Unit :- 2 Knowledge Representation Issues Subject :- Intelligent System

⇒ Knowledge Representation and Mappings :-

A representation is a way of describing certain fragment or information so that any reasoning system can easily adopt it for inferencing purposes.

A knowledge representation system should provide ways of representing complex knowledge and should possess the following characteristics :

- The representation scheme should have a set of well-defined syntax and semantics.
- The knowledge representation scheme should have a good expressive capacity.
- Representation must be efficient.

In order to solve the complex problems, one needs both a large ~~amount~~ amount of knowledge and some mechanisms for manipulating that knowledge to create solutions to new problem.

We represents the knowledge in the form of the facts in AI. We all dealing with two entities :

- facts : truth in some relevant world. These are the things we want to represent.
- Representations of the facts in some chosen formalism.

One way to think of ~~structuring~~ structuring these entities is as two levels:

→ The knowledge level, at which facts are described.

→ The symbol level, at which representations of objects at the knowledge level are defined in terms of symbols that can be manipulated by programs.

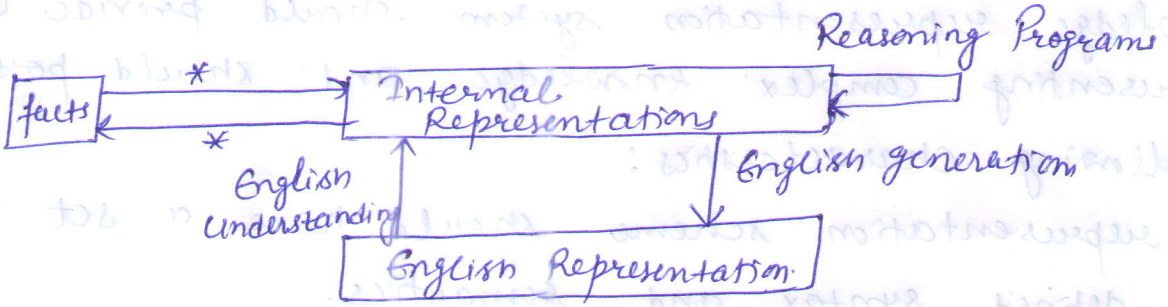


Figure:- Mappings between facts and Representations

Here, we will focus on facts, on representations and on the two way mappings that must exists between them. We will call these links representation mappings.

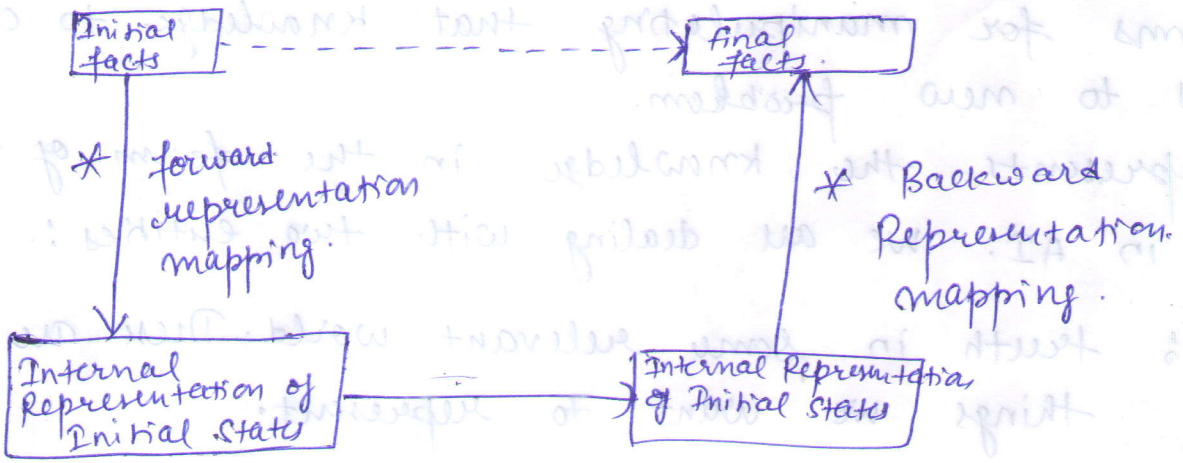


Figure:- Representation of the facts

* Forward Representation mapping :- maps from facts to representation
* Backward Representation mapping :- maps from representation to facts.

