Hall Ticket Number:
14IT705A

IV/IV B.Tech (Regular / Supplementary) DEGREE EXAMINATION

Jan/Feb, 2021 — **Information Technology**

Seventh Semester — **Artificial Intelligence**

Time: Three Hours — **Maximum :** 60 Marks

---

*Answer **All** Questions from Part - A.* (1X12 = 12 Marks)
*Answer **Any FOUR** Questions from Part - B.* (4X12=48 Marks)

Part - A

1 Answer all questions (1X12=12 Marks)

a) What is an AI Technique
b) What is Heuristic function
c) What is ISA relationship
d) Define Forward Reasoning
e) Define AI.
f) List the different types of planning.
g) Define learning
h) Define an expert system.
i) List any two expert systems
j) What is rote learning?
k) What are the advantages of scripts
l) Define declarative knowledge

Part - B

2 a) Explain about Water jug problem. 6M
b) Compare Breadth-First Search and Depth-First Search 6M

3 a) Explain means-ends analysis Procedure with algorithm. 6M
b) Write the algorithm for Generate and Test 6M

4 a) Compare and Contrast Procedural knowledge and Declarative knowledge. 6M
b) Write the resolution procedure for prepositional logic 6M

5 a) Explain in detail about Restaurant script. 6M
b) Discuss about Control Knowledge in detail? 6M

6 a) Discuss about partition of semantic nets with an example 6M
b) Construct semantic net representation for the following:
- Pompeian(Marcus), Blacksmith(Marcus)
- Mary gave the green flowered vase to her favorite cousin. 6M

7 a) Briefly explain about rote learning. 6M
b) Describe about goal stack planning with an example 6M

8 a) Write short notes on the
i) Learning from examples ii) Learning in problem solving 6M
b) Discuss about Knowledge acquisition. 6M

9 a) Draw neat diagram of Expert system and its applications. 6M
b) Explain about Expert System Shells 6M

1 Answer all questions (1X12=12 Marks)

a) **What is an AI Technique**
Ans: **AI Technique** is a manner to organize and use the knowledge efficiently

b) **What is Heuristic function**
Ans: The **heuristic function** is a way to inform the search about the direction to a goal. It provides an informed way to guess which neighbour of a node will lead to a goal.

c) **What is ISA relationship**
Ans:
The semantic representation consists of object and arc nodes. The **relationship** that is consisted of this network is the **Isa**-instance **relationship**

d) **Define Forward Reasoning**
Ans:
Forward chaining is a popular implementation strategy for expert systems

e) **Define AI.**
Ans:
Artificial intelligence or ***AI*** refers to the simulation of human intelligence in machines that are programmed to think and act like humans.

f) **List the different types of planning**.
Ans:
1.Goal stack Planning
2.Hierarchical planning

g) **Define learning**
Ans:
Learning is the process of acquiring
new understanding, knowledge, behaviours, skills, values, attitudes, and preferences

h) **Define an expert system.**
Ans:
In artificial intelligence, an ***expert system*** is a computer ***system*** emulating the decision-making ability of a human ***expert***. ***Expert systems*** are designed to solve complex problems by reasoning

i) **List any two expert systems**
Ans:
MYCIN: This was one **of the** earliest **expert systems that** was based on backward chaining.
DENDRAL: This was an **AI** based **expert system** used essentially for chemical analysis

j) **What is rote learning?**

Ans:

Rote methods are routinely used when fast memorization is required, such as learning one's lines in a play or memorizing a telephone number.

k) **What are the advantages of scripts**

Ans:

Ability to predict events.

A single coherent interpretation may be build up from a collection of observations.

l) **Define declarative knowledge**

Ans:

In epistemology, descriptive knowledge is knowledge that can be expressed in a declarative sentence or an indicative proposition. "Knowing-that" can be contrasted with "knowing-how",

Part - B

2a) Explain about Water jug problem. **6m**

Description-3

Problem solving-3

State representation: **(x, y)**

— x: Contents of four gallon

— y: Contents of three gallon

Start state: **(0, 0)**

Goal state **(2, n)**

Operators

— Fill 3-gallon from pump, fill 4-gallon from pump

— Fill 3-gallon from 4-gallon , fill 4-gallon from 3-gallon

— Empty 3-gallon into 4-gallon, empty 4-gallon into 3-gallon

— Dump 3-gallon down drain, dump 4-gallon down drain

State Space Search: Water Jug Problem...

