دى مراجعة شاملة كل حاجة .... هكتب اللى اتقال فى المحاضرة بالنص

و يتبقى عليكوا اختيار اللى يناسبكوا تذاكروا ايه ....

1-Definition of : Expert, Knowledge acquisition, Inference engine

2-Complete (و هتلاقوا فيه كل التعريفات اصلا)

3- Component Of expert , Main Component

xDالرسومات يعنى من الاخر

4- Agenda

5-OAV

6-Frames (Car Example)

7-Decision tree Examples

8-Semantic net (شوفوا البوست بتاع تفاصيل الامتحان)

(Travelling Sales Man, Computer types example)

9-Conflict Resolution

(Traffic lights example, Salience, MEA)

10- Clips (deffact, deftemplate with programs tb3n)

مرة كمان .... ده مجرد ملخص ل80% من المنهج مش كل المنهج

1. Complete :

.....Expertise.........is the implicit knowledge and skills of the expert that must be extracted and made explicit so that it can be encoded in an expert system.

Knowledge acquisition facility
-------------------------- automatic way for the user to enter knowledge in the system bypassing the explicit coding by knowledge engineer.

Expert system
-------------------has the capability to explain how the decision was made.

Shell
-----------------------a piece of software which contains "The user interface" , A format for declarative knowledge in the knowledge base and An inference engine

Domain
----------------The area of knowledge addressed by the expert system

Knowledge acquisition facility
---------------------------------Provides a convenient and efficient means of capturing and storing all components of the knowledge base

2. _____ is the study of making valid inferences.
   a. Answer: Logic

3. _____ refers to giving meaning to symbols.
   a. Answer: Semantics

4. Using one's experience to solve a problem is referred to as ____ .
   a. Answer: heuristics

5. Epistemology is the study of _____.
   a. Answer: knowledge

6. A mile is 5280 feet is an example of _____ knowledge.
   a. Answer: a priori

7. "Don't stand in the middle of a busy street" is an example of _____ knowledge.
   a. Answer: declarative

8. When expert systems reach conclusions, we use the term -------, but when humans reach conclusions, we use the term ---------
   a. Answer: inferencing, reasoning

9. _____ is knowledge about knowledge and expertise.
   a. Answer: Metaknowledge

10. Syntax refers to _____ , while semantics refers to _____
    a. Answer: form, meaning

11. A set of terminals is called a(n) _____ of the language.
    a. Answer: string

12. A compiler creates a(n) _____ tree when it tries to determine if statements in a program conform to valid syntax rules.
    a. Answer: parse

13. Compilers translate source code into units of smallest meaning, called _____.
    a. Answer: tokens

14. The structure of a semantic set is shown graphically in terms of _____ and _____ connecting them.
    a. Answer: nodes, arcs

15. Two commonly used links are _____ and _____.
    a. Answer: IS-A, A-KIND-OF

16. The expert system MYCIN used a type of knowledge representation called a(n) _____ triplet.
    a. Answer: object-attribute-value

17. In PROLOG, the symbol _____ means IF.
    a. Answer: –

18. A PROLOG interpreter will try to satisfy subgoals by conducting a(n) _____ search.
    a. Answer: depth-first

19. XML stands for _____ _____ _____.
    a. Answer: Extensible markup language

20. A(n) _____ is a group of slots and fillers that defines a stereotypical object.
    a. Answer: frame

21. An object that has all the typical characteristics is called a(n) _____.
    a. Answer: prototype

22. In a syllogism, the _____ provide the evidence from which the ____ must necessarily follow.
    a. Answer: premises, conclusion

23. The set under discussion is called the ____ set.
    a. Answer: universal

24. An assertion that is always true is called a(n) _____.
    a. Answer: tautology

25. Under the Aristotelian view a subject must have ______.
    a. Answer: existence

26. In a tree, knowledge is stored in the _____.
    a. Answer: nodes

27. In a binary tree, there is a maximum of ______ children per node.
    a. Answer: 2

28. A directed acyclic graph is called a(n) ______.
    a. Answer: lattice

29. A state space is a set of states showing the ______ between states that the object can experience.
    a. Answer: transitions

30. FMS stands for ______ ______ ______.
    a. Answer: finite state machine

31. A well-formed problem is ______ if when an operator is applied to a state, we are sure of the next state.
    a. Answer: deterministic

32. In an AND-OR tree, an AND-node can be satisfied only if ______ of its subgoals are satisfied.
    Answer: all

33. Induction is a type of inference from the _____ case to the ______.
    a. Answer: specific / general
34. Heuristics is inference based on ______.
    a. Answer: experience

35. Reasoning back from a true conclusion to all the premises that may have caused the conclusion is called ______.
    a. Answer: abduction

36. ______ is logical reasoning where conclusions must follow from their premises.
    a. Answer: Deductive

37. In the statement, "All collies are dogs", the predicate is _____, and the subject is ______.
    a. Answer: dogs / collies

38. Universal affirmation is a categorical ______ - form statement.
    a. Answer: A

39. Some S is not P – is called ______ negation.
    a. Answer: particular

40. According to the general rules for drawing categorical syllogisms, if a class is empty it is _____.
    a. Answer: shaded.