Consider the english sentence :-

Spot is a dog.

Representation in logic : dog (Spot).

all dog has tails → $\forall x$: dog (x) → hastail (x)

→ using deductive mechanisms of logic, we may generate the new representation object :

hastail (Spot).

→ using backward mapping function, we could then generate the English sentence.

Spot has a tail.

⇒ PROCEDURAL VS DECLARATIVE REPRESENTATION :-

Declarative knowledge refers to the information that an expert knows ~~of~~ from different sources such as manuals or from experience.

for Example :-

" All Dog has tail "

" Spot is a Dog "

This can be represented using a declarative representation as

$\forall x$ (Dog (x) → hastail (x))

Spot (Dog)

for example :- " All carnivorous have sharp teeth ",
" Cheetah is a carnivore ".

This can be represented using a declarative representation as ;

$\forall x$ (carnivore (x) $\rightarrow$ sharp_teeth (x))

carnivore (Cheetah)

using two representation, it is possible to deduce that " Cheetah has sharp teeth ".

=> A procedural representation, represents knowledge as procedures and inferencing mechanism manipulate these procedure to arrive at result.

```
procedure carnivore (x);
    If (x = cheetah ) then return true
    else return false
end procedure carnivore (x).

procedure sharp_teeth (x);
    if carnivore (x) then return true
    else return false
end procedure sharp_teeth (x).
```

To see whether cheetah has sharp teeth, one should activate procedure sharp-teeth with variable x instaniated to value cheetah. This procedure calls procedure carnivore (x) in turn with value of (x = cheetah). Procedure carnivore returns a value and so is procedure sharp-teeth.

Unit :- Subject :- Intelligent System

⇒ **Representing knowledge :-** Representing knowledge is concerned with how to record knowledge in the computer. This is a central technical issue because it dictates how understandable the knowledge will, how easy it will be to modify, and how efficiently it can be accessed by the computer. The best-known representation schemes are :

1) Associative Networks or semantic networks.
2) Frames
3) Conceptual Dependency
4) Scripts.

1) Associative Networks or semantic networks :-

A semantic network or semantic net is a structure for representing knowledge as a pattern of interconnected nodes and arcs.

It is also known as graphical representation of the knowledge as shown in figure. The objects serves as the node and the relationship with another node gives the Arcs.

for example :- Cheetah is a carnivore.

Here cheetah and carnivore are objects and the relationship between the objects is described by the term is_a.

The following are the rules about node and arcs apply to semantic networks.

a) Node in semantic network can be either
   (i) States (ii) Entities (iii) Attributes (iv) Events.
b) Arcs in the net gives the relationship between the nodes and labels on the arcs show what type of relationship exists.

