most probably suffering from the disease indicated. For a particular case of 'Typhoid Fever' this rule is illustrated with the help of 'Example-3'.

B. Definite-Rule (D-Rule):

This rule is based on clinical settings as well as on laboratory tests and X-ray reports etc. According to this type of rule, if all symptoms listed in the premise part of the P-Rule Hypothesis are fulfilled, The system asks some more questions regarding laboratory tests and x-ray reports etc. If all these symptoms are satisfied, then the system concludes that the patient is definitely suffering from the disease indicated. For a special case of 'Typhoid Fever' this rule is illustrated with the help of 'Example-4'.

From the careful observations of the examples for P-Rule and D-Rule, it is clear that the English descrition to reach at the conclusions is the same except on more symptom of blood test. However, in Prolog the conclusion in D-Rule is made with extra rule added to the P-Rule.

3.3. The Expert System Architecture:

Consultative advice is often required when the attending physician may not be an expert. The main purpose of the system is to provide advice and it does so via the consultation system. The principal components of this system are shown in Fig. 3.1.

/* Example-3 */

Engish	Prolog
IF There is a sudden onset	hypothesis ("Typhoid Fever"):-
of fever	symptom (onset_s),
AND there is a malaise	symptom (malaise),
AND there is a headache	symptom (headache),
AND there is a drowsiness	symptom (drowsiness),
AND there is a continued	symptom (aching_1),
pain in limbs	symptom (vomiting),
	symptom (cough),
AND There is a cough	symptom (epistaxis),
AND There is a bleeding	symptom (constipation),
from the nose	symptom (tachycardia),
AND there is constipation	symptom (rash_c1)
AND there is a abnormal	symptom (rash_ p_2)
neart action	p_diagnosis ("Typhoid
	Fever").
AND there is a rose-red spots	
AND there is a rash on the upper abdomen and on the back	

THEN the <u>most probable</u> disease is : Typhoid Fever.

/* Example-4 */

English

IF there is a sudden onset
   of fever
AND there is a malaise
AND there is a headache
AND there is a drowsiness
AND there is a continued
    pain in limbs
AND there is vomiting
AND there is a cough
AND there is a bleeding
    from the nose
AND there is a constipation
AND there is an abnormal
    heart action
AND there is a rose-red spots
AND there is a rash on the
    upper abdomen **and** on
    the back
AND there is a positive
    blood culture for s-typhi
    organism
THEN the disease is <u>definitely</u>: Typhoid Fever.

Prolog

```
hypothesis ("Typhoid Fever"):-
symptom (onset_s),
symptom (malaise),
symptom (headache),
symptom (drowsiness),
symptom (aching_l),
symptom (vomiting),
symptom (cough),
symptom (epistaxis),
symptom (constipation),
symptom (tachycardia),
symptom (rash_cl),
sympton (rash_p2).
p_diagnosis        ("Typhoid
                   Fever").

p_diagnosis ("Typhoid Fever"):-
symptom (blood_tl),

diagnosis ("Typoid Fever").
```

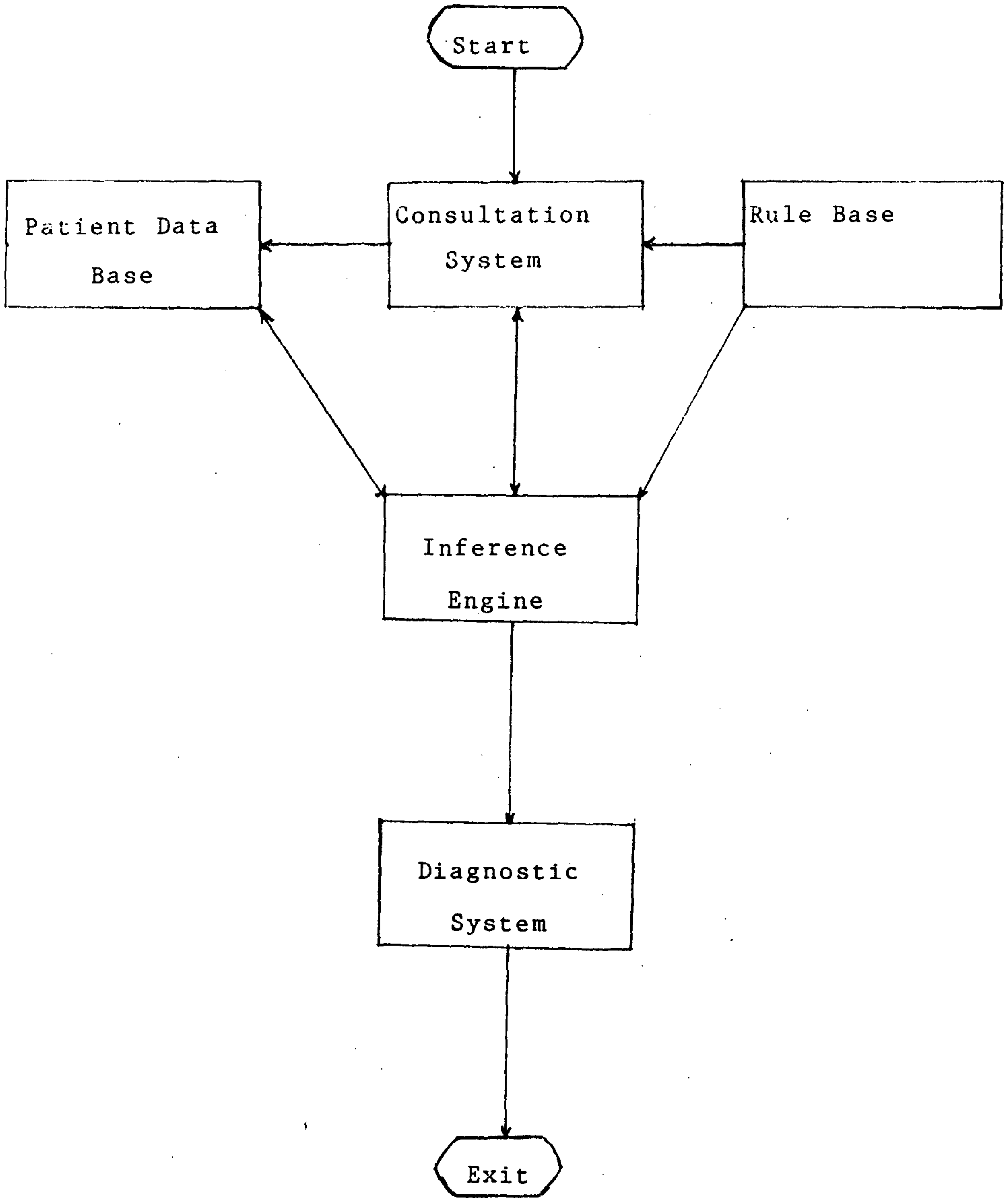

Fig. 3.1 Components of our expert system:

The consultation system asks questions about the patient under consideration. A great deal of attention has been paid to making the interaction easy and natural. Questions have been carefully worded so as to sert up the comprehensive description. Patient's information can be entered by answering each question in turn simply in hes or no (y/n) form. An illustration of the dialogue session is given below in 'Example-5'.

/* Example-5 * /

Does the patient have a sudden onset of fever (y/n)?

Does the patient have a sore throat(y/n)?

Does the patient have headache (y/n) ?

Does the patient have continued pain in limbs (y/n) ?

Does the patient have vomiting (y/n) ?

Does the patient have cough (y/n)?

Does the patient have bleeding from nose (y/n)?

Does the patient have rose-red spots (y/n)?

Does the patient have constipation (y/n)?

Does the patient have palpable and soft spleen (y/n)?

Is the blood culture positive for s-typhi organism (y/n)?

After acquiring information on the questions, The system applies rules for the diagnosis of particular type of fever. If more information is required by the system, additional questions will appear on the screen.

The rule-base consists of all the informations expressed in rule forms. According to the information received from the consultation system during the dialog session, the decision rules are used by the system to draw inferences. The rule base of the system is comprised of two types of knowledge as described earlier in this chapter (section 3.2).

The patient data base stores the data about a specific patient collected during a consultation. This data is a temporary storage of deductions and conclusions about the patient recently attended.

The inference engine performs unification, determines the order in which to scan the rules, and performs conflict resolution. The system is implemented in Prolog and therefore uses Prolog theorem proving and backtracking for its inference. Prolog has really its own inference engine. This feature makes designing an expert system relatively easy because one needs only to write the knowledge base. This feature also means, however, that the user has less control. For example, Prolog works backward. Prolog is also limited to depth-first scanning. All rules relative to a particular goal are scanned as deeply as possible for a solution before Prolog backtracks and tries an alternative goal.