41. 18. If an argument is invalid, it is (true / false) that the conclusion is always incorrect.
    a. Answer: false

42. 19. Logic systems rely on _____ to prove theorems.
    a. Answer: axioms or postulates

43. CLIPS is a multiparadigm program language that provides support for _____, _____ , and ____.
    a. Answer: rule-based / object-oriented / procedural programming

44. Commands in CLIPS are surrounded by _____.
    a. Answer: matching parentheses

45. When CLIPS reads characters from the keyboard (or files), it groups them into _____.
    a. Answer: tokens

46. A group of tokens is known as a(n) _____.
    a. Answer: field

47. There are ___ types of field used by CLIPS.
    a. Answer: 8

48. The numeric fields consist of types _____ and _____.
    a. Answer: float / integer

49. A(n) _____ data type must begin and end with double quotation marks.
    a. Answer: string

50. The command to leave CLIPS is _____.
    a. Answer: (exit)

51. Facts consist of a(n) _____ name, followed by zero or more _____.
    a. Answer: relation / slots

52. A group of facts sharing the same relation name and certain common information can best be described using the _____ construct.
    a. Answer: deftemplate

53. Slots specified with the _____ keyword in their corresponding deftemplates are allowed to contain zero or more values.
    a. Answer: multislot

54. Whenever CLIPS encounters an ordered fact, it automatically creates a(n) _____ deftemplate.
    a. Answer: implied

55. To add a fact, one uses the _____ command.
    a. Answer: assert

56. To display the facts in a fact list, one uses the _____ command.
    a. Answer: facts

57. To remove a fact from the fact list, one uses the _____ command.
    a. Answer: retract

58. The _____ command is most useful during debugging.
    a. Answer: watch

59. A rule starts with the keyword _____.
    a. Answer: defrule

60. A list of rules can be displayed with the _____ command.
    a. Answer: agenda

61. The term given for rules not firing more than once for a specific set of facts is _____.
    a. Answer: refraction

62. The _____ command allows execution to be halted before any rule from a specified group of rules is fired.
    a. Answer: set-break

63. Variables are preceded by the _____ symbol.
    a. Answer: ?

64. A single-field wildcard is used to _____.
    a. Answer: test for the existence of a field slot w/o assigning a value to it

2. Nzry :

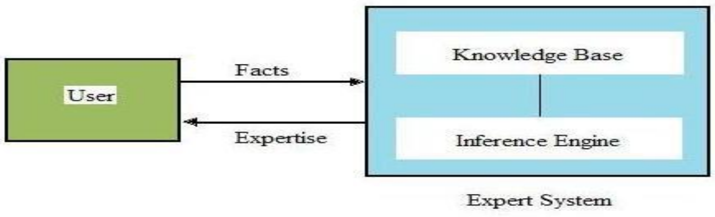
- Expert System is an artificial intelligence program that has expert-level knowledge about a particular domain and knows how to use its knowledge to respond properly

- Applications of ES:

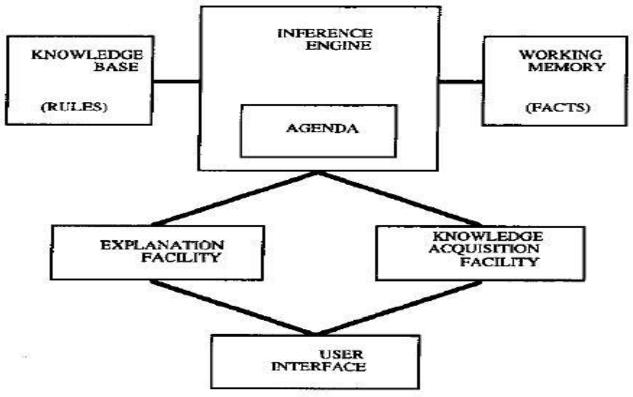
- **Diagnose human illnesses**
- **Make financial forecasts**
- **Schedule routes**
- **Delivery vehicles**

Main Components of ES:

- **Knowledge base – contains the domain knowledge which is used by the inference engine to draw conclusions it is obtained from books, magazines, knowledgeable persons, etc.**
- **Inference engine – is the generic control mechanism that applies the axiomatic knowledge to the task-specific data to arrive at some conclusion. It draws conclusions from the knowledge base**



Elements of an Expert System(main components of ES from CLIPS point of view):



- **Knowledge acquisition facility – automatic way for the user to enter knowledge in the system bypassing the explicit coding by knowledge engineer.**
- **Working memory – global database of facts used by rules.**
- **Inference engine – makes inferences deciding which rules are satisfied and prioritizing.**
- **Exploration facility – explains reasoning of expert system to user.**
- **Agenda – a prioritized list of rules created by the inference engine, whose patterns are satisfied by facts or objects in working memory.**
- **User interface – mechanism by which user and system communicate**

Advantage	Description
Increased availability	Expertise is available on any suitable computer hardware and is the manifestation of mass produced expertise.
Reduced cost	The cost of providing expertise is usually lower.
Reduced danger	Expertise systems can function in environments potentially hazardous to humans.
Performance	Expertise is permanent – the system's knowledge will last indefinitely.
Multiple expertise	Multiple experts can work simultaneously and continuously. The combined efforts may exceed those of the individuals systems working separately.
Increased reliability	Confidence levels may be increased by providing a second opinion to that of a human or in tie-breaking situations.
Explanation	Expert systems can readily verify and support conclusions, thus increasing confidence.
Fast response	Real-time response is readily available and is a function of the software/hardware used – far surpassing the response of a human.
Steady, unemotional, and complete responses at all times	This may compensate in stressful or fatigue situations which may affect a human expert.
Intelligent tutor	Students may interact with the system which serves as a tutor – affords practice / scenarios along with explanations
Intelligent database	Access of databases in an intelligent manner – e.g., data mining