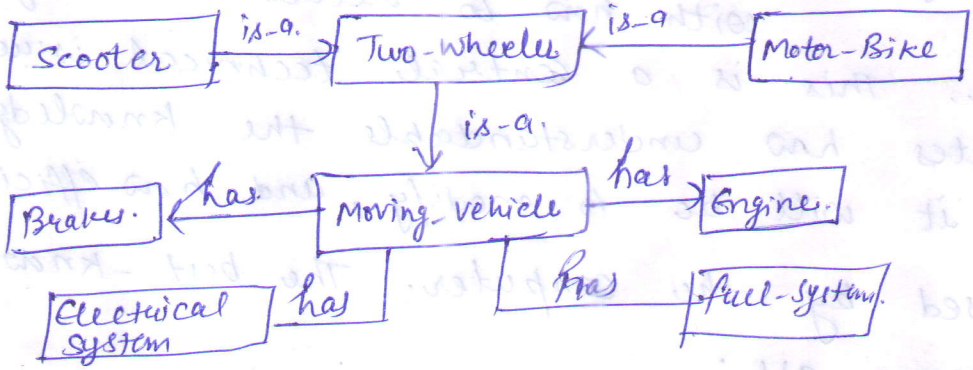


figure :- A sample semantic net

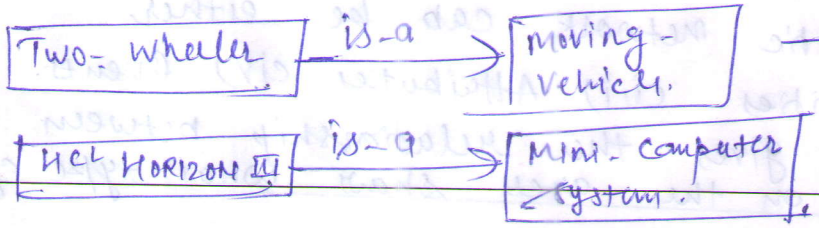
TYPES OF NODES IN SEMANTIC NET :-

→ Generic NODE
→ Individual or Instance Nodes.

* A generic node is a very general node. In figure for the semantic network of computer centre, the mini-computer system is a generic node because many

many mini computer systems exists.

* Individual or Instance nodes are specific instances of a generic node.
In figure, HCL Horizon III is an individual node because it is very specific instance of mini-computer system.



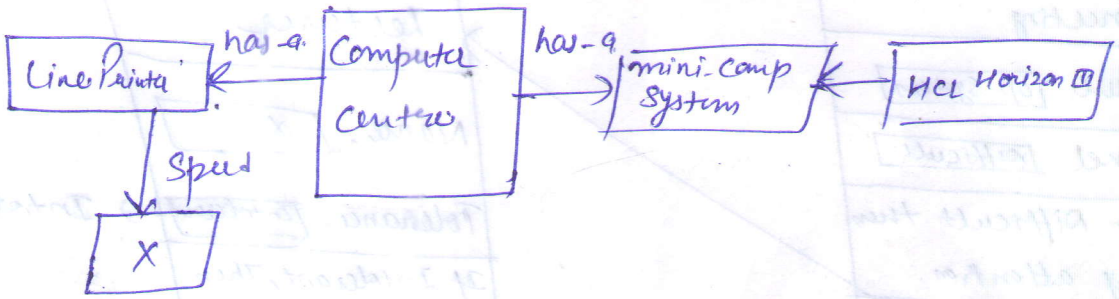
Unit :-

Subject :- Intelligent System

= **Reasoning using semantic Networks** :- Reasoning in semantic network is an easy task. firstly, we have to define the intial node, other nodes are pursued using the links until the final node is reached.

In figure, If one wishes to find " what is speed of the line printer?"

The reasoning mechanism has to find the node line printer and identify the are that has characteristics " Speed."



X has a variable value which could be a numeric one.

=2) **FRAMES** :- Frames are used to represent a combination of descriptive and operational knowledge

Each frame represents a combination of descriptive and operational knowledge.

A frame can be defined as a data structure that has slots for various objects and a collection of frames consists of expectations of a given situation.

Semantic nets initially we used to represent labelled connections between objects. As tasks became more complex the representation need to be more structured.