The diagnosis system displays the conclusions about a particular type of fever for the patient under consisderation. Not only the diagnosis but the treatment with additional advice of prevention, precaution, and comments is also displayed whenever required.
This system is basically incorporated into the knowledge base with the help of built-in predicate feature of Turbo Prolog. Conclusions appear in two forms depending upon either the P-Rule or D-Rule is satisfied. Suppose P-Rule is satisfied for 'Typhoid Fever', then the output will be in the form as shown in 'Example-6'.

/* Example-6 */

```
-------------------PROVISIONAL     DIAGNOSIS------------
       The most probable disease is: Typhoid Fever
---------------------ADVICE--------------------------
       I am very sorry that I could not make a THERAPY
RECOMMENDATION for you. Please contact your doctor for
further medical conclusion.
       Thank you for your visit
                                                Good bye
```

Further suppose that D-Rule is satisfied for "Typhoid Fever". then the output will be in the form as shown in 'Examle-7'.

```
                    /* Example-7 */
----------------------DIAGNOSIS----------------------
Disease      : Typhoid Fever
Aetiology    :  Bacterial Infection
----------------------TREATMENT----------------------
General      :  Isolation, rest in bed, special attention
                to nutrition, and fluid intake.
Medicine     :  Chloramphenicol
Dose         : 3g daily  for 14 days
Cement       :   Dose should be reduced to 2g daily as
                patient resposnes.
----------- ----------PREVENTION---------------------


       Those who wish to travel or live in countries
where infections are endemic should be inoculated   with
vaccine   containing killed s-typhi and s-paratyphi A
and B (TAB).


       Thank you for your visit
                                                Good bye
```

3.4. Diseases Diagnosed by The Expert system

Our Expert System is aimed at patients whose chief complaint is fever. The knowledge for the classification of fevers used by the system has been extracted from the book entitled "Davidson's Principles and Practice of Medicine, John Macleod (1977)" and checked by expert doctor at 'All India Institute of Medical Sciences (AIIMS)', New Delhi. The diseases in order of their occuring are listed below:

1. Scarlet Fever
2. Typhoid Fever
3. Paratyphoid Fever
4. Abortus Fever
5. Rheumatic Fever
6. Cerebrospinal Fever
7. Malaria
8. Black-water Fever
9. Query Fever
10. Scrup Typhus Fever
11. Rat-bite Fever
12. Sandfly Fever
13. Pharyngoconjunctival Fever
14. Glandular Fever
15. Haemorrhagic Fever
16. Epidemic Typhus Fever
17. Endemic Typhus Fever
18. Rocky Mountain Spotted Fever
19. Lassa Fever
20. Oroya Fever
21. Louse-borne Relaps- Fever
22. Tick-borne Relapsing Fever
23. Yellow Fever
24. Trench Fever

3.5. Working Environment:

The medical knowledge used in our expert system has been coded into software using the language 'Turbo Prolog'. This knowledge has been represented in a predicate-logic manner of Prolog. The pattern matching feature of Prolog is a very efficient and useful function. which makes the retrieval and evaluation of information quick and convenient (Kimura et al., 1985].

The expert system is operative on any IBM PC or compatible computers using MS-DOS operating system. Because it is operative on a micro computer,the system is portable and readily available. This system can also be used in modular package. That is, small expert systems developed separately can be combined with this expert system in order to build a large expert system. This can also be used to handle diseases other than fevers. This task can be achieved simply by adding new rules using the same original program. The rules have been expressed in a very manner that can be followed by anyone interested to use it for other diseases.

CHAPTER-IV

DISCUSSION AND CONCLUSION

4.1. Discussion:

Our expert system has been designed with the aim to diagnose almost all types of fever. The most important feature of this system is that it not only diagnoses the diseases but also prescribes the treatment with additional advices of prevention, precaution, and comments.

The knowledge expressed in the rule forms has been derived from the book entitled "Davidson's Principles and Practice of medicine, edited by John Macleod (1977)". Moreover, the symptoms, causes, and treatment measures regarding each and every type of fever have been checked by expert doctor at 'All India Institute of Medical Sciences(AIIMS), New Delhi'. This medical knowledge has been coded into the rule form using the logic-based language 'Prolog'. Logic is ideally suited for this kind of knowledge-based system. It allows high level of specifications of the rules and when expressed in Prolog provided execution of the specification. The translation of the English description of the rules into logic is easy and in particular the separation of knowledge in the forms of rules from control gave clear semantics to the system. The language 'Turbo Prolog' used in this system has the following features [Hankley, 1987] :

i) Prolog being its own inference engine provides a great facility to the system designers because they have only to write the knowledge base.

ii) Stand-alone programs can be complied to be executed on a machine that is not running 'Turbo Prolog'. These stand-alone programs can be sold or distributed to users without paying any royalty to Borland International.

iii) A full complement of standard predicates for many functiions such a string functions, random file access, cursor control, graphics, winnowing, and sound are available.

iv) A functional interface to other languages is provided allowing procedural language support to be added to any Prolog System.

v) Declared variables are used to provide more secure development control.

vi) Both integer and real (floating-point) arithmatic operations are allowed.

vii) An integrated editor is provided, making program development, compilation, and debugging very easy.

Inspite of the above features of Turbo Prolog, there are certain limitations in the use of this language. Some of the most important are as follows:

i) Turbo Prolog does not support virtual memory. The size of the systems developed with Turbo Prolog, however, is limited by the amount of memory available.

ii) Turbo Prolog is inefficient for numeral processing.

iii) Turbo Prolog in some ways reflects the structural aspects of the procedural languages. The variable declaratiion requirement and the division of the program into sections impose some limitations on Turbo Prolog symbolic processing that do not exist for more traditional Prolog languages.

iv) Expressive power of the Horn Clause is limited and therefore explanation facility is not provided in our expert system.

As our expert system has been designed with the help of production rules, therefore it offers the following advantages [Shaefer, P.R.m 1985]:

i) New rules can be added for novel situations without debalancing te rest of the system.

ii) An effective control strategy can handle complex domains using only simple rules.

iii) The small chunks of knowledge can be incrementally assembled to augment the behaviour of the systems.

Our Expert System can also be used to handle diseses other than fevers. This can be done simply by adding new rules using the same original program. The description of the rules in this system is very clear and precise that can be adapted by anyone who is interested to use it for other diseases.

Despite the above features, the system will serve as a tool in the hands of non-expert physicians specially working in villages and small towns. The system may be very helpful for the poor people who can not afford the high cost of modern medical treatment.

The mortality rate in India due to poverty and negligence of medical attention may also be reduced considerably. It is therefore suggested that this type of systems should be used in every Government Hospital specially in villages and small towns.

Although field testing to evaluate the performance of our Expert System has yet to be done, the investigator is optimistic for the success of this expert system in medical world. This system has provided good results when tested with hypothetical symptoms of a particular type of fever in the laboratories.

It is worthwhile to point out that our Expert system is lacking the facilities of explanation, natural language interfacing, and knowledge acquisition. The shortcoming of explanation facility is only because of the very nature of the language 'Prolog' itself. In general, three capabilities, namely, action, description,and reasoning are needed for an expert system language. Historically, the language strong in one of these capabilities tended to be relatively weak in others. Prolog is a reasoning-oriented language, that is, limited by its inefficiency of description and action.

The explanation facility can be included in this system but it will create complexities. It is desirable, although not yet common, to have user-friendly natural language interface to facilitate the use of the system. A natural language interface simulates causal conversation, using everyday expression in gramatically correct sentences. Obviously, the more natural the interface, the greater the demands on permanent storage and memory. Hence, system with extraordinary user sensitivity are largely restricted to mainframe resources accessible from remote work station. Building tools designed to operate within the constraints of PC environment necessarily sacrifice "friendliness" for the sake of efficiency. As knowledge acquisition system is a learning module of an expert system, very few existing expert systems have such a module.

Uncertainty is not a major feature of this system and the qualitative method of the expert is used. Conclusions may be qualified either by <u>most</u> <u>probably</u> or by <u>definitely.</u> The rules used in our system are based on these two qualitative description of <u>probabilistic reasoning.</u> The first rule involves <u>physical examinations</u> of the patient while the second rule, in addition to the first, involves laboratory tests and X-ray reports etc.