ES Disadvantages:

- **Limited knowledge**
  - "shallow" knowledge
    - No "deep" understanding of the concepts and their relationships
  - No "common-sense" knowledge
  - No knowledge from possibly relevant related domains
  - "closed world"
    - The ES knows only what it has been explicitly "told"
    - It doesn't know what it doesn't know
- **Mechanical reasoning**
  - May not have or select the most appropriate method for a particular problem
  - Some "easy" problems are computationally very expensive
- **Lack of trust**
  - Users may not want to leave critical decisions to machines

Methods of Inference

1. Goal-Driven Reasoning

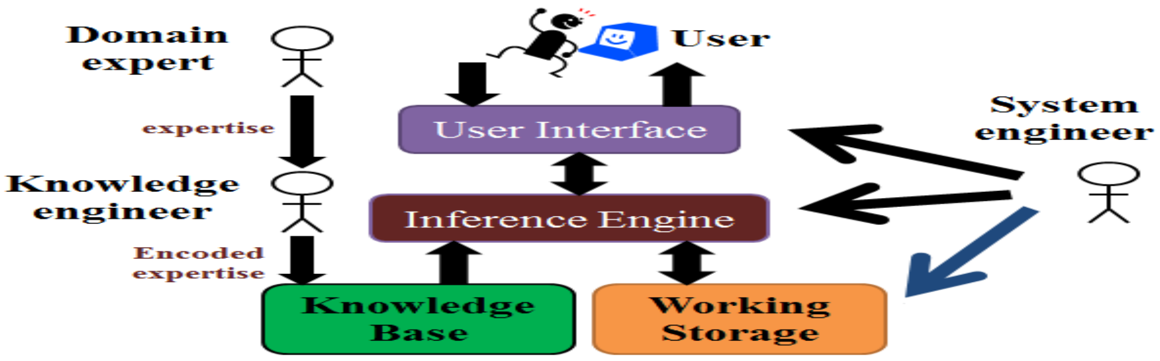
- **Aim: to pick the best choice from many enumerated possibilities**
- **The knowledge is structured in rules**
- **The rule breaks the problem into sub-problems**

2. Data Driven Reasoning

- **Problem of enumerating all of the possible answers before hand**
- **Example - configuration problems**
- **Keeps track of the current state, looks for rules moving the state closer to a final solution**

ES Structure / Architecture

Components of an Expert System



Conflict Resolution:

Priority (salience):

- ☐ in some systems, rules can have a priority (salience) attached, indicating how likely the rule is to be a good solution
- ☐ fire rule with highest priority
- ☐ priority should be used to:
  - ■ emphasise the importance of a rule
  - ■ delay firing of a non-promising rule
  - ■ NOT to establish the order in which all the rules should fire

agenda

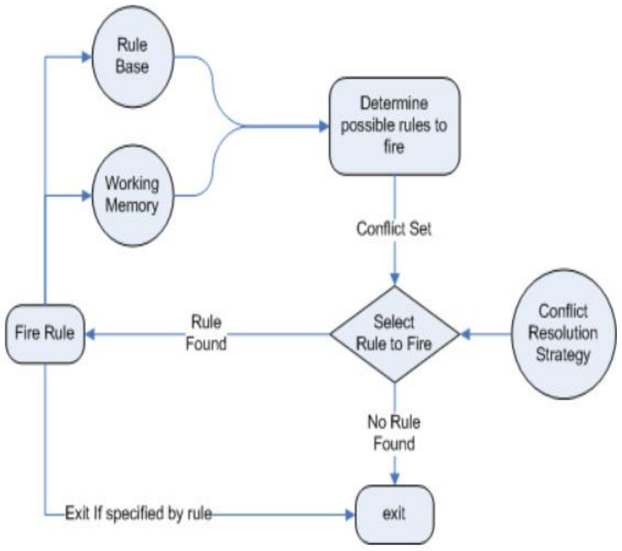
- ☐ a prioritized list of rules created by the inference engine(and have not yet been executed). The agenda acts similar to a stack (the top rule on the agenda is the first one to be executed)
- ☐ When a rule is newly activated, itís placement on the agenda

MEA

- ❑ Activations for each saliency are sorted based on FactIndex of the FIRST CE, then by specificit

Example of conflict

```
CLIPS> (defrule red-light (light red)
=>
(printout t "Stop" crlf))
CLIPS> (defrule yellow-light (light yellow)
=>
(printout t "reduce speed" crlf))
CLIPS> (defrule green-light (light red)
=>
(printout t "Go" crlf))
CLIPS> (assert(light red))
<Fact-1>
CLIPS> (agenda)
0      red-light: f-1
0      green-light: f-1
For a total of 2 activations.
CLIPS> (run)
Stop
Go
```