1. (x, y) if $x < 4$ → (4, y) if x is less than 4, fill the 4 gallon jug
2. (x, y) if $y < 3$ → (x, 3) fill the 3-gallon jug
3. (x, y) if $x > 0$ → $(x - d, y)$ pour some water out of the 4 gallon jug
4. (x, y) if $y > 0$ → $(x, y - d)$ pour some water out of the 3 gallon jug
5. (x, y) if $x > 0$ → (0, y) empty the 4-gallon jug on the ground
6. (x, y) if $y > 0$ → (x, 0) empty the 3-gallon jug on the ground

7.	(x, y) if $x + y \geq 4, y > 0$	$\rightarrow (4, y - (4 - x))$	pour water from the 3- gallon jug into the 4-gallon jug until the 4-gallon jug is full
8.	(x, y) if $x + y \geq 3, x > 0$	$\rightarrow (x - (3 - y), 3)$	pour water from the 4- gallon jug into the 3-gallon jug until the 3-gallon jug is full
9.	(x, y) if $x + y \leq 4, y > 0$	$\rightarrow (x + y, 0)$	pour all the water from the 3-gallon jug into the 4-gallon jug
10.	(x, y) if $x + y \leq 3, x > 0$	$\rightarrow (0, x + y)$	pour all the water from the 4-gallon jug into the 3-gallon jug
11.	$(0, 2)$	$\rightarrow (2, 0)$	pour the 2 gallons from the 3-gallon jug into the 4-gallon jug
12.	$(2, y)$	$\rightarrow (0, y)$	empty the 2 gallons in the 4 gallon jug on the ground

2b Compare Breadth-First Search and Depth-First Search **6M**

BFS-3

DFS-3

DFS:

1. The search begins by expanding the initial node, generate all successors of the initial node and test them.
2. Depth-first search always expands the deepest node in the current frontier of the search tree.
3. Depth-first search uses a **LIFO** approach.

BFS:

1. Searching processes level by level unlike depth first search which goes deep into the tree.
2. An operator is employed to generate all possible children of a node

3 a) Explain means-ends analysis Procedure with algorithm. 6M

Means-Ends Analysis. This **means** we could solve major parts of a problem first and then return to smaller problems when assembling the final solution. ... STRIPS (A robot Planner) is an advanced problem solver that incorporates **means-ends analysis** and other techniques

- **Means-Ends Analysis**
- Allows both backward and forward searching.
- This means we could solve major parts of a problem first and then return to smaller problems when assembling the final solution.
- GPS was the first AI program to exploit means-ends analysis.
- STRIPS (A robot Planner) is an advanced problem solver that incorporates means-ends analysis and other techniques.
- Very loosely the means-ends analysis algorithm is:
- Until the goal is reached or no more procedures are available:
  - Describe the current state, the goal state and the differences between the two.
  - Use the difference the describe a procedure that will hopefully get nearer to goal.
  - Use the procedure and update current state.
- If goal is reached then **success** otherwise **fail**.

b) **Write the algorithm for Generate and Test** 6M

Algorithm

1. Generate a possible solution.
2. Test to see if this is actually a solution.
3. Quit if a solution has been found. Otherwise, return to step 1.

Generate-and-Test

- Exhaustive generate-and-test.
- Heuristic generate-and-test: not consider paths that seem unlikely to lead to a solution.
- Plan generate-test:
  - Create a list of candidates.
  - Apply generate-and-test to that list.