4.2. CONCLUSION:

Our expert system is a small expert system developed to diagnose almost all types of fever existing in the world at present. In addition, it not only diagnoses the diseases but also prescribes the treatment with extra informations of prevention, precaution, and comments whenever required. Our expert system is implemented by a programming language 'Turbo Prolog'. All the knowledge used in this system is represented in a predicate-logic manner of Prolog. This system is operative under MS-DOS or PC-DOS ON ANY IBM PC or compatible computer (with a memory of 640k RAM). Becuase of the modular characteristic of this system, the other facilities like knowledge acquisition, language processor, and explanation can be added in near future.

The default code array for this system to run is 16k bytes, whereas the size of the whole program is more than the default value. Therefore the investigator added a code directive (code = 2048) in the beginning of the program. This code directive increases the space available for the array; otherwise system can not be run.

The field testing has yet to be done to evaluate the performance of our system but this system has provided satisfactory results when tested with hypothetical symptoms in the laboratories.

A - APPENDIX - PROGRAM

```
/* ----------------------------------------*/
/* aziza aqel                  15/11/89     */
/* alaa aqel                                */
/* medical diagnostic system for fevers     */
/* in turbo prolog            version 2.0   */
/* ----------------------------------------*/
   code = 2048

   domains
      disease , query = string
      symptom         = symbol
      reply           = char

    database
      xpositive(symptom)
      xnegative(symptom)

    predicates
      go
      title
      positive(query,symptom)
      negative(query,symptom)
      symptom(symptom)
      clear_facts
      remember(symptom,reply)
      ask(query,symptom,reply)
      go_once
      hypothesis(disease)
      p_diagnosis(disease)
      diagnosis(disease)

    clauses
      go:-
        go_once,
        write("Woud you like another consultation (y/n) ? "),
        readchar(Reply),nl,

        Reply = 'y',
        go.

      go_once:-
        title,nl,
        hypothesis(_),!,
        clear_facts.
      go_once:-
        write("Sorry,I don't seem to be "),nl,
        write("Able to diagnose the disease."),nl,
        clear_facts.

      title:-
      write("medical diagnostic system"),nl,
      write("For fevers"),nl.

   positive(_,Symptom):-
      xpositive(Symptom), ! .

   positive(Query,Symptom):-
      not(xnegative(Symptom)),
      ask(Query,Symptom,Reply),
      Reply = 'y'.

   negative(_,Symptom):-
      xnegative(Symptom), !.

   negative(Query,Symptom):-
      not(xpositive(Symptom)),
      ask(Query,Symptom,Reply),
      Reply = 'n'.

   ask(Query,Symptom,Reply):-
      write(Query),
      readchar(Reply),
      write(Reply),nl,
      remember(Symptom,Reply).

   remember(Symptom,'y'):-
      asserta(xpositive(Symptom)).

   remember(Symptom,'n'):-
      asserta(xnegative(Symptom)).

   clear_facts:-
      retract(xpositive(_)),fail.

   clear_facts:-
      retract(xnegative(_)),fail.
   clear_facts.

   symptom(Symptom):-
      xpositive(Symptom), !.

   symptom(Symptom):-
      xnegative(Symptom), !,fail.

symptom(onset_s):-
   positive("Does the patient have asudden onset of fever (y/n)?", onset_s)
symptom(sore_throat):-
   positive("Does the patient have asore throat (y/n)? ", sore_throat).
symptom(shivering):-
   positive("Does the patient have shivering (y/n)?",shivering).
```

```
symptom(pyrexia):-
 positive("Does the patient have temperature above normal (y/n)?",pyrexia
symptom(headache):-
  positive("Does the patient have headache (y/n)?",headache).
symptom(vomiting):-
  positive("Does the patient vomiting (y/n)?",vomiting).
symptom(lymph_nodes_t):-
  write("Does the patient have tonsillak lymph nodes enlaged and "),nl,
  positive("tender (y/n)?",lymph_nodes_t).
symptom(rash_p1):-
  write("Does the patient have rash first appearing behind the ear,"),nl,
  positive("then it becomes reddish spot (y/n)?",rash_p1).
symptom(rash_i):-
 positive("Is the rash intense in flexure of arms and legs (y/n)?",rash_i)
symptom(throat_swab_1):-
  write("Is the throat_swab culture strongly positive or beta"),nl,
  positive("hamolytic streptococci (y/n)?",throat_swab_1).
symptom(aso):-
  positive("Is there is rising aso titre in serum (y/n)?",aso).
symptom(onset_g):-
  positive("Does the patient have gradual onset of fever (y/n)?",onset_g).
symptom(malaise):-
  positive("Does the patient feel some malaise (y/n)?",malaise).
symptom(drawsiness):-
  positive("Does the patient have drawsiness (y/n)?",drawsiness).
symptom(aching_l):-
  positive("Does the patient have continued pain in limb (y/n)?",aching_l)
symptom(cough):-
  positive("Does the patient have cough (y/n)?",cough).
symptom(epistaxis):-
  positive("Does the patient have bleeding from nose (y/n)?",epistaxis).
symptom(constipation):-
  positive("Does the patient have constipation (y/n)?",constipation).
symptom(tachycardia):-
  positive("Does patient have abnormal heart action (y/n)?",tachycardia).
symptom(rash_p2):-
  write("Does the patient have rash on upper abdomen and"),nl,
  positive("on the back (y/n)?",rash_p2).
symptom(rash_c1):-
  positive("Does the patient have rose_red spots (y/n)?",rash_c1).
symptom(spleen_ps):-
  positive("Is the patient spleen palpable and soft (y/n)?",spleen_ps).
symptom(blood_t1):-
  write("Is the blood culture positive for s_typhi"),nl,
  positive("organism (y/n)?",blood_t1).
symptom(blood_t2):-
  write("Is the blood culture positive for s_paratyphi"),nl,
  positive("organism (y/n)?",blood_t2).
symptom(sweating):-
  positive("Does the patient have sweating (y/n)?",sweating).
symptom(weakness):-
  positive("Does the patient have weakness (y/n)?",weakness).
symptom(anorexia):-
  positive("Does the patient have loss of appetite (y/n)?",anorexia).
symptom(pain_lb):-
  positive("Does the patient have pain in limbs and back (y/n)?",pain_lb).
symptom(pain_j):-
  positive("Does the patient have joint pain (y/n)?",pain_j).
symptom(spleen_p):-
  positive("Is the spleen of the patient palpable (y/n)?",spleen_p).
symptom(arthritis):-
  write("Does the patient have inflammation in one or more"),nl,
  positive("joints (y/n)?",arthristis).
symptom(radiology):-
  write("Is there demonstration of radiological change"),nl,
  positive("akin to bone inflammation (y/n)?",radiology).
symptom(agg_cf):-
  write("Is the blood of the patient positive for agglatination"),nl,
  positive("and complement_fixation test (y/n)?",agg_cf).
symptom(fatigue):-
  positive("Does the patient feel tirednss (y/n)?",fatigue).
symptom(weight):-
  positive("Does the patient have loss of weight )y/n)?",weight).
symptom(joint_1):-
  write("Are the large joints of the patient principally affected"),nl,
  positive("e.g knees,ankles,sholders and wrists (y/n)?",joint_1).
symptom(polyarth):-
  write("Does the patient have inflammation in several"),nl,
  positive("joints (y/n)?",polyarth).
symptom(tongue_f):-
  write("Is there dust like deposition on patient's"),nl,
  positive("tongue (y/n)?",tongue_f).
symptom(proteinuria):-
  positive("Is the urine test positive for protein (y/n)?",proteinuria).
symptom(erythema_c):-
  positive("Does the patient have pink rash on trunk (y/n)?",erythema_c).
symptom(x_ray):-
  write("Does the chest x_ray show enlargement of cardiac"),nl,
  positive("shadow (y/n)?",x_ray).
symptom(throat_swab_2):-
  write("Is the throat _swab culture positive for"),nl,
  positive("group A streptococci (y/n)?",throat_swab_2).
symptom(rash_p3):-
  write("Does the patient have few lesions on buttocks,then"),nl,
  positive("involving limbs and trunk (y/n)?",rash_p3).
symptom(blood_t3):-
  write("Is the blood culture positive for"),nl,
  positive("N.meningitidis (y/n)?",blood_t3).
symptom(diplococci):-
  positive("Does the CSF show gram positive diplococci (y/n)?",diplococci).
```

```
symptom(teeth):-
  positive("Is the patient teeth chattering (y/n)?",teeth).
symptom(spleen_pt):-
  positive("Is the spleen palpable and tender (y/n)?",spleen_pt).
symptom(locality):-
  positive("Does the patient belong to malarial locality (y/n)?",locality).
symptom(fever_i):-
  positive("Does the patient have irregular fever (y/n)?",fever_i).
symptom(blood_film1):-
  write("Is the blood smear positive for malarial"),nl,
  positive("parasite (y/n)?",blood_film1).
symptom(spleen_et):-
  positive("Is the patient spleen enlarged and tender (y/n)?",spleen_et).
symptom(liver_et):-
  positive("Does the patient liver enlarger and tender (y/n)?",liver_et).
symptom(urine_c):-
  write("Does the colour of urine vary from dark red to"),nl,
  positive("black (y/n)?",urine_c).
symptom(jaundice):-
  positive("Does the patient have jaundice (y/n)?",jaundice).