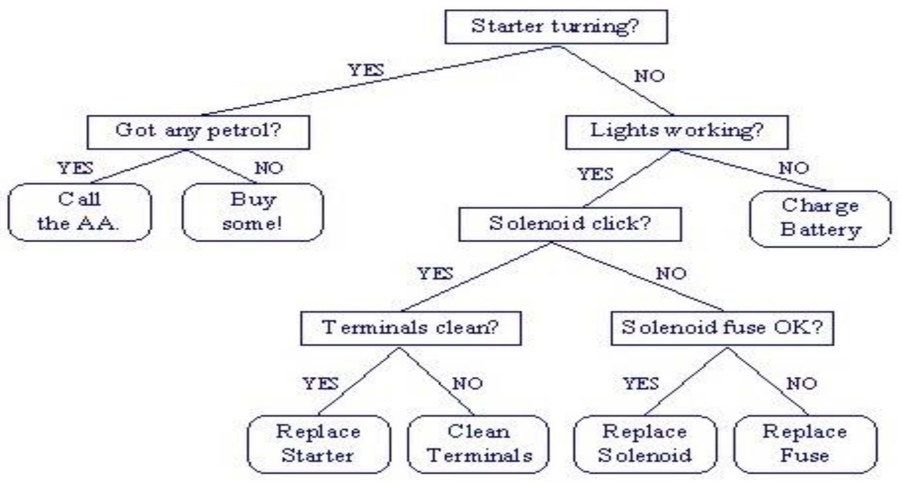


هتلاقوا فى محاضرة افتحوها اقروها بردو احتياطى

Tree:

- ❖ A tree is a hierarchical data structure consisting of:
  - Nodes – store information
  - Branches – connect the nodes
- ❖ The top node is the root, occupying the highest hierarchy.
- ❖ The leaves are at the bottom, occupying the lowest hierarchy.
- ❖ Every node, except the root, has exactly one parent.
- ❖ Every node may have zero or more child nodes.
- ❖ A binary tree restricts the number of children per node to a maximum of two
- ❖ Degenerate trees have only a single pathway from root to its one leaf.
- ❖ Trees are special case of a general mathematical structure called (Graph)

Example :



Answer:

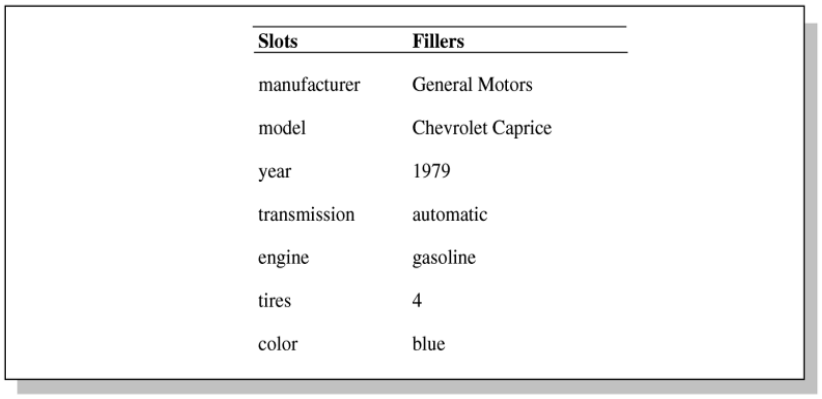
```
(defrule init
 ?x <- (initial-fact) =>
 (retract ?x)
 (assert (start)))

 (defrule start_turner
 (start)
 =>
 (printout t "starter turner" crlf)
  (assert (starter turner =(read))))

  (defrule petrol
  (starter turner yes)
  =>
  (printout t "got any petrol" crlf)
   (assert (petrol =(read))))
 (defrule petrol-yes
  (starter turner yes)
  (petrol yes)
  =>
  (printout t "Call the AA" crlf))

(defrule petrol-no
  (starter turner yes)
  (petrol no)
  =>
  (printout t "Buy some" crlf))

  (defrule lights
  (starter turner no)
  =>
  (printout t "lights working?" crlf)
   (assert (lights =(read))))

(defrule lights-no
  (starter turner no)
  (lights no)
  =>
  (printout t "charge battery" crlf))

 (defrule lights-yes
  (starter turner no)
  (lights yes)
  =>
  (printout t "solinoid click?" crlf))
   (assert (solinoid =(read))))

(defrule solinoid-yes
  (starter turner no)
  (lights yes)
  (solinoid yes)
  =>
  (printout t "terminals clean?" crlf)
  (assert (terminal =(read))))

(defrule solinoid-no
  (starter turner no)
  (lights yes)
  (solinoid no)
  =>
  (printout t "solinoid fuse ok??" crlf)
  (assert (fuse =(read))))
```

```
(defrule fuse-no
  (starter turner no)
  (lights yes)
  (solinoid no)
  (fuse no)
  =>
  (printout t "replace fuse" crlf)
  )

(defrule fuse-yes
  (starter turner no)
  (lights yes)
  (solinoid no)
  (fuse yes)
  =>
  (printout t "replace solinoid" crlf)
  )

(defrule terminal-yes
  (starter turner no)
  (lights yes)
  (solinoid yes)
  (terminal yes)
  =>
  (printout t "Replace starter" crlf)
  )

(defrule terminal-no
  (starter turner no)
  (lights yes)
  (solinoid yes)
  (terminal no)
  =>
  (printout t "clean terminals" crlf)
  )
```

Frames :

- **One type of schema is a frame (or script – time-ordered sequence of frames).**
- **Frames are useful for simulating commonsense knowledge.**
- **Semantic nets provide 2-dimensional knowledge; frames provide 3-dimensional.**
- **Frames represent related knowledge about narrow subjects having much default knowledge.**
- **A frame is a group of slots and fillers that defines a stereotypical object that is used to represent generic / specific knowledge.**
- **Commonsense knowledge is knowledge that is generally known.**
- **Prototypes are objects possessing all typical characteristics of whatever is being modeled.**