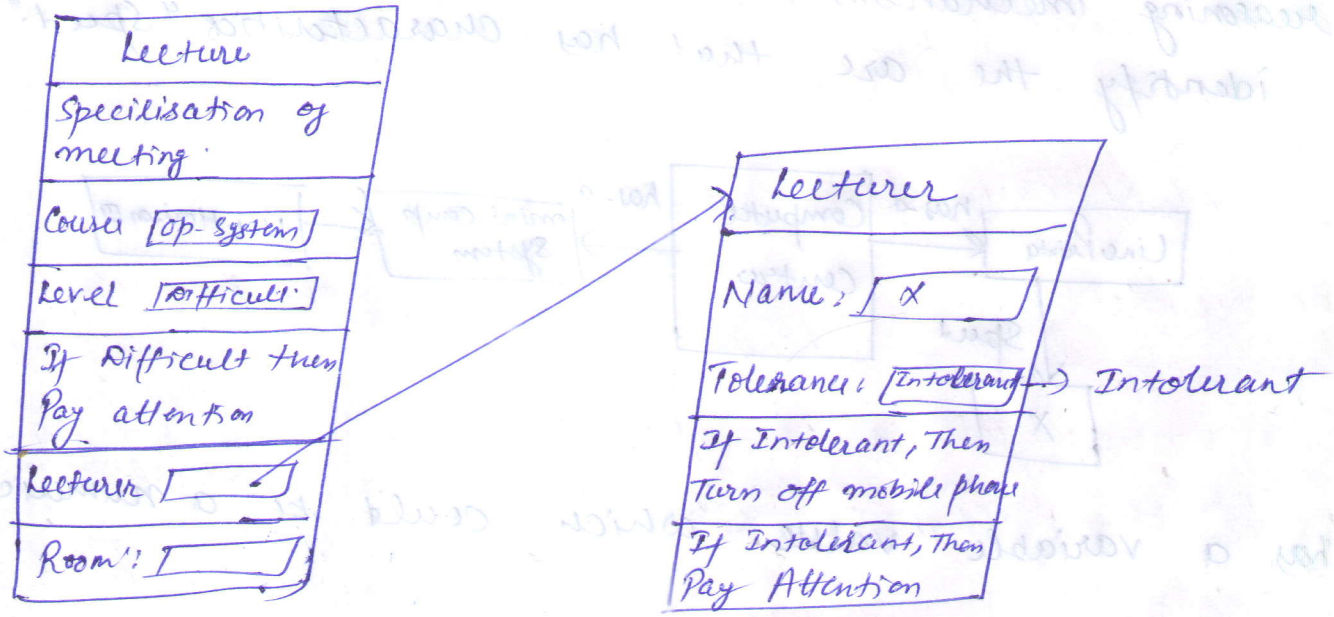
Unit :-

Subject :- Intelligent System

A frame is collection of attributes or slots and associated value that describe some real world entity.

Each frame represents :

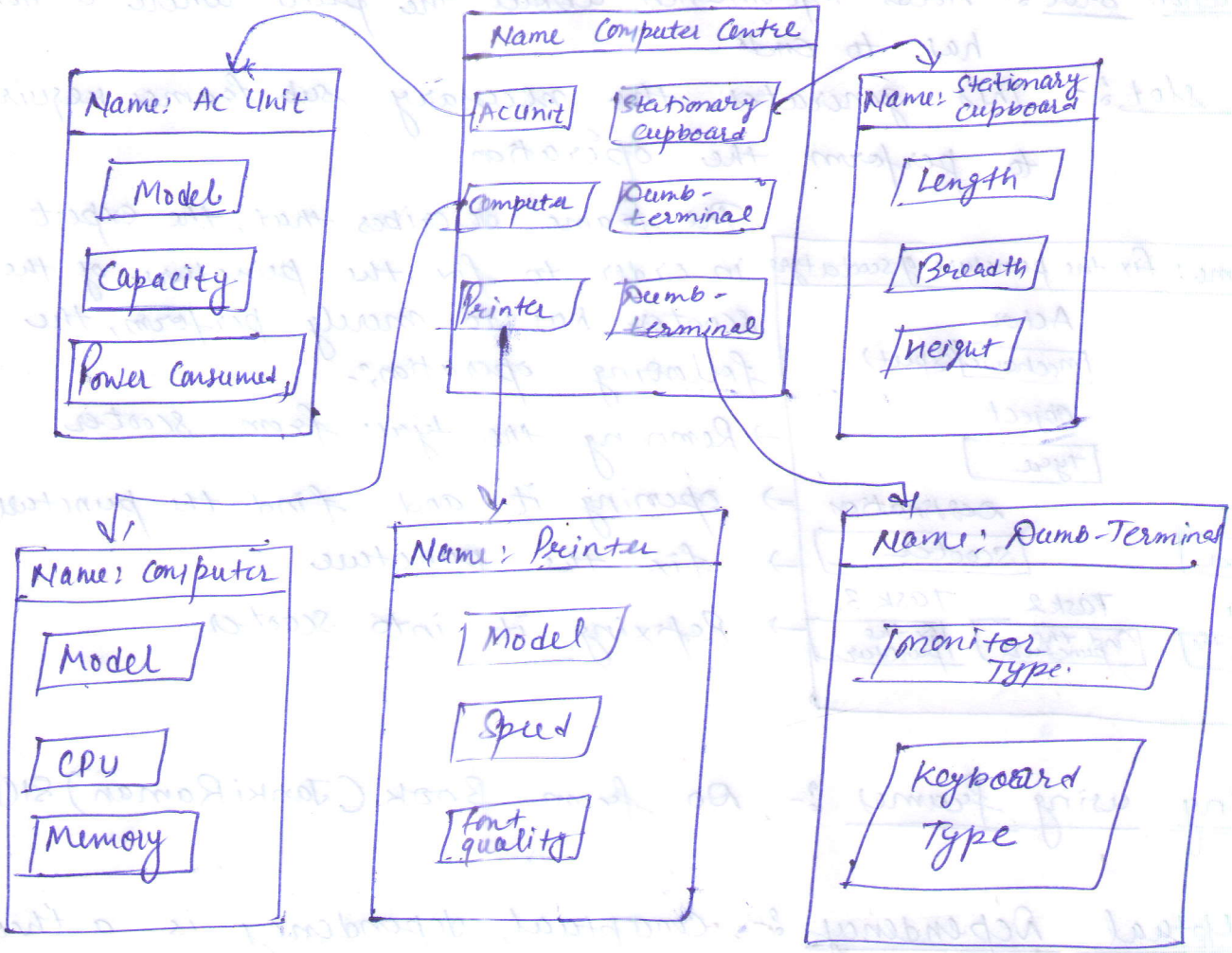
- a class (set), or
- an instance (an element of a class)