symptom(chest):-
  positive("Does the patient have pain in chest (y/n)?",chest).
symptom(rash_s):-
  positive("Does the patient have sparse rash (y/n)?",rash_s).
symptom(diarrhoea):-
  positive("Does the patient have diarrhoea (y/n)?",diarrhoea).
symptom(blood_t4):-
  write("Is the blood culture positive for coxiolla"),nl,
  positive("burneti organism (y/n)?",blood_t4).
symptom(rash_c2):-
  write("Does the patient have first small reddish flat then"),nl,
  positive("large raised rash (y/n)?",rash_c2).
symptom(conjuctival):-
  positive("Does patient have conjunctival infection (y/n)?",conjuctival).
symptom(lymph_nodes):-
  positive("Does  patient have enlarged lymph nodes (y/n)?",lymph_nodes).
symptom(rash_p4):-
  write("Does the patient have rash on face,neck,arms,palms,"),nl,
  positive("legs and soles (y/n)?",rash_p4).
symptom(blood_t5):-
  write("Is the blood culture positive for"),nl,
  positive("R.tsutsugamushi organism (y/n)?",blood_t5).
symptom(wound):-
  write("Is the patient's wound inflammed,indurated,purplish and"),nl,
  positive("painful (y/n)?",wound).
symptom(rash_c3):-
  write("Does the patient have macular or maculopopuler dusky red"),nl,
  positive("rash (y/n)?",rash_c3).
symptom(sparse):-
  positive("Is the rash sparse over trunk and extremeties (y/n)?",sparse).
```

```
symptom(s_minus):-
  write("Is the spirillum minus demonstrable for inflammed"),nl,
  positive("bite under dark ground illumination (y/n)?",s_minus).
symptom(blood_t6):-
  write("Is blood culture positive for spirillius"),nl,
  positive("minus organism (y/n)?",blood_t6).
symptom(consti):-
  positive("Does the patient have constitutional symptom (y/n)?",consti).
symptom(aches):-
  write("Does the patient have continued pain in"),nl,
  positive("orbital areas (y/n)?",aches).
symptom(eyes):-
  positive("Does the patient have painful movemet of eyes (y/n)?",eyes).
symptom(photophobia):-
  positive("Does patient have photophobia (y/n)?",photophobia).
symptom(nausea):-
  positive("Does the patient have inclination to vomit (y/n)?",nausea).
symptom(lymph_nodes_s):-
  positive("Is the superficial lymph nodes enlarged (y/n)?",lymph_nodes_s).
symptom(virus_1):-
  positive("Is the blood culture positive for arbovirus (y/n)?",virus_1).
symptom(insomnia):-
  positive("Does the patient have problem in sleeping (y/n)?",insomnia).
symptom(depression):-
  positive("Does the patient have depression (y/n)?",depression).
symptom(conjunctivitis):-
  positive("Does the patient have conjunctivitis (y/n)?",conjunctivitis).
symptom(inflammed):-
  write("Does the patient have inflammed pharynx,tonsils and"),nl,
  positive("adenoids (y/n)?",inflammed).
symptom(throat_swab_3):-
  write("Is the throat swab culture positive for"),nl,
  positive("strep,pyogenes organism (y/n)?",throat_swab_3).
symptom(pain_m):-
  positive("Does the patient have pain in muscles (y/n)?",pain_m).
symptom(rash_m):-
  positive("Does  patient have small flat to raised rash (y/n)?",rash_m).
symptom(hepatitis):-
  positive("Does  patient have inflammation of livers (y/n)/",hepatitis).
symptom(spleen_es):-
  positive("Is the spleen enlarged and soft (y/n)?",spleen_es).
symptom(heterophile):-
  positive("Does  blood show heterophile antibodies (y/n)?",heterophile).
symptom(pb_reaction):-
  write("Is the paul_bunnel reaction positive in titres of"),nl,
  positive("1 in 200 diluation (y/n)?",pb_reaction).
symptom(skin):-
  positive("Does the patient have bleedig spot in skin (y/n)?",skin).
symptom(urine_cell):-
  write("Does the patient's urine contain red cells and alarge"),nl,
  positive("amount of protein (y/n)?",urine_cell).
symptom(leucocyte):-
  write("Is the white blood cell count low initially and"),nl,
  positive("icrease further (y/n)?",leucocyte).
symptom(blood_c):-
  positive("Does the colour of blood change successively (y/n)?",blood_c).
symptom(frontal):-
  positive("Does the patient have frontal headache (y/n)?",frontal).
symptom(face):-
  positive("Is the patient face red and bluish (y/n)?",face).
symptom(congested):-
  positive("Does the patient have red eyes (y/n)?",congested).
symptom(dullness):-
  positive("Does the patient have dullness (y/n)?",dullness).
symptom(rash_p5):-
  write("Does the rash appear on anterior folds of armpits,sides of"),nl,
  positive("abdomen,thence on trunk and forearms (y/n)?",rash_p5).
symptom(petechial):-
  positive("Does the patient have minute bleeding spots (y/n)?",petechial).
symptom(sordes):-
  positive("Is there accumulation of brown crust on lips (y/n)?",sordes).
symptom(cf_test):-
  write("Does the blood show positive for complement_fixation test"),nl,
  positive("for R_prowazeki organism (y/n)?",cf_test).
symptom(tongue_d):-
  positive("Is the patient's tongue dry and brown (y/n)?",tongue_d).
symptom(scf_test):-
  write("Does the blood show positive specific complement_fixation"),nl,
  positive("test for R.mooseri organism (y/n)?",scf_test).
symptom(rash_f):-
  positive("Does  patient have flat raised bleeding spots (y/n)?",rash_f).
symptom(rash_p6):-
  write("Does the rash appear on wrist,forearms and ankles,then to"),nl,
  positive("back,limbs,chest and finally to abdomen (y/n)?",rash_p6).
symptom(agg_cf3):-
  write("Does  blood show positive for agglutination and complement"),nl,
  positive("_fixation test for R.rickettsi organism (y/n)?",agg_cf3).
symptom(pressure):-
  positive("Does the patient have low blood pressure (y/n)?",pressure).
symptom(leucopenia):-
  write("Does the blood show decreased white blood"),nl,
  positive("cell count (y/n)?",leucopenia).
symptom(virus_2):-
  write("Are the secretion and urine positive for"),nl,
  positive("arena virus (y/n)?",virus_2).
symptom(exudates):-
  write("Is there adherent yellowish secretion on the"),nl,
  positive("black of mouth (y/n)?",exudates).
symptom(blood_skin):-
  write("Is the culture from blood or skin lesion positive"),nl,
  positive("for bartonella bacilliformis organism (y/n)?",blood_skin).
symptom(rash_p7):-
  positive("Is the rash distributed on head and limbs (y/n)?",rash_p7).
symptom(rash_c4):-
  positive("Does the patient have cherry_red eruption (y/n)?",rash_c4).
symptom(blood_film2):-
  write("Is the blood film positive for"),nl,
  positive("borreliae organism (y/n)?",blood_film2).
symptom(blood_film3):-
  write("Is the blood film test positive for"),nl,
  positive("borreliae duttoni organism (y/n)?",blood_film3).
symptom(meningism):-
  positive("Does the patient have tightness in nick (y/n)?",meningism).
symptom(backache):-
  positive("Does the patient have pain in back (y/n)?",backache).
symptom(pain_b):-
 positive("Does the patient have pain in bones (y/n)?",pain_b).
symptom(tongue_2):-
  positive("Is the tongue coated edges are red (y/n)?",tongue_2).
symptom(vomiting_b):-
  positive("Does the patient have vomiting with blood (y/n)?",vomiting_b).
symptom(pain_e):-
  write("Does the patient have pain in upper part of"),nl,
  positive("abdomen (y/n)?",pain_e).
symptom(irritability):-
  positive("Does  patient have mental irritability (y/n)?",irritability).
symptom(urine_p):-
  write("Does the urine contain protein in icreasing quantities,later"),nl,
  positive("casts appear and volume of urine decreases (y/n)?",urine_p).
symptom(blood_t7):-
  positive("Is the blood positive for arbovirus (y/n)?",blood_t7).
symptom(sera):-
  write("Does the test on sera show increasing amount of"),nl,
  positive("specific antibodies (y/n)?",sera).
symptom(pain_lt):-
  positive("Does the patient have pain in limb and trunk (y/n)?",pain_lt).
symptom(rash_e):-
  positive("Does the patient have reddish rash on trunk (y/n)?",rash_e).
symptom(blood_t8):-
  positive("Is  blood positive for R.quintana organism (y/n)?",blood_t8).


hypothesis("Scarlet Fever "):-
  symptom(onset_s),
  symptom(sore_throat),
  symptom(shivering),
  symptom(pyrexia),
  symptom(vomiting),
  symptom(lymph_nodes_t),
  symptom(rash_p1),
  symptom(headache),
  symptom(rash_i),
  p_diagnosis("Scarlet Fever").
hypothesis("Typhid Fever"):-
  symptom(onset_s),
  symptom(malaise),
  symptom(headache),
  symptom(drawsiness),
  symptom(aching_l),
  symptom(vomiting),
  symptom(cough),
  symptom(epistaxis),
  symptom(constipation),
  symptom(tachycardia),
  symptom(rash_c1),
  symptom(rash_p2),
  p_diagnosis("Typhoid Fever").
hypothesis("Paratyphoid Fever"):-
  symptom(onset_s),
  symptom(malaise),
  symptom(headache),
  symptom(drawsiness),
  symptom(aching_1),
  symptom(vomiting),
  symptom(cough),
  symptom(epistaxis),
  symptom(rash_c1),
  symptom(constipation),
  symptom(tachycardia),
  symptom(rash_p2),
  symptom(spleen_ps),
  p_diagnosis("Paratyphoid Fever").
hypothesis("Abortus Fever"):-
  symptom(onset_g),
  symptom(sore_throat),
  symptom(sweating),
  symptom(anorexia),
  symptom(headache),
  symptom(weakness),
  symptom(pain_lb),
  symptom(shivering),
  symptom(pain_j),
  symptom(cough),
  symptom(constipation),