4 a) Compare and Contrast Procedural knowledge and Declarative knowledge. 6M

Procedural knowledge-3

Declarative knowledge-3

S.NO	Procedural Knowledge	Declarative Knowledge
1.	It is also known as Interpretive knowledge.	It is also known as Descriptive knowledge.
2.	Procedural Knowledge means how a particular thing can be accomplished.	While Declarative Knowledge means basic knowledge about something.
3.	Procedural Knowledge is generally not used means it is not more popular.	Declarative Knowledge is more popular.
4.	Procedural Knowledge can't be easily communicate.	Declarative Knowledge can be easily communicate.
5.	Procedural Knowledge is generally process oriented in nature.	Declarative Knowledge is data oriented in nature.
6.	In Procedural Knowledge debugging and validation is not easy.	In Declarative Knowledge debugging and validation is easy.
7.	Procedural Knowledge is less effective in competitive programming.	Declarative Knowledge is more effective in competitive programming.

b) Write the resolution procedure for prepositional logic 6M

Resolution in Propositional Logic

1. Convert all the propositions of KB to clause form (S).

$$L_1 \vee L_2 \vee \ldots \vee L_n$$

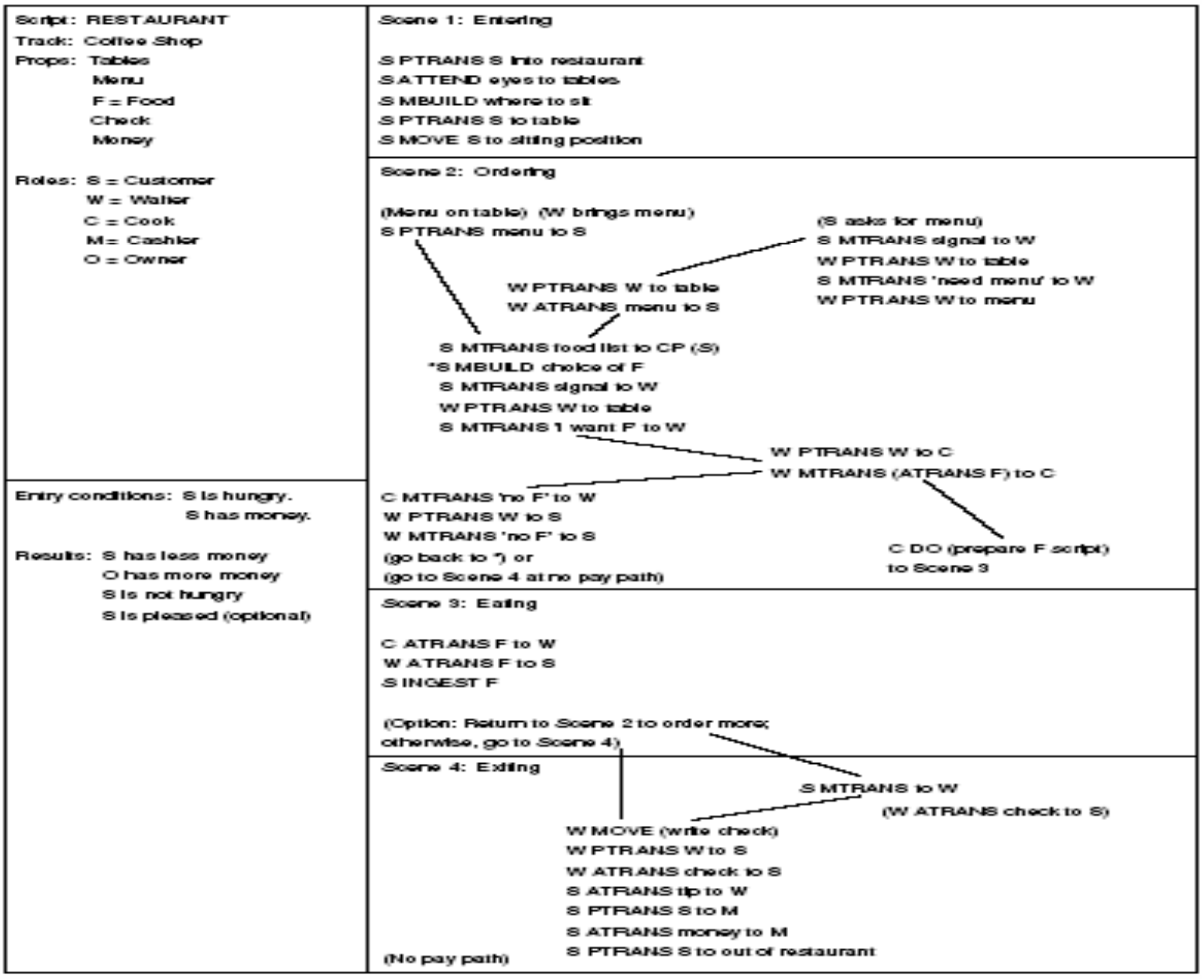
P or ¬P

Resolution in Propositional Logic

1. Convert all the propositions of KB to clause form (S).

2. Negate α and convert it to clause form. Add it to S.
3. Repeat until either a contradiction is found or no progress can be made:
   a. Select two clauses $(\alpha \vee \neg P)$ and $(\gamma \vee P)$.
   b. Add the resolvent $(\alpha \vee \gamma)$ to S.

5 a) Explain in detail about Restaurant script. 6M