Types of frames :-

- Declarative / factual frames
- Procedural frame.

(a) Declarative / factual frame :- A frame the merely contains description about objects is called declarative type / factual / situational frame.



b) Procedural frame :- It is also possible to attach slots which explain how to perform things. In other words, it is possible to have procedural knowledge represented in a frame. Such frame which have procedural knowledge embeded in it are called action - procedure frame. The action-frame has the following slots :-

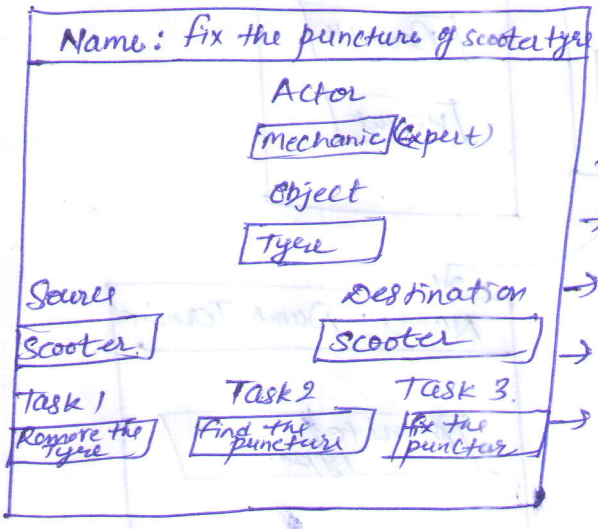
→ Actor slot :- which holds the information about who is performing the activity.

→ Object slots : This frame has information about the item to be operated on.

→ Source slots :- Source slots holds the information from where the action has to begin.

→ Destination slot :- Holds information about the place where action has to ends.

→ Task slot :- This generates the necessary sub-frames required to perform the operation.



The frame describes that, the expert in order to fix the puncture of the scooter has to merely perform, the following operations:-

→ Removing the tyre from scooter
→ opening it and find the puncture
→ fix the puncture
→ Refixing it into scooter.

⇒ Reasoning using frames :- Do from Book (Janki Raman) 81 (Page no)

13) Conceptual Dependency :- Conceptual dependency is a theory of natural processing which mainly deals with representation of semantics of a language. The main motivation for the development of CD as a knowledge representation technique are given below :-

→ to construct the computer programs that can understand natural language.
→ to make inference from the statements and also to identify conditions in which two sentence can have similar meaning.
→ to provide facilities for the system to take part in dialogues and questions answers.
→ to provide the necessary plank that sentence in one language can be easily translated into other languages
→ to provide a means of representation which are language independent.