  symptom(spleen_p),
  symptom(arthritis),
  symptom(fever_i),
  p_diagnosis("Abortus Fever").
hypothesis("Rheumatic Fever"):-
  symptom(onset_s),
  symptom(malaise),
  symptom(pain_j),
  symptom(sweating),
  symptom(fatigue),
  symptom(tachycardia),
  symptom(weight),
  symptom(joint_1),
  symptom(polyarth),
  symptom(anorexia),
  symptom(tongue_f),
  symptom(constipation),
  symptom(proteinuria),
  symptom(erythema_c),
  p_diagnosis("Rheumatic Fever").
hypothesis("Cerebrospinal Fever"):-
  symptom(onset_s),
  symptom(headache),
  symptom(vomiting),
  symptom(rash_p3),
  p_diagnosis("Cerebrospinal Fever").
hypothesis("Malaria"):-
  symptom(teeth),
  symptom(pyrexia),
  symptom(headache),
  symptom(vomiting),
  symptom(fever_i),
  symptom(spleen_pt),
  symptom(locality),
  p_diagnosis("Malaria").
hypothesis("Black_water Fever"):-
  symptom(onset_s),
  symptom(fever_i),
  symptom(headache),
  symptom(vomiting),
  symptom(spleen_et),
  symptom(jaundice),
  symptom(liver_et),
  p_diagnosis("Black_water Fever").
hypothesis("Query Fever"):-
  symptom(onset_s),
  symptom(malaise),
  symptom(sweating),
  symptom(anorexia),
  symptom(pyrexia),
  symptom(cough),
  symptom(tachycardia),
  symptom(rash_s),
  symptom(chest),
  symptom(spleen_p),
  p_diagnosis("Query Fever").
```

```
hypothesis("Scrub Typhus Fever"):-
  symptom(onset_s),
  symptom(malaise),
  symptom(cough),
  symptom(headache),
  symptom(weakness),
  symptom(diarrhoea),
  symptom(conjunctival),
  symptom(lymph_nodes),
  symptom(rash_c2),
  symptom(rash_p4),
  p_diagnosis("Scrub Typhus Fever").
hypothesis("Rate_bite Fever"):-
  symptom(wound),
  symptom(sparse),
  symptom(rash_c3),
  p_diagnosis("Rate_bite Fever").
hypothesis("Sandfly Fever"):-
  symptom(onset_s),
  symptom(malaise),
  symptom(headache),
  symptom(consti),
  symptom(aches),
  symptom(eyes),
  symptom(photophobia),
  symptom(conjunctival),
  symptom(nausea),
  symptom(vomiting),
  symptom(anorexia),
  symptom(insomnia),
  symptom(lymph_nodes_s),
  symptom(depression),
  symptom(tachycardia),
  p_diagnosis("Sandfly Fever").
hypothesis("Pharyngoconjunctival Fever"):-
  symptom(conjunctivitis),
  symptom(sore_throat),
  symptom(inflammed),
  symptom(headache),
  symptom(lymph_node_t),
  symptom(shivering),
  symptom(malaise),
  symptom(anorexia),
  p_diagnosis("Pharyngoconjunctival Fever").
hypothesis("Glandular Fever"):-
  symptom(sore_throat),
  symptom(fatigue),
  symptom(malaise),
  symptom(headache),
  symptom(pain_m),
  symptom(pyrexia),
  symptom(lymph_node_s),
  symptom(rash_m),
  symptom(hepatitis),
  symptom(spleen_es),
  p_diagnosis("Glandular Fever").
hypothesis("Haemorrhagic Fever"):-
  symptom(onset_s),
  symptom(urine_cell),
  symptom(skin),
  p_diagnosis("Haemorrhagic Fever").
hypothesis("Epdemic Typhus Fever"):-
  symptom(onset_s),
  symptom(frontal),
  symptom(pain_lb),
  symptom(face),
  symptom(pyrexia),
  symptom(congested),
  symptom(dullness),
  symptom(petechial),
  symptom(sordes),
  symptom(spleen_p),
  symptom(tongue_d),
  symptom(constipation),
  symptom(rash_p5),
  p_diagnosis("Epidemic Typhus Fever").
hypothesis("Endemic Typhus Fever"):-
  symptom(onset_s),
  symptom(pain_lb),
  symptom(face),
  symptom(pyrexia),
  symptom(congested),
  symptom(dullness),
  symptom(constipation),
  p_diagnosis("Endemic Typhus Fever").
hypothesis("Rocky Mountain Spotted Fever"):-
  symptom(onset_s),
  symptom(frontal),
  symptom(pain_lb),
  symptom(face),
  symptom(pyrexia),
  symptom(congested),
  symptom(dullness),
  symptom(rash_f),
  symptom(constipation),
  symptom(rash_p6),
  p_diagnosis("Rocky Mountain Spotted Fever").
hypothesis("Lassa Fever"):-
  symptom(pyrexia),
  symptom(pain_m),
  symptom(tackycardia),
  symptom(pressure),
  symptom(exudates),
  p_diagnosis("Lassa Fever").
hypothesis("Oroya Fever"):-
  symptom(pain_m),
  symptom(pain_j),
  symptom(pyrexia),
  symptom(nausea),
  symptom(vomiting),
  symptom(diarrhoea),
  symptom(spleen_et),
  symptom(liver_et),
  symptom(rash_c4),
  symptom(rash_p7),
  p_diagnosis("Oroya Fever").
hypothesis("Louse_borne Relapsing Fever"):-
  symptom(onset_s),
  symptom(pyrexia),
  symptom(tackycardia),
  symptom(headache),
  symptom(aching),
  symptom(conjunctival),
  symptom(petechial),
  symptom(spleen_pt),
  symptom(liver_pt),
  symptom(epistaxis),
  symptom(jaundice),
  symptom(meningism),
  p_diagnosis("Louse_borne Relapsing Fever").
hypothesis("Tick_borne Relapsing Fever"):-
  symptom(onset_s),
  symptom(fever_f),
  symptom(tackycardia),
  symptom(headache),
  symptom(aching),
  symptom(conjunctival),
  symptom(petechial),
  symptom(spleen_pt),
  symptom(liver_pt),
  symptom(epistaxis),
  symptom(jaundice),
  symptom(meningism),
  p_diagnosis("Tick_borne Relapsing Fever").
hypothesis("Yellow Fever"):-
  symptom(face),
  symptom(pain_b),
  symptom(frontal),
  symptom(vomiting_b),
  symptom(backache),
  symptom(conjunctival),
  symptom(tongue_2),
  symptom(prostration),
  symptom(irritability),
  symptom(photophobia),
  symptom(tackycardia),
  p_diagnosis(" Yellow Fever").
hypothesis("Trench Fever"):-
  symptom(onset_s),
  symptom(headache),
  symptom(pain_lt),
  symptom(rash_e),
  p_diagnosis("Trench Fever").
p_diagnosis("Scarlet Fever"):-
  symptom(throat_swab_1),
  symptom(aso),
  diagnosis("Scarlet Fever"),
  write("Aetiology : Bacterial Infection"),nl,
  write("----------------------TREATMENT----------------------"),nl,
  write("General    : Isolatio,rest and balanced diet"),nl,
  write("                  are of utmost importance."),nl,
  write("Medicine   : Phenoxymethylpenicillin"),nl,
  write("Dose       : 250 mg for children"),nl,
  write("             500 mg for adults"),nl,
  write("             every 6 hours for 7 days"),nl,
  write("Comment    : Erythromycin is indicated for persons"),nl,
  write("             sensitive to penicillin."),nl,
  write("----------------------PREVENTION----------------------"),nl,
  write("An institutional epidemic calls for chemopropylaxis"),nl,
  write("with penicillin."),nl,nl,
  write("Thank you for your visit             Good_bye"),nl.
p_diagnosis("Typhoid Fever"):-
  symptom(blood_t1),
  diagnosis("Typhoid Fever"),
  write("Aetiology  : Bacterial Infection"),nl,
  write("----------------------TREATMENT----------------------"),nl,
  write("General    : Isolation,rest in bed,special attention"),nl,
  write("             to nutration and fluid intake."),nl,
  write("Medicine   : Chloramphenicol"),nl,
  write("Dose       : 3 gm daily for 14 days"),nl,
  write("Comment    : Dose shoud be reduced to 2 g daily as"),nl,
  write("             patient responds."),nl,
  write("---------------------PREVENTION----------------------"),nl,
  write("Those who propose to travel or live in countries where"),nl,
  write("infections are endemic shoud be inoclutated with vaccine"),nl,
  write("cntaining killer S.typhi and S.paratyphi A and B (TAB)."),nl,n
  write("Thank you for your visit             Good_bye"),nl.
p_diagnosis("Paratyphoid Fever"):-
  symptom(blood_t2),
  diagnosis("Paratyphoid Fever"),
  write("Aetiology    : Bacterial Infection"),nl,
  write("----------------------TREATMENT----------------------"),nl,
  write("General      : Isolation,rest in bed, special attention"),nl,
  write("               to nutration and fluid intake."),nl,
  write("Medicine     : Co_trimoxazole"),nl,
  write("Dose         : 3 g daily for 14 days"),nl,
  write("Comment      : Dose shoud be reduced to 2 g daily as the"),nl,
  write("               patient responds."),nl,
  write("----------------------PREVENTION----------------------"),nl,
  write("Those who are propose to travel or live in countries where"),nl,
  write("infections are endemic shoud be inoculated with vaccine"),nl,
  write("containing killed S.typhi and S.paratyphi A and B (TAB)."),nl,nl,
  write("Thank you for your visit              Good_bye"),nl.
p_diagnosis("Abortus Fever"):-
  symptom(radiology),
  symptom(agg_cf),
  diagnosis("Abortus Fever"),
  write("Aetiology     : Bacterial Infection"),nl,
  write("----------------------TREATMENT--------------------"),nl,
  write("Medicine      : Tetracycline"),nl,
  write("Dose          : 500 mg 6 hourly for 21 days"),nl,
  write("Comment       : For chronic or relapsing disease,adaily"),nl,
  write("                dose of tetracycline (1g),streptomycin (1g)"),nl,
  write("                and sulphadimidine (4g) for 2 to 3 weeks."),nl,
  write("----------------------PREVENTION---------------------"),nl,
  write("Undulent fever is prevented by boiling or pasteurisation"),nl,
  write("of all milk used for human consumtion."),nl,nl,
  write("Thank you for your visit             Good_bye"),nl.
p_diagnosis("Rheumatic Fever"):-
  symptom(x_ray),
  symptom(throat_swab_2),
  symptom(aso),
  diagnosis("Rheumatic Fever"),
  write("Aetiology      : Bacterial Infection"),nl,
  write("----------------------TREATMENT---------------------"),nl,