- **Problems with frames include allowing unrestrained alteration / cancellation of slots.**
- **Example: A Car Frame**



Slots	Fillers
manufacturer	General Motors
model	Chevrolet Caprice
year	1979
transmission	automatic
engine	gasoline
tires	4
color	blue

Semantic Nets

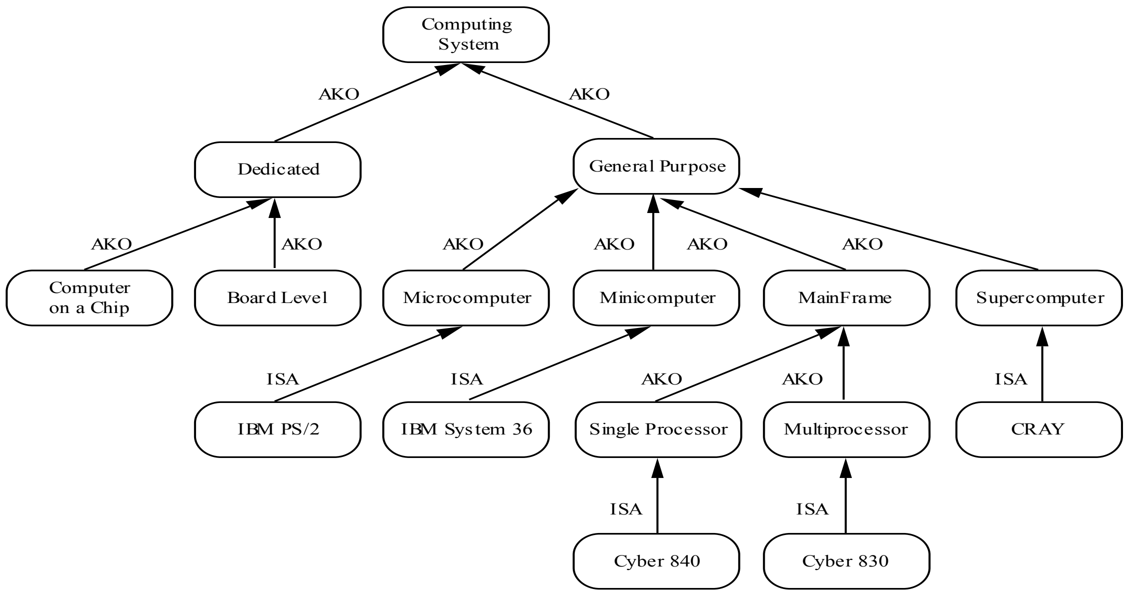
- **A classic representation technique for propositional information**
- **Propositions – a form of declarative knowledge, stating facts (true/false)**
- **Propositions are called "atoms" – cannot be further subdivided.**
- **Semantic nets consist of nodes (objects, concepts, situations) and arcs (relationships between them).**
- **Two Types of Nets: - General … - Semantic**

- **Common Types of Links:**

1. **IS-A – relates an instance or individual to a generic class**
2. **A-KIND-OF – relates generic nodes to generic nodes**

Example :

Draw a semantic net for computers using AKO and IS-A links. Consider the classes of microcomputer, minicomputer, mainframe, supercomputer, computing system, dedicated, general purpose, Board-level, computer-on-a chip, single processor, and multi processors, Include specific instances.



3. Clips : شرح للاوامر و بعد كدة شوية برامج

The general format of a deftemplate is:

```
(deftemplate <relation-name> [<optional-comment>]
<slot-definition>*)
```

The syntax description <slot-definition> is defined as:

```
(slot <slot-name>) | (multislot <slot-name>)
```

The deftemplate for the person fact:

```
(deftemplate person "An example deftemplate"
(slot name)
(slot age)
(slot eye-color)
(slot hair-color)
```

New facts can be added to the fact list using the assert command..

```
(assert <fact>+)
```

Example

```
CLIPS>
        (deftemplate person
        (slot name)
        (slot age)
        (slot eye-color)
(slot hair-color))
```

CLIPS>

```
(assert (person (name "John Q. Public")
(age 23)
(eye-color blue)
(hair-color black)))
```

More than one fact can be asserted using a single assert command. For example, the command:

```
(assert (person (name "John Q. Public")
                (age 23)
                (eye-color blue)
                (hair-color black))
        (person (name "Jane Q. Public")
                (age 36)
                (eye-color green)
                (hair-color red)))
```

The facts command can be used to display the facts in the fact list.

```
(facts)
```

For example:

```
CLIPS> (facts)
f-0 (person (name "John Q. Public")
(age 23)
(eye-color blue)
(hair-color black))
For a total of 1 fact.
```

Every fact that is inserted into the fact list is assigned a unique fact identifier starting with the letter f and followed by an integer called the fact index.

Removing facts

Removing facts from the fact list is called retraction and is done with the retract command. The syntax of the retract command is:

```
(retract <fact-index>+)
(retract 0)
```

A single retract command can be used to retract multiple facts at once.

```
(retract 0 1)
```

Modifying facts

Slot values of deftemplate facts can be modified using the modify command

```
(modify <fact-index> <slot-modifier>+)
where <s lo t -mod if ier>  is :
(<slot-name> <slot-value>)
```

For example

```
CLIPS>  modify 0 (age 24)) modify
<Fact-2>
CLIPS> (facts)
          f-2 (person (name "John Q. Public")
          (age 24)
          (eye-color blue)
          (hair-color black))
          For a total of 1 fact.
```

- A new fact index is generated for a modified fact.