Example:- 1) Consider two sentences:-

"Ravinder gave Rajni a ball" and

"Rajni took a ball from Ravinder"

the structure of statements are different but meaning is same.

CD Diagram for both statements:-

Primitive	ATRANS	ATRANS
Actor	RAVINDER	RAJNI
Object	BALL	BALL
Source	RAVINDER	RAVINDER
Destination	RAJNI	RAJNI
Ravinder Gave Rajni A Ball		Rajni took a ball from Ravinder

'ATRANS' is a particular primitive that represents the action of transfer of an abstract relationship.

2) Sentence:- "I went to three chemist this morning."

This would generally lead to hearer to assume that I failed to get what I wanted at the first two, and this inference must be made part of the meaning of the phrase.

Primitive CD forms

1) ATRANS: Transfer of abstract relationship (e.g give)
2) PTRANS: Transfer of physical location of an object (e.g go)
3) PROPEL: Appn of physical force of an object (e.g throw)
4) MOVE: Movement of a body part of animal by the animal (e.g kick)
5) GRASP: Grasping of an object by an actor (e.g hold)
6) INGEST: Taking of an object by an animal to inside of that animal (i.e drink, eat)
7) EXPEL: Expulsion of an object from inside the body by an animal to the world (spit)
8) MTRANS: Transfer of mental information (eg. tell)
9) MBUILD: Construction of new info from old (e.g decide)
10) SPEAK: Action of producing sound (e.g say)
11) ATTEND: focussing a sense organ towards a stimulus (e.g listen)

4) **Scripts** :- frames represented a general knowledge representation structure which can accomodate all kinds of knowledge. Scripts, on the other hand, help exclusively in representing stereotype events that takes place in day to day activities.

Scripts tell people what happen in a given situation, what events follow and what role every actor plays.

some such events are :-

→ Going to hotel, eating something, paying the bill & exiting.

→ Going to theatre, getting a ticket, viewing the film and leaving.

The script consists of following important components :-

- **Entry Condition** :- Basic conditions that must be fulfilled to invoke a script.
- **Result** :- Presents the situations, which describes what happens after the scripts has occured.
- **Props or things** :- These indicate the objects that are existing in the script. In a restaurant, one has tables, chairs, menu, food, money etc.
- **Roles** :- What various characters play is brought under the slot of roles. Some character explicitly involve and some are implicitly.

Waiter & cashier → explicitly, cook and owner → implicitly.

- **TRACK** :- Represent a specific instance of a generic pattern. Restaurant is specific instance of a hotel.
- **Scenes** : The script is broken into a sequence of scenes. sequence of activities are described in details :.

Script : Going to a Restaurant Props : Food Tables money menu Roles : owner Customer waiter Cashier Cook	Scene 1 : Entering the Restaurant customer enters the restaurant scans the table Choose the best one Decide to sit here Goes there occupies the seat
	Scene 2 : Ordering the food. customer asks for menu waiter brings ~~the~~ it customer glances it. choose what to eat orders that item.
Entry Condition : Customer is hungry. Customer has money owner has food	Scene 3 : eating the food. waiter brings the food Customer eats it.
Results : Customer is not hungry owner has more money Cutomer has less money owner has less food.	Secene 4. Paying the bill customer asks for the bill waiter brings it customer pays for it waiter hands the cash to the cashier waiter brings the balance amount Customer tips him Customer moves out the restaurant.

figure :- Pseudo-form of a restaurant script.

Unit :- Subject :- Intelligent System

LISP :- LISP (List Processing) is an AI programming language developed by John McCarthy in late 1950's. LISP is a symbolic processing language that represents information in lists and manipulates these lists to derive information.

LISP language is used for manipulating lists. The major characteristics is that the basic elements are treated as symbols irrespective of whether they are numeric or alphanumeric.

The basic data elements of LISP is an atom, a single word or number that stands for some object or value. Being the basic data element in LISP, an atom is indivicible. LISP has two basic types of atoms, numbers and symbols.

Numbers represent numerical values. Any type of number such as positive or negative integers, floating point or decimals are accepted.

Symbols represent alphanumeric characters. Symbols can be combination of alphabets and numerals.