  write("General        : Rest in bed is essential throughout the"),nl,
  write("                 active stage of disease."),nl,
  write("Medicine       : Phenoxymethylpenicillin"),nl,
  write("                 for 7 to 10 days routinly"),nl,
  write("               : Aspirin"),nl,
  write("Dose           : 50 mg per kg body weight 4 hourly daily"),nl,
  write("                 double dose at night"),nl,
  write("Comment        : If toxic symptom,such as nausea,headache,"),nl,
  write("                 dizziness,tinnitus and deafness develod"),nl,
  write("                 the dose must be reduced."),nl,
  write("----------------------PREVENTION---------------------"),nl,
  write("This can be prevented largely in children and adolescents"),nl,
  write("by phenoxymethylpenicillin (125mg b.d) or sulphadimidine"),nl,
  write("(0.5g daily) shoud be taken regularly and continued"),nl,
  write("until about 20 years of age."),nl,nl,
  write("Thank you for your visit                Good_bye"),nl.
p_diagnosis("Cerebrospinal Fever"):-
  symptom(blood_t3),
  symptom(diplococci),
  diagnosis("Cerebrospinal Fever"),
  write("Aetiology          : Bacterial Infection"),nl,
  write("---------------------TREATMENT---------------------"),nl,
  write("Medicine           : Benzylpenicilline"),nl,
  write("                   : given initialy by intravenous injection"),nl,
  write("Comment            : Because of superior penteration of blood_"),nl,
  write("                     brain barrier sulphadimidine is frequently"),nl,
  write("                     not necessarily."),nl,
  write("---------------------PREVENTION---------------------"),nl,
  write("Close contacts of patients with this disease,specially"),nl,
  write("children shoud be given a 5_day course of "),nl,
  write("sulphadimadine."),nl,nl,
  write("Thank you for your visit                      Good_bye"),nl.
p_diagnosis("Malaria"):-
  symptom(blood_film1),
  diagnosis("Malaria"),
  write("Aetiology            : Protozol Infection"),nl,
  write("---------------------TREATMENT---------------------"),nl,
  write("General            : Rest in bed and drinking fluid be provided"),nl,
  write("Medicine           : Aminoquinolines,Chloroquine,or Amodiaquine"),nl,
  write("Dose               : First 600 mg base (4 tablets)"),nl,
  write("                     then 300 mg base in six hours and further"),nl,
  write("                     150mg base twice daily for 3 to 7  days"),nl,
  write("Comment            : If the response to Chloroquine is not good"),nl,
  write("                     the quinine dihydrochloride 650 mg three"),nl,
  write("                     times daily for 2 days shoud be given,"),nl,
  write("                     followed be asingle dose of sulfoxine 1.5 g"),nl
  write("                     combined with pyrimethamine 75 mg."),nl,
  write("---------------------PREVENTION---------------------"),nl,
  write("Control of anopheline mosquitoes especially be spraying of"),nl,
  write("houses with insecticide such as DDT has greatly reduced"),nl,
  write("or abolished the risk of malaria in many areas ,however"),nl,
  write("unless eradication is complete,all visitors and non_immune"),nl,
  write("residents shoud take regular prophlatic or "),nl,
  write("suppressive drugs"),nl,nl,
  write("Thank you for your visit                     Good_bye"),nl.
p_diagnosis("Black_water Fever"):-
  symptom(urine_c),
  diagnosis("Black_water Fever"),
  write("Aetiology              : Protozol Infection"),nl,
  write("--------------------TREATMENT---------------------"),nl,
  write("General                : Physical and mental rest shoud be "),nl,
  write("                         secured ,enough fluid must be given."),nl,
  write("Medicine               : Chloroquine"),nl,
  write("Dose                   : Ampoules of chloroquine in acqueous "),nl,
  write("                         solution cntaining 200 mg in 5 ml"),nl,
  write("                         are avilaible."),nl,nl,
  write("Thank you for your visit                    Good_bye"),nl.
p_diagnosis("Query Fever"):-
  symptom(blood_t4),
  diagnosis("Query Fever"),
  write("Aetiology                : Rickettsial  Infection"),nl,
  write("----------------------TREATMENT---------------------"),nl,
  write("Medicine                 : Tetracycline"),nl,
  write("Dose                     : 250 mg 4 times daily"),nl,
  write("                           500mg 4 times daily for "),nl,
  write("                           severe infection"),nl,
  write("Comment                  : If the temperture high,cold sponging"),nl,
  write("                           gives great confort."),nl,
  write("----------------------PREVENTION--------------------"),nl,
  write("Step shoud be taken to get rid of all lice and fleas"),nl,
  write("and their fauces."),nl,
  write("Active Immunization :- Those likely to be at risk can be"),nl,
  write("protected by vaccines from killed cultures of strains of"),nl,
  write("R.prowazeki,R.mooseri or R.rickettsi culture in eggs."),nl,nl,
  write("Thank you for your visit                    Good_bye "),nl.
 p_diagnosis("Scrub Typhus Fever"):-
  symptom(blood_t5),
  diagnosis("Scrub Typhus Fever"),
  write("Aetiology                : Rickettsial Infection"),nl,
  write("----------------------TREATMENT---------------------"),nl,
  write("General                  : Isolation is very important."),nl,
  write("Medicine                 : Tetracycline"),nl,
  write("Dose                     : 250 mg 4 times daily"),nl,
  write("                           500 mg 4 times daily for"),nl,
  write("                           severe infection"),nl,
  write("Comment                  : If the temperature is high,cold "),nl,
  write("                           sponging gives great confort."),nl,
  write("----------------------PREVENTION--------------------"),nl,
  write("Step shoud be taken to get rid of lice and fleas"),nl,
  write("and their fauces."),nl,
  write("Active Immunization  :- Those likely to be at risk can be"),nl,
  write("protected by vaccives from killed culture of strains of"),nl.
  write("R.prowazeki,R.mooseri or R.rickettsi cultured in eggs."),nl,nl,
  write("Thank you for your visit                 Good_bye "),nl.
 p_diagnosis("Rat_bite Fever"):-
  symptom(s_minus),
  symptom(blood_t6),
  symptom(aso),
  diagnosis("Rat_bite Fever"),
  write("Aetiology                   : Spirochaetal "),nl,
  write("----------------------TREATMENT---------------------"),nl,
  write("Medicine                    : Penicillin,Streptomycin,"),nl,
  write("                              Tetracycline,provide fill care."),nl,nl.
  write("Thank you for your visit                    Good_bye "),nl.
 p_diagnosis("Sandfly Fever"):-
```

```
  symptom(virus_1),
  diagnosis("Sandfly Fever"),
  write("Aetiology                : Viral Infection "),nl,
  write("---------------------TREATMENT---------------------"),nl,
  write("Medicine                 : There is no special treatment."),nl,
  write("Comment                  : Severe pain can be relieved by "),nl,
  write("                           Aspirin or Paracetamal,but "),nl,
  write("                           occasionally opiates are "),nl,
  write("                           required."),nl,nl,
  write("Thank you for your visit                    Good_bye "),nl.
p_diagnosis("Pharyngoconjuctival  Fever"):-
  symptom(throat_swab_3),
  diagnosis("Phoryngoconjuctival  Fever"),
  write("---------------------TREATMENT---------------------"),nl,
  write("General                  : Gargle,bed rest and adequate fluids"),nl,
  write("Medicine                 : Antibiotic are unnecessary."),nl,
  write("Comment                  : Lozenges containing local "),nl,
  write("                           anaesthetic e.g.Benzocaine compound"),nl,
  write("                           lozenges are helpful when throat is"),nl,
  write("                           painful,Amildanalgestic such as"),nl,
  write("                           codiene compound tablets relieves"),nl,
  write("                           systemic symptoms."),nl,nl,
  write("Thank you for your visit                    Good_bye "),nl.
p_diagnosis("Glandular  Fever "):-
  symptom(heterophile),
  symptom(pb_reaction),
  diagnosis("Glandular  Fever "),
  write("Aetiology                : Viral Infection "),nl,
  write("---------------------TREATMENT---------------------"),nl,
  write("General                  : Rest in bed is the best option."),nl,
  write("Comment                  : No treatment is avilaible ."),nl,
  write("                           corticosteroid therapy may be "),nl,
  write("                           helpful if constitutional "),nl,
  write("                           symptoms are severe."),nl,
  write("---------------------PRECAUTION---------------------"),nl,
  write("Sometimes Glandular Fever is mistaken by bacterial "),nl,