b) **Discuss about Control Knowledge in detail?** 6M

Description-3

Explanation-3

Knowledge about which paths are most likely to lead quickly to a goal state is often called search control knowledge.

– Which states are more preferable to others?

– Which rule to apply in a given situation.

– The order in which to pursue sub goals

– Useful sequences of rules to apply.

Search control knowledge = Meta knowledge

There are a number of AI systems that represent their control knowledge with rules. Example SOAR, PRODIG. SOAR is a general architecture for building intelligent systems.

1. Long-term memory is stored as a set of productions (or, rules).
2. Short-term memory (also called *working memory)* is a buffer that is affected by perceptions and serves as a storage area for facts deduced by rules in long-term memory. Working memory is analogous to the state description in problem solving.
3. All problem-solving activity takes place as state space traversal. There are several classes of problem-solving activities, including reasoning about which states to explore, which rules to apply in a given situation, and what effects those rules will have.
4. All intermediate and final results of problem solving are remembered (or, *chunked)* for future reference.[4]

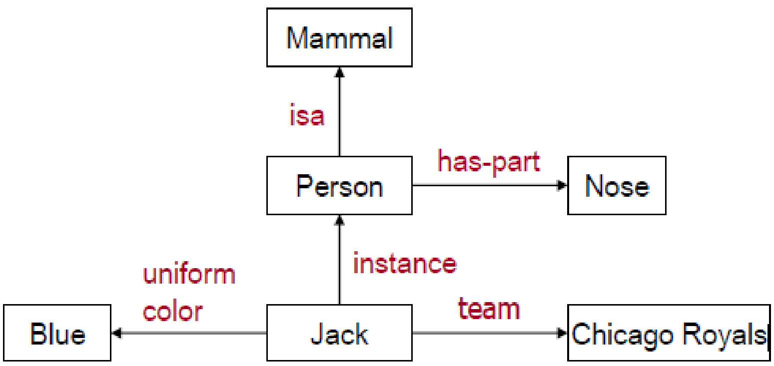
6 a) Discuss about partition of semantic nets with an example 6M

Description-3

Diagram-3

Semantic nets

- In semantic nets information is represented as:
  - set of nodes connected to each other by a set of labelled arcs.
- **Nodes represent**: various objects / values of the attributes of object .
- **Arcs represent:** relationships among nodes.