Duplicating facts

```
CLIPS> (duplicate 2 (name "Jack S. Public“))
<Fact-3>
CLIPS> (facts)
          f-2 (person (name "John Q. Public")
          (age 24 )
          (eye-color blue)
          (hair-color black))
          f-3 (person (name "Jack S. Public")
          (age 24)
          (eye-color blue)
          (hair-color black))
For a total of 2 facts.
```

To enable the duplicate command, the following command may be required:

```
          (set-fact-duplication TRUE)
```

The Deffacts Construct

```
           (deffacts people "Some people we know"
          (person (name "John Q. Public") (age 24)
          (eye-color blue) (hair-color black))
          (person (name "Jack S. Public") (age 24)
          (eye-color blue) (hair-color black))
          (person (name "Jane Q. Public") (age 36)
          (eye-color green) (hair-color red)))
```

The general format of a deffacts is:
(deffacts <deffacts name> [<optional comment>] <facts>*)

- The facts in a deffacts statement are asserted using the CLIPS reset command.
- The reset command removes all facts from the fact list and then asserts the facts from existing deffacts statement. (reset)
- Even if you have not defined any deffacts statements, a reset will assert the

fact (initial-fact).

The Components of a Rule

- Rules can be typed directly into CLIPS or loaded in from a file of rules.

```
          IF the emergency is a fire
          THEN the response is to activate the sprinkler system
```

.

```
(deftemplate emergency (slot type))
```

Where the type field of the emergency fact would contain symbols such as fire,, flood,,and power outage

```
(deftemplate response (slot action))
```

Where the action field of the response fact indicates the response to be taken..

The rule expressed in CLIPS is:

```
(defrule fire-emergency “An example rule"
(emergency (type fire))
=>
     (assert (response (action activate-sprinkler-system))))
```

The general format of a rule is:

```
(defrule <rule name> [<comment>]
<patterns>* ; Left-Hand Side (LHS) of the rule
=>
<actions>* ); Right-Hand Side (RHS) of the rule
```

- The list of rules on the agenda can be displayed with the agenda command. (agenda)

The PRINTOUT Command

Besides asserting facts in the RHS of rules, the RHS can also be used to print out information using the printout command.

```
(printout <logical-name> <print-items>*)
```

Example of rule using the printout command:

```
(defrule fire-emergency
(emergency (type fire))
=>
(printout t “Activate the sprinkler system"
crlf) )
```

The logical name t tells CLIPS to send the output to the standard output device of the computer, usually the terminal.

Variables

• Variables start with a ?
– E.g. ?age
– Bindings are valid within a rule only

Bind

Associate symbols (e.g. “bill”, “<Fact-1>”, “4” etc.) to variables (e.g.“?name”)
• (bind ?percent (random 1 100))

Deffunction

Functions compute simple values
• Many built-in:

```
> (+ 5 2)
> (sin 0.2)
```

• We can define functions using **deffunction**
• Use **variables – a variable starts with “?”, e.g. ?name**
• When referring to variables inside a function, don’t forget to always include “?”

```
> (deffunction increment (?i)
(+ ?i 1)
)
> (increment 6) ...returns "7"
> (list-deffunctions)
> (undeffunction increment) ...remove the function
```

Programs:

1. Represents the below facts in Knowledge base and find out who miss the range of the salary according to employee major.

Who has more or less than regular salary?

Answer:

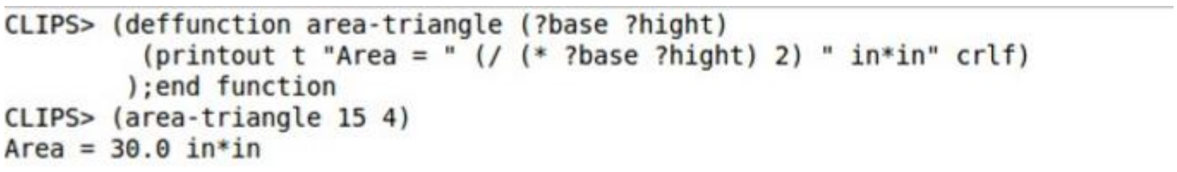
Major	Minimum salary	Maximum Salary
Computer Science	7000	20000
Biology	4000	9000
Physics	9000	23420

Name	Major	Salary
Abdullah	Computer Science	7300
Hassan	Biology	2400
Lana	Physics	50123

```
 (deffacts sal "facts about st"
(sal cs 7000 20000)
(sal phy 4000 9000)
(sal bio 9000 23420)
)

(deffacts emp "facts about st"
(emp abdallah cs 7300)
(emp hassan bio 2400)
(emp lana phy 50123)
)
(defrule rule2 (sal ?smajor ?min ?max)
(emp ?ename ?emajor ?sal)
(test(eq ?emajor ?smajor))
=>
(if(and(>= ?sal ?min)(<= ?sal ?max))
then
(printout t ?ename ?sal crlf)
else
(printout t "out of range" crlf)))
```

2. Find the area of triangle with the base of 15 inches and the height of 4 inches ? Hint: Area formula = ½ (base * height)