  write("sore_throat and Ampicillin is given for treatment "),nl,
  write("it causes askin eruption.It is therefore unsafe to give"),nl,
  write("Ampicillin as adiagostic test."),nl,nl,
  write("Thank you for your visit                    Good_bye "),nl.
p_diagnosis("Haemorrhagic  Fever "):-
  symptom(leucocyte),
  symptom(blood_c),
  diagnosis("Haemorrhagic  Fever "),
  write("Aetiology                : Viral  Infection "),nl,
  write("---------------------TREATMENT---------------------"),nl,
  write("The disease is managed symptomatically oliguria and "),nl,
  write("Anuria are treated as advised for acute renal failure."),nl,nl,
  write("Thank you for your visit                Good_bye  "),nl.
p_diagnosis("Epidemic Typhus Fever"):-
```

```
  symptom(aso),
  symptom(cf_test),
  diagnosis("Epidemic Typhus Fever"),
  write("Aetiology   : Rickettsial Infection "),nl,
  write("----------------------TREATMENT----------------------"),nl,
  write("General     : Isolation is very important."),nl,
  write("Medicine    : Tetracycline "),nl,
  write("Dose        : 250 mg 4 times daily "),nl,
  write("              500 mg 4 times daily for severe infection "),nl,
  write("Comment     : As there is atendency to relapse,"),nl,
  write("              Tetacycline shoud be continued for 5 to 7 "),nl,
  write("              days after the patient is afebrile."),nl,
  write("----------------------PREVENTION----------------------"),nl,
  write("Step shoud be taken to get rid of all lice and fleas "),nl,
  write("and their fauces."),nl,
  write("Active Immunization :- Those likely to be at risk can be "),nl,
  write("protected by vaccines from killed cultures of strains of "),nl,
  write("R.prowazeti,R.mooseri or R.rickettsi cultured in eggs."),nl,nl,
  write("Thank you for your visit                Good_bye "),nl.
p_diagnosis("Endemic Typhus Fever"):-
  symptom(scf_test),
  diagnosis("Endemic Typhus Fever"),
  write("Aetiology     : Rickettsial Infection "),nl,
  write("----------------------TREATMENT----------------------"),nl,
  write("General       : Isolation is very important."),nl,
  write("Medicine      : tetracycline "),nl,
  write("Dose          : 250 mg 4 times daily "),nl,
  write("                500 mg 4 times daily for severe infection "),nl,
  write("Comment       : As there is atendency to relapse, "),nl,
  write("                Tetracycline shoud be continued for 5 to 7 "),nl,
  write("                days after the patient is afebrile."),nl,
  write("----------------------PREVENTION----------------------"),nl,
  write("Step shoud be taken to get rid of all lice and fleas "),nl,
  write("and their fauces."),nl,
  write("Active Immunization :- Those likely to be at risk can be "),nl,
  write("protected by vaccines from killed cultures of strains of "),nl,
  write("R.prowazeti,R.mooseri or R.rickettsi cultured in eggs."),nl,nl,
  write("Thank you for your visit                Good_bye "),nl.
p_diagnosis("Rocky Mountain Spotted Fever "):-
  symptom(agg_cf3),
  diagnosis("Rocky Mountain Spotted Fever "),
  write("Aetiology       : Rickettsial Infection "),nl,
  write("----------------------TREATMENT----------------------"),nl,
  write("General         : Isolation is very important."),nl,
  write("Medicine        : Tetracycline "),nl,
  write("Dose            : 250 mg 4 times daily "),nl,
  write("                  500 mg 4 times daily for severe infection "),nl,
  write("Comment         : As there is atendency to relapse,"),nl,
  write("                  Tetracycline shoud be continued for 5 to 7 "),nl,
  write("                  days after the patient is afebrile."),nl,
  write("-----------------------PREVENTION----------------------"),nl,
  write("Step  shoud be taken to get rid of all lice and fleas "),nl,
  write("and their fauces."),nl,
  write("Active Immunization :- Those likely to be at risk can be "),nl,
  write(" protected by vaccines from killed cultures of strains of "),nl,
  write("R.prowazeti,R.mooseri or R.rickettsi cultured in eggs."),nl,nl,
  write("Thank you for your visit              Good_bye "),nl.
p_diagnosis("Lassa Fever"):-
  symptom(leucopenia),
  symptom(virus_2),
  diagnosis("Lassa Fever"),
  write("Aetiology    : Viral Infection "),nl,
  write("----------------------TREATMENT----------------------"),nl,
  write("General      : Isolation and general supportive measures,"),nl,
  write("               prefrably in an itensive care unit,"),nl,
  write("               are required."),nl,
  write("Comment      : There is no proved specific therapeutic"),nl,
  write("               measure."),nl,nl,
  write("Thank you for your visit              Good_bye "),nl.
p_diagnosis("Oroya Fever"):-
  symptom(blood_skin),
  diagnosis("Oroya Fever"),
  write("Aetiology     : Bacterial Infection "),nl,
  write("----------------------TREATMENT----------------------"),nl,
  write("Medicine      : Penicillin,Streptomycin, or Tetracycline "),nl,
  write("Comment       : Blood transfusion,fluids and electrolytes "),nl,
  write("                may urgently be required."),nl,
  write("-----------------------PREVENTION----------------------"),nl,
  write("The use of insecticides,insect repllants and suitable "),nl,
  write("clothing is advisable for personal protection."),nl,nl,
  write("Thank you for your visit              Good_bye "),nl.
p_diagnosis("Louse_borne Relapsing Fever"):-
  symptom(blood_film2),
  diagnosis("Louse_borne Relapsing Fever"),
  write("Aetiology     : Spirachaetal Infection "),nl,
  write("----------------------TREATMENT----------------------"),nl,
  write("Medicine      : Procaine Penicilline "),nl,
  write("Dose          : 200 mg intramuscularly "),nl,
  write("                0.5 g Tetracycline next day "),nl,
  write("Comment       : The patient must be confind strictly to "),nl,
  write("                bed for 48 hours after treatment."),nl,
  write("-----------------------PREVENTION----------------------"),nl,
  write("The patient and his clothing and all contacts must be "),nl,
  write("freed from lice."),nl,nl,
  write("Thank you for your visit              Good_bye "),nl.
p_diagnosis("Tick_borne Relapsing Fever"):-
  symptom(blood_film3),
  diagnosis("Tick_borne Relapsing Fever"),
  write("Aetiology     : Spirochaetal Infection "),nl,
  write("----------------------TREATMENT----------------------"),nl,
  write("Medicine      : Tetracycline "),nl,
  write("Dose          : 1 g daily for 7 days "),nl,
  write("Comment       : The course must be repeted after an "),nl,
  write("                interval of aweek."),nl,
  write("Ticks can be killed by lindane (gamma BHC) applied to the "),nl,
  write("inside of walls,floors and across the entrance to houses."),nl,nl,
  write("Thank you for your visit               Good_bye "),nl.
p_diagnosis("Yellow Fever"):-
  symptom(urine_p),
  symptom(blood_t7),
  symptom(sera),
  diagnosis("Yellow Fever"),
  write("Aetiology      : Viral Infection "),nl,
  write("----------------------TREATMENT----------------------"),nl,
  write("General        : The patient shoud be nursed under a "),nl,
  write("                 mosquito_net for the first 4 days of "),nl,
  write("                 illness because the blood is infectious."),nl,
  write("Medicine       : There is no specific antiviral agent."),nl,
  write("Comment        : When vomiting is troublesome,dehydration "),nl,
  write("                 shoud be connected by intravenous glucose"),nl,
  write("                 aline and blood trasfusions shoud be givin"),nl,
  write("                 if blood loss is severe."),nl,nl,
  write("Thank you for your visit                  Good_bye "),nl.
p_diagnosis("Tranch Fever"):-
  symptom(blood_t8),
  diagnosis("Tranch Fever"),
  write("Aetiology       : Rickettsial Infection "),nl,
  write("----------------------TREATMENT----------------------"),nl,
  write("Medicine        : Tetracycline "),nl,
  write("Dose              250 mg 4 times daily "),nl,
  write("                  500 mg 4 times daily for severe infection "),nl,
  write("Comment         : If the trmperature is high, cold sponging "),nl,
  write("                  gives great confort."),nl,nl,
  write("Thank you for your visit              Good_bye "),nl.
p_diagnosis(Disease):-
  write("---------------PROVISIONAL DIAGNOSIS---------------"),nl,
  write("The most probable disease is : " ),nl,
  write(Disease),nl,
  write("-----------------------ADVICE-----------------------"),nl,
  write("I am very sorry I could not make"),nl,
  write("a THERAPY RECOMMENDATION for you.Please contact your"),nl,
  write("doctor for further medical conclusion."),nl,nl,
  write("Thank you for your visit             Good_bye "),nl.
diagnosis(Disease):-
  write("---------------------DIAGNOSIS----------------------"),nl,
  write("I have made the following conclusion for you :"),nl,
  write("Disease   ." ,Disease),nl.
```