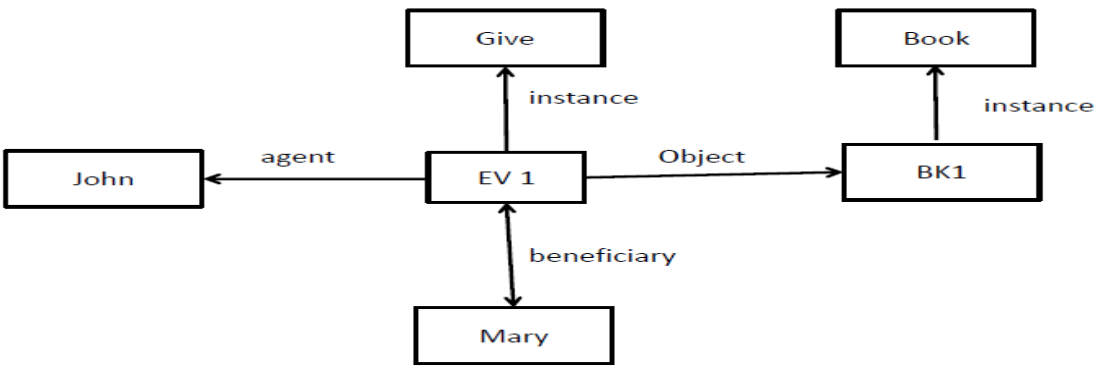
b) **Construct semantic net representation for the following:** 6M

Description-3

Diagram-3

- Pompeian(Marcus), Blacksmith(Marcus)
- Mary gave the green flowered vase to her favorite cousin.
  Man(Marcus) can be converted into:
  instance(Marcus, Man)

Ex. 2. “john gave the book to Mary”
give(john,mary,book)



7 a) **Briefly explain about rote learning.** **6**m

Description-3

Explanation-3

Rote Learning is basically *memorization.*

- Saving knowledge so it can be used again.
- Retrieval is the only problem.
- No repeated computation, inference or query is necessary.

A simple example of rote learning is caching

- Store computed values (or large piece of data)
- Recall this information when required by computation.
- Significant time savings can be achieved.
- Many AI programs (as well as more general ones) have used caching very effectively.

Memorization is a key necessity for learning:

- It is a basic necessity for any intelligent program -- is it a separate learning process?
- Memorization can be a complex subject -- how best to store knowledge?

Samuel's Checkers program employed rote learning (it also used parameter adjustment which will be discussed shortly).

- A minimax search was used to explore the game tree.
- Time constraints do not permit complete searches.
- It records board positions and scores at search ends.

- o Now if the same board position arises later in the game the stored value can be recalled and the end effect is that deeper searched have occurred.

Rote learning is basically a simple process. However it does illustrate some issues that are relevant to more complex learning issues.

b) Describeaboutgoal stack planning with an example 6m
Description-3
Problem explanation-3

Basic Idea to handle interactive compound goals uses goal stacks, here the stack contains:

- goals,
- operators -- ADD, DELETE and PREREQUISITE lists
- a database maintaining the current situation for each operator used.

Consider the following where wish to proceed from the *start* to *goal* state.

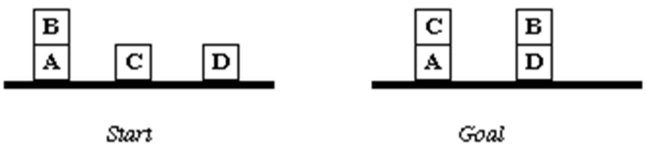


Fig. 24 Goal Stack Planning Example

We can describe the *start* state:

ON(*B*, *A*) ∧ ONTABLE(*A*) ∧ ONTABLE(*C*) ∧ ONTABLE(*D*) ∧ ARMEMPTY

and *goal* state:

ON(*C*, *A*) ∧ ON(*B*,*D*) ∧ ONTABLE(*A*) ∧ ONTABLE(*D*)

- Initially the goal stack is the goal state.
- We then split the problem into four sub problems
- Two are solved as they already are true in the initial state -- ONTABLE(*A*), ONTABLE(*D*).

- With the other two -- there are two ways to proceed:

1. ON(C,A)

   ON(*B*,*D*)

   ON(*C*,*A*) ∧ ON(*B*,*D*) ∧ ONTABLE(*A*) ∧ONTABLE(*D*)

2. ON(*B*,*D*)

ON(C,A)

ON(C,A) $\wedge$ ON(B,D)

$\wedge$ ONTABLE(A) $\wedge$

ONTABLE(D)

The method is to

- Investigate the first node on the stack ie the top goal.
- If a sequence of operators is found that satisfies this goal it is removed and the next goal is attempted.
- This continues until the goal state is empty.