Answer:



```
CLIPS> (deffunction area-triangle (?base ?hight)
         (printout t "Area = " (/ (* ?base ?hight) 2) " in*in" crlf)
        );end function
CLIPS> (area-triangle 15 4)
Area = 30.0 in*in
```

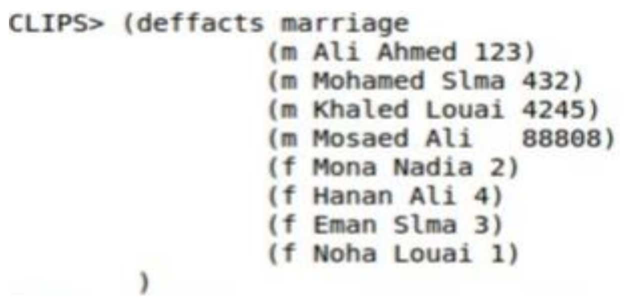
3. The following table has the relation between each father and his son let's call this table M, and the table next table F has relation between Mother and her son. If we supposed that sons in both table are same could you figure out whom married whom?

Father	Son	Marriage ID
Ali	Ahmed	123
Mohamed	Slma	432
Khaled	Louai	4245
Mosaed	Ali	88808

Mother	Son	Number of Suns
Mona	Nadia	2
Hanan	Ali	4
Eman	Slma	3
Noha	Louai	1

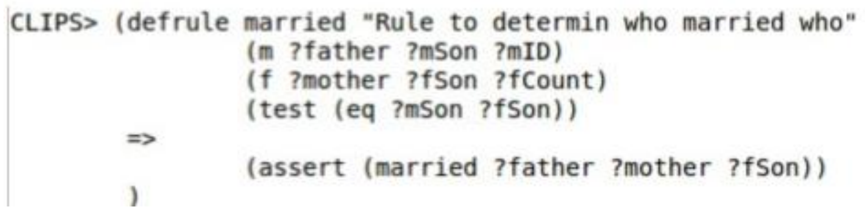
1- Represents the facts in table.
2- Show the facts using (facts) command.
3- Add Assert to add facts.

Answer :



```
CLIPS> (deffacts marriage
                (m Ali Ahmed 123)
                (m Mohamed Slma 432)
                (m Khaled Louai 4245)
                (m Mosaed Ali   88808)
                (f Mona Nadia 2)
                (f Hanan Ali 4)
                (f Eman Slma 3)
                (f Noha Louai 1)
        )
```

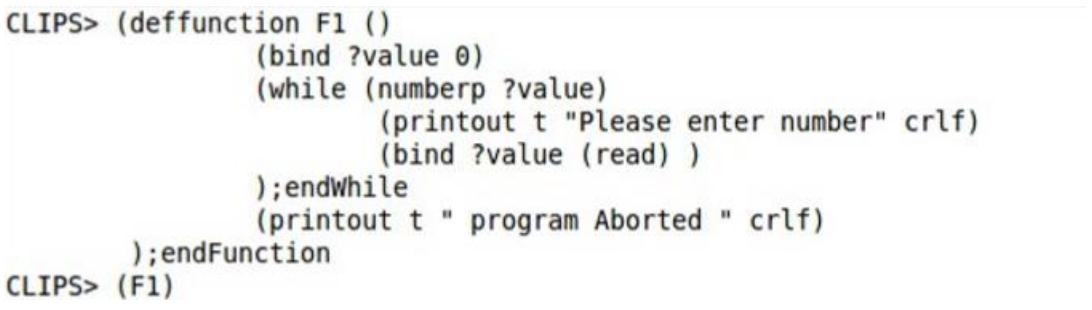
Clips> (Facts).



```
CLIPS> (defrule married "Rule to determin who married who"
                (m ?father ?mSon ?mID)
                (f ?mother ?fSon ?fCount)
                (test (eq ?mSon ?fSon))
        =>
                (assert (married ?father ?mother ?fSon))
        )
```

4. write a program that read input form the user and check wither input is a number, if it's a number then let user input again otherwise abort from the program.

Answer:

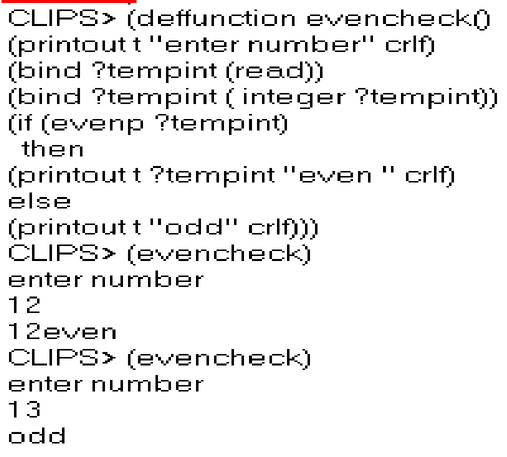


```
CLIPS> (deffunction F1 ()
                (bind ?value 0)
                (while (numberp ?value)
                        (printout t "Please enter number" crlf)
                        (bind ?value (read) )
                );endWhile
                (printout t " program Aborted " crlf)
        );endFunction
CLIPS> (F1)
```

---

5. Write a simple program that read integer form user and check wither if it's even or odd and assert facts about user input.