B - REFERENCES:

1. Avion, B and Feigenbaum, E.A.(1982). The Handbook of Artificial Intelligence. 3 Vols. (Losaltos, CA :William Kaufman), 1:3.

2 Breuker. J H., and Wielinka, B.A.(1984) "Techniques for Knowledge Elicitation and Analysis", Report 1.5, Esprit Project 12, Memorandum 28 Of the Research Project: The Acquisition of Expertise.

3. Bruce, G. Buchanan and Edward H. Shortliffe, "Rule Based Expert Systems", (Reading, M.A.:Addison - Wesley).

4. Brattle Research Corporation, Artificial Intelligence and Fifth Generation Computer Technologies (Boston).

5. Clancey. W.J. and Letsineer, R. (1981) "Neimllcin: Reconfiguring A rule-Based Expert System for Applications to teaching : ITCAI -81.2.

6. Clancey, W.J. and Shortliffe, E.H., (1984) "Reading and Medical Artificial Intelligence": The First Decade (Addison-Weslèy).

7. Elaine Rich (1983) "Artificial Intelligence" (New York: McGraw Hill).

8. Encyclopedia Britannica (1985) Year Book of Science and The Future, S.V. "Artificial Intelligence: Toward Machines that think", By Bruce E. Buchanan.

9 Fagan, L., Shortliffe, E. And Buchanan,B. (1980), "Computer-Based Medical Decision Making from Mycin to VM, Automedica, Vol.3.

10.Gammack, J.G. and Young, R.M., (1985), "Psychological Techniques for Eliciting Expert Knowledge in Research and Development in Expert System", Edited by M.A. Bramer (Cambridge University Press).

11.Hayes-Roth, F., Waterman, D.A. and Lenat, D.B. (1983), "Building Expert Sytems" (Addison-Wesley)

12.Harmon, P. and King D. (1985), "Expert-Systems",(New Year: Wiley Press).

13.Hector, J.L. (1986) :Making Believers out of Computers", Artificial Intelligence 30.

14.Holman, J.G. and Cookson, M.J., (1987), "Expert Systems for Medical Applications", Journal of Medical Engineering & Technology,Vol.11, No.4.

15.Jackson, P.(1986), "Introduction to Expert System", Addison Wesley.

16.Jorgens, J. (1988), "Expert Systems in Clinical Engineering", J. of clinical Engg May/June.

17. Macleod J., (1984), "Davidson's Principles and Practice of Medicine",

18. Mishkoff, H. (1985) "Understanding AI", Dallas TX. Texas Instrument INC.

19. Nilsson, N.J., "Principles of AI", (Springer-Verlae, Heidelberg).

20. Rich, E (1983), A I , New York: McGraw Hill .

21. Shortliffe, E.H. and Buchanan, B.G., (1975), "A Model of Inexact Reasoning in Medicine", Math. Bio. Sci. 23.

22. Shortliffe, E.H., (1976), Computer Based Medical Consultations: Mycin, Elsevier/North Holland, New York.

23. Shortliffe, E.H., Bachanan, B.E and Davis, R. (1977) "Production Roles as a Repersentation for Acknowledge Based Consultation System", AF,8.

24. Sortliffe, E.H., Buchanan, B.G., and Feigenbaum E A., (1979), "Knowledge Engg. for Medical Decision Making A Review of Computer Based Clinical Decision AIDS" Proceeding of the IEEE, 67.

25. Shaw, M.L. and Gaines, B.R (1983), "A Computer Aid to Knowledge Engg."

26. Townsend, C. (1988), "Introduction to Turbo Prolog. BPB Publications.

27. Turbo Prolog, Reference Guide, Version 2.0, IBM, Borland

28. Turbo, Prolog, User's Guide, Version 2.0, IBM, Borland

29. Van Melle, W. (1979), "A Domain Independent Production Rule System for Consultation Programs", IJCAI-79,2.

30. Yu, V.L., Buchanan, B.G., Shortliffe, E.H , Wraith, S.M., Davis, R., Scott, A.C., and Cohen, S.N. (1979), "Evaluation the Performance of A Computer Based Consultant", Comp. Prog. Biomed., 9.