8 Write short notes on following **6m**

i) Learning from examples-3m ii) Learning in problem solving-3m

I .ANS) Learning from examples

Example Learning System – FOO

Learning the game of hearts

FOO (First Operational Operationalize) tries to convert high level advice (principles, problems, methods) into effective executable (LISP) procedures.

Hearts:

- Game played as a series of tricks.
- One player - who has the lead - plays a card.
- Other players follow in turn and play a card.

  The player must follow suit.

  If he cannot he play any of his cards.

- The player who plays the highest value card wins the trick and the lead.
- The winning player takes the cards played in the trick.
- The aim is to avoid taking points. Each heart counts as one point the queen of spades is worth 13 points.
- The winner is the person that after all tricks have been played has the lowest points score.

Hearts is a game of partial information with no known algorithm for winning.

Although the possible situations are numerous general advice can be given such as:

- Avoid taking points.
- Do not lead a high card in suit in which an opponent is void.
- If an opponent has the queen of spades try to flush it.

In order to receive advice a human must convert into a FOO representation (LISP clause) (avoid (take-points me) (trick))

FOO *operationalises* the advice by translating it into expressions it can use in the game. It can UNFOLD avoid and then trick to give:

(achieve (not (during

(scenario

(each p1 (players) (play-card p1))

(take-trick (trick-winner)))

(take-points me))))

II **ANS**) Learning in problem solving

There are three basic methods in which a system can learn from its own experiences.

- Learning by Parameter Adjustment.
- Learning by Macro Operators.
- Learning by Chunking.

Learning by Parameter Adjustment

Many programs rely on an evaluation procedure to summarize the state of search etc. Game playing programs provide many examples of this.

However, many programs have a static evaluation function.

In learning a slight modification of the formulation of the evaluation of the problem is required.

Here the problem has an evaluation function that is represented as a polynomial of the form such as:

$$c_1 t_1 + c_2 t_2 + c_3 t_3 + \ldots$$

The *t* terms a values of features and the c terms are weights.

In designing programs it is often difficult to decide on the exact value to give each weight initially.

So the basic idea of idea of parameter adjustment is to:

- Start with some estimate of the correct weight settings.
- Modify the weight in the program on the basis of accumulated experiences.
- Features that appear to be good predictors will have their weights increased and bad ones will be decreased.

Samuel's Checkers programs employed 16 such features at any one time chosen from a pool of 38.

Learning by Macro Operators

The basic idea here is similar to Rote Learning:

Avoid expensive recompilation

Macro-operators can be used to group a whole series of actions into one.

For example: Making dinner can be described a lay the table, cook dinner, serve dinner. We could treat laying the table as on action even though it involves a sequence of actions.

The STRIPS problem-solving employed macro-operators in its learning phase.

Consider a blocks world example in which ON(C, B) and ON (A, TABLE) are true.

STRIPS can achieve ON (A, B) in four steps:

UNSTACK(C, B),

PUTDOWN(C),

PICKUP (A),

STACK (A, B)

STRIPS now builds a macro-operator MACROP with preconditions ON(C,B), ON(A,TABLE), post conditions ON(A,B), ON(C,TABLE) and the four steps as its body.

MACROP can now be used in future operation.

But it is not very general. The above can be easily generalised with variables used in place of the blocks.

However generalisation is not always that easy (See Rich and Knight).

Learning by Chunking

Chunking involves similar ideas to Macro Operators and originates from psychological ideas on memory and problem solving.

The computational basis is in production systems (studied earlier).