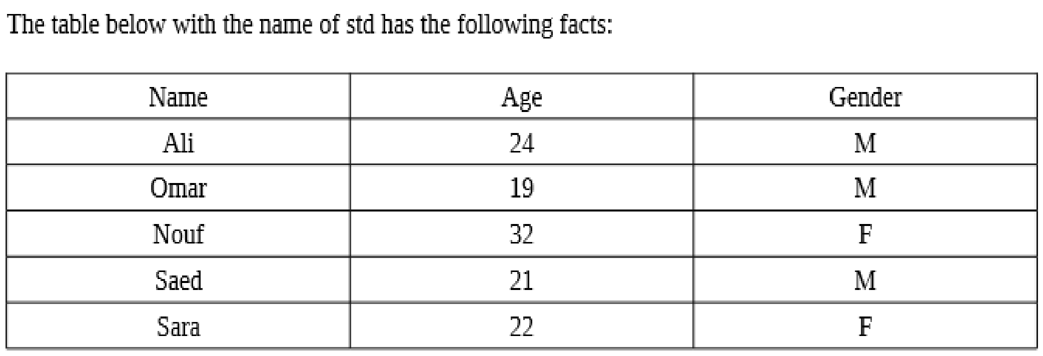
Answer:



```
CLIPS> (deffunction evencheck()
(printout t "enter number" crlf)
(bind ?tempint (read))
(bind ?tempint ( integer ?tempint))
(if (evenp ?tempint)
 then
(printout t ?tempint "even " crlf)
else
(printout t "odd" crlf)))
CLIPS> (evencheck)
enter number
12
12even
CLIPS> (evencheck)
enter number
13
odd
```

---

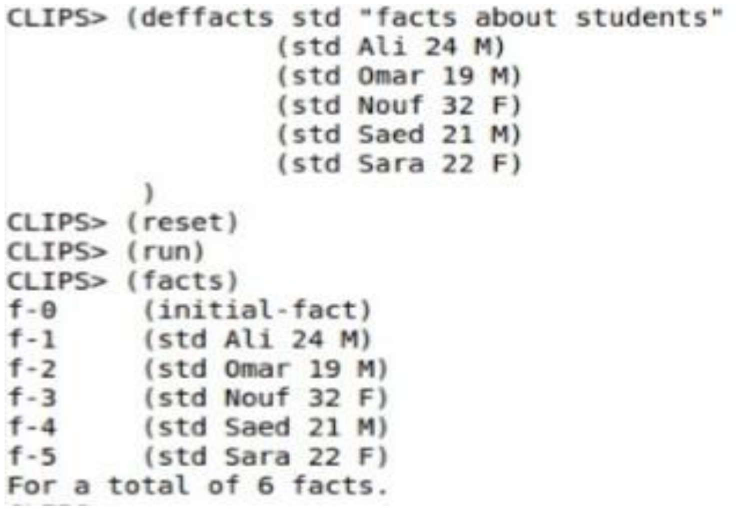
6. given this table write a program to define the following table



The table below with the name of std has the following facts:

Name	Age	Gender
Ali	24	M
Omar	19	M
Nouf	32	F
Saed	21	M
Sara	22	F

Answer:



```
CLIPS> (deffacts std "facts about students"
                 (std Ali 24 M)
                 (std Omar 19 M)
                 (std Nouf 32 F)
                 (std Saed 21 M)
                 (std Sara 22 F)
        )
CLIPS> (reset)
CLIPS> (run)
CLIPS> (facts)
f-0     (initial-fact)
f-1     (std Ali 24 M)
f-2     (std Omar 19 M)
f-3     (std Nouf 32 F)
f-4     (std Saed 21 M)
f-5     (std Sara 22 F)
For a total of 6 facts.
```

---

7. Using CLIPS define facts for a patient with symptoms of “Measles” الحصبة. A "Patient" may have values for fever, spots, rash and sore-throat.

Answer:

```
(deftemplate Patient
  (slot fever)
  (slot spots)
  (slot rash)
  (slot sore_throat))

(deftemplate Diagnosis
  (slot diagnosis))

(deftemplate Treatment
  (slot treatment))

(defrule Measles
  (declare (salience 100))
  (Patient (fever high) (spots yes))
  =>
  (assert (Diagnosis (diagnosis measles)))
  (printout t "Measles diagnosed" crlf))
(defrule Allergy1
  (and (Patient (spots yes))
      (not (Diagnosis (diagnosis measles))))
  =>
  (assert (Diagnosis (diagnosis allergy)))
  (printout t "Allergy diagnosed from spots and lack of measles" crlf))
```

```
(defrule Allergy2
  (Patient (rash yes))
  =>
  (assert (Diagnosis (diagnosis allergy)))
  (printout t "Allergy diagnosed from rash" crlf))

(defrule Flu
  (Patient (sore_throat yes) (fever mild | high))
  =>
  (assert (Diagnosis (diagnosis flu)))
  (printout t "Flu diagnosed" crlf))

(defrule Penicillin
  (Diagnosis (diagnosis measles))
  =>
  (assert (Treatment (treatment pennicillin)))
  (printout t "Penicillin prescribed" crlf))

(defrule Allergy_pills
  (Diagnosis (diagnosis allergy))
  =>
  (assert (Treatment (treatment allergy_shot)))
  (printout t "Allergy shot prescribed" crlf))
(defrule Bed_rest
  (Diagnosis (diagnosis flu))
  =>
  (assert (Treatment (treatment bed_rest)))
  (printout t "Bed rest prescribed" crlf))

(deffacts Symptoms
  (Patient (fever high)
        (spots yes)
        (rash no)
        (sore_throat no)))
```