A collection of numbers or symbols constitute a list. Some examples of a list are:

(APPLE ORANGE, GRAPES, MANGO)

(78 65, 71, 70. 68)

Note that the entire list is enclosed within a set of parentheses.

Nested or Multiple or Complex Lists :-

(Block - CART (HERCULES ATLAS HERO) (TVS 50 KELVINATOR MOFA))

Such a combination of lists within lists are called "nested" or "multiple" or "complex" list. Only one thing to be borne in mind is that number of parentheses, opening and closing must be same. If not, system will flash an error message.

LISP FUNCTIONS :-

A LISP program is a collection of small routines which define simple functions. In order that complex functions may be written, the language comes with a set of basic functions called primitives.

The basic primitives of LISP are classified as

* Arithmetic primitives
* Boolean primitives
* List manipulation primitives.

* **Arithmetic primitives :-** The arithmetic primitives correspond to basic arithmetic operation of addition, subtraction, multiplication and division. Since LISP uses prefix representation, all arithmetic operators are carried out on data represented in prefix form. To add two numbers, say 20 and 20, the prefix notation is +20 20.

Example :- If one has to perform the following addition

25+35+45+55+65+75

the command is

(+25(+35(+45(+55(+65 75)))))

other arithmetic primitives are

1. Difference or - sign
2. Times or * sign.
3. Quotient or / sign

* Boolean Primitives :- These primitives provides a result which is Boolean in nature i.e TRUE or FALSE.

1) ATOM : The purpose is to find out whether element is atom or not

Examples (ATOM RAMAN)

T

(ATOM (26 35 42 86))

NIL

2) NUMBERP : Determine if the atom is number or not

Examples (NUMBERP 20)

T

(NUMBERP RAMAN)

NIL.

3) LISTP 6) EVENP 9) LESSERP
4) ZEROP 7) EQUAL
5) ODDP 8) GREARP

* LIST MANIPULATION PRIMITIVES :- The purpose of the list manipulation primitives are for

1) creating a new list
2) modifying an existing list with addition, deletion or replacement of an atom.
3) Extracting portions of a list.

In LISP, value are assigned to variables by SETQ primitives. The primitive has two arguments:
→ variable
→ value to be assigned to the variable.

The value could be an atom or a list itself.

Examples :-

* (SETQ A 22) When evaluated assigns 22 to variable A.
* (SETQ TV 'ONIDA) would assign TV = ONIDA when evaluated.
* (SETQ CARS '(AMBASSADOR MARUTI FIAT)) would assign the list (AMBASSADOR MARUTI FIAT) to the CARS.

In examples, an apostrophe symbol (') . This single quote exactly differentiates between data and procedures.

The single quote is not needed in example 1 because the data is a numeric one.

LIST DEFINITION :- To define a list of fruits we can adopt any of the methods

(SETQ FRUITS '(APPLE ORANGE GRAPES) or

(LIST 'APPLE 'ORANGE 'GRAPES) or

(SETQ FRUITS2 '(ORANGE)

(LIST 'APPLE FRUIT2 'GRAPES)

LIST CONSTRUCTION :→ Lists are constructed using CONS primitive. CONS primitive adds a new element to the front of the existing list.

example :-

1) (CONS 'P '(Q R S))

(P Q R S)

2) (CONS 'RAM NIL)

(RAM)

While the CONS primitive adds a new list to the front of the existing list, the primitive APPEND adds it to the tail of existing list. Here also, the lists are identified by single quotes.

Example:

(APPEND 'RAM '(RAKESH SHAM MOHAN))

(RAKESH SHAM MOHAN RAM)

⇒ EXTRACTING PORTIONS OF LIST :- Extracting portions of a list pertains to decomposing the list and getting an atom out of it. For this, LISP provides two major primitives. They are CAR and CDR.

The CAR primitives returns the first element of the list.

(CAR '((32,36) (56,54) (67 31) 67 87 89))

(32.36.)

The CDR primitive returns the list excluding the first element.

(CDR '((32 36) (56 54) (67 31) 67 87 89))

((32 36) (56 54) (67 31) 67 87 89))