SOAR is a system that use production rules to represent its knowledge. It also employs chunking to learn from experience.

b) **Discuss about Knowledge acquisition.** **6m**

Description-3

Explanation-3

Typically, a knowledge engineer interviews a domain expert to elucidate expert knowledge, which is then translated into rules. After the initial system is built, it must be iteratively refined until it approximates expert-level performance. This process is expensive and time-consuming, so it is worthwhile to look for more automatic ways of constructing expert knowledge bases. While no totally automatic knowledge acquisition systems yet exist, **there are many programs that interact with domain experts to extract expert knowledge efficiently.**

These programs provide support for the following activities:

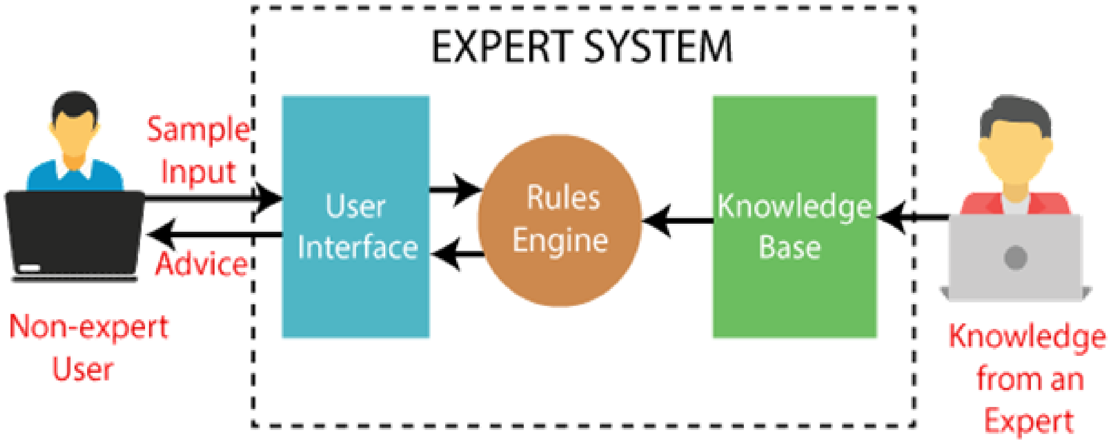
- Entering knowledge.
- Maintaining knowledge base consistency.
- Ensuring knowledge base completeness

The most useful knowledge acquisition programs are those that are restricted to a particular problem-solving paradigm, e.g., diagnosis or design. It is important to be able to enumerate the roles that knowledge can play in the problem-solving process.

9 a **Draw neat diagram of Expert system and its applications.** **6m**

Diagram-3m

Applications-3m



o

o

Designing and manufacturing domain

It can be broadly used for designing and manufacturing physical devices such as camera lenses and automobiles.

In the knowledge domain

These systems are primarily used for publishing the relevant knowledge to the users. The two popular ES used for this domain is an advisor and a tax advisor.

- **In the finance domain**

  In the finance industries, it is used to detect any type of possible fraud, suspicious activity, and advise bankers that if they should provide loans for business or not.

- **In the diagnosis and troubleshooting of devices**

  In medical diagnosis, the ES system is used, and it was the first area where these systems were used.

- **Planning and scheduling**

  The expert systems can also be used for planning and scheduling some particular tasks for achieving the goal of that task.

b **Explain about Expert System Shells** **6**m

Description-3m

Explanation-3m

Initially, each expert system that was built was created from scratch, usually in LISP. But, after several systems had been built this way, it became clear that these systems often had a lot in common.

In particular, since the systems were constructed as a set of declarative representations (mostly rules) combined with an interpreter for those representations, **it was possible to separate the interpreter from the domain-specific knowledge and thus to create a System that could be used to construct new expert systems by adding new knowledge corresponding to the new problem domain. The resulting interpreters are called shells**.

One influential example of such a shell is EMYCIN (for Empty MYCIN) [Buchanan and Shortleaf, 1984], which was derived from MYCIN.

- These shells provide much greater flexibility in representing knowledge and in reasoning with it than MYCTN did.
- They typically support rules, frames, truth maintenance systems, and a variety of other reasoning mechanisms.

Signature of the HOD.

Signature of the internal Examiner (B. Krishnaiah)

Name of the external Examiners	Name of the college	Dept.	Signature