Expert Systems

What are they

Why are they Useful

Architecture

Forward Chaining

Backward Chaining

Shells

Example

What is an Expert System?

Software that stores knowledge from a human expert on a particular area and applies that knowledge to solve problems in that domain similar to how the human expert would do it.

The term “Expert System” is also used for systems that store knowledge that is not necessarily expert – but that is structured and used in a similar way ..

The term “Knowledge-Based System” is also used

<u>Many Applications in:</u>

Business

Medicine

Manufacturing

Military

Banking

etc

Why are they Useful ?

1. They are considered a “strong” method of searching for solutions

   In sharp contrast to “weak” methods which search through many alternatives using a heuristic evaluation function applied to many of all the possible alternatives

   An important lesson learned by AI researchers –
   “Knowledge gets to the answer much more quickly than sophisticated search methods”

Example: Find a restaurant that serves “paella”

Exhautive :	Call every restaurant in town
Heuristic – weak	Call every spanish restaurant in town
Expert – strong	In Tampa, the Columbia serves paella

Why are they Useful ?

2. They can preserve valuable human expertise

Usually this expertise is expensive and difficult to obtain

A Little History

late 1950's - - General Problem Solver (GPS) Newell and Simon

Emphasized the use of powerful search methods that could be applied to a large set of problems

Use of **rules** to represent human knowledge and to reason

Based on the cognitive model of a **long term memory** where rules are stored and **short term memory** where temporary knowledge is stored while solving a specific problem

Use of a **cognitive processor** to select and activate the rules depending on specific input data

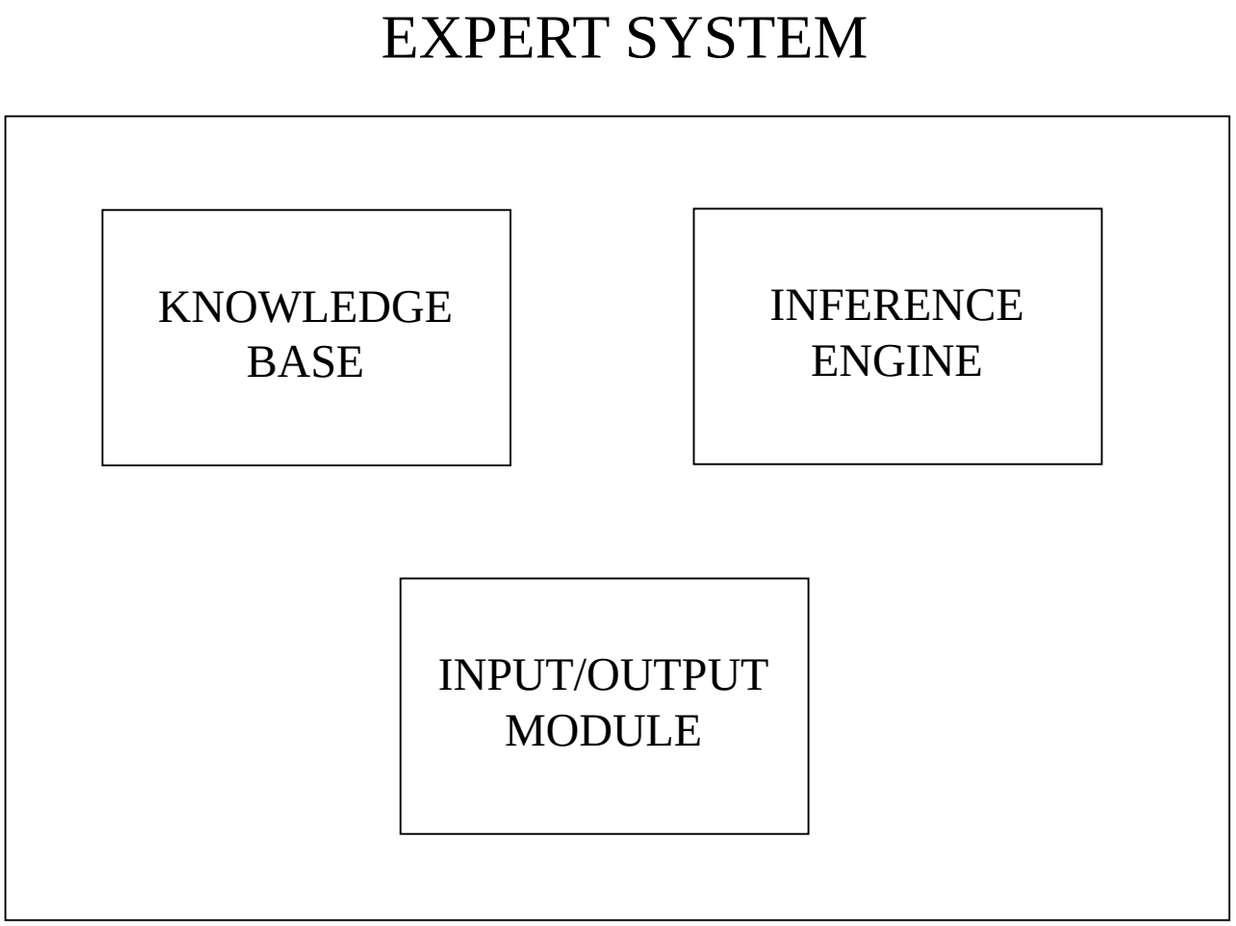
late 1960"s - - Use of Specific Domain Knowledge to Solve Problems

DENDRAL – in chemistry

MYCIN – in medicine

PROSPECTOR – for mineral exploration

Separation of Knowledge from Its Application



Compare this architecture to the traditional approach of designing and implementing an algorithm to solve a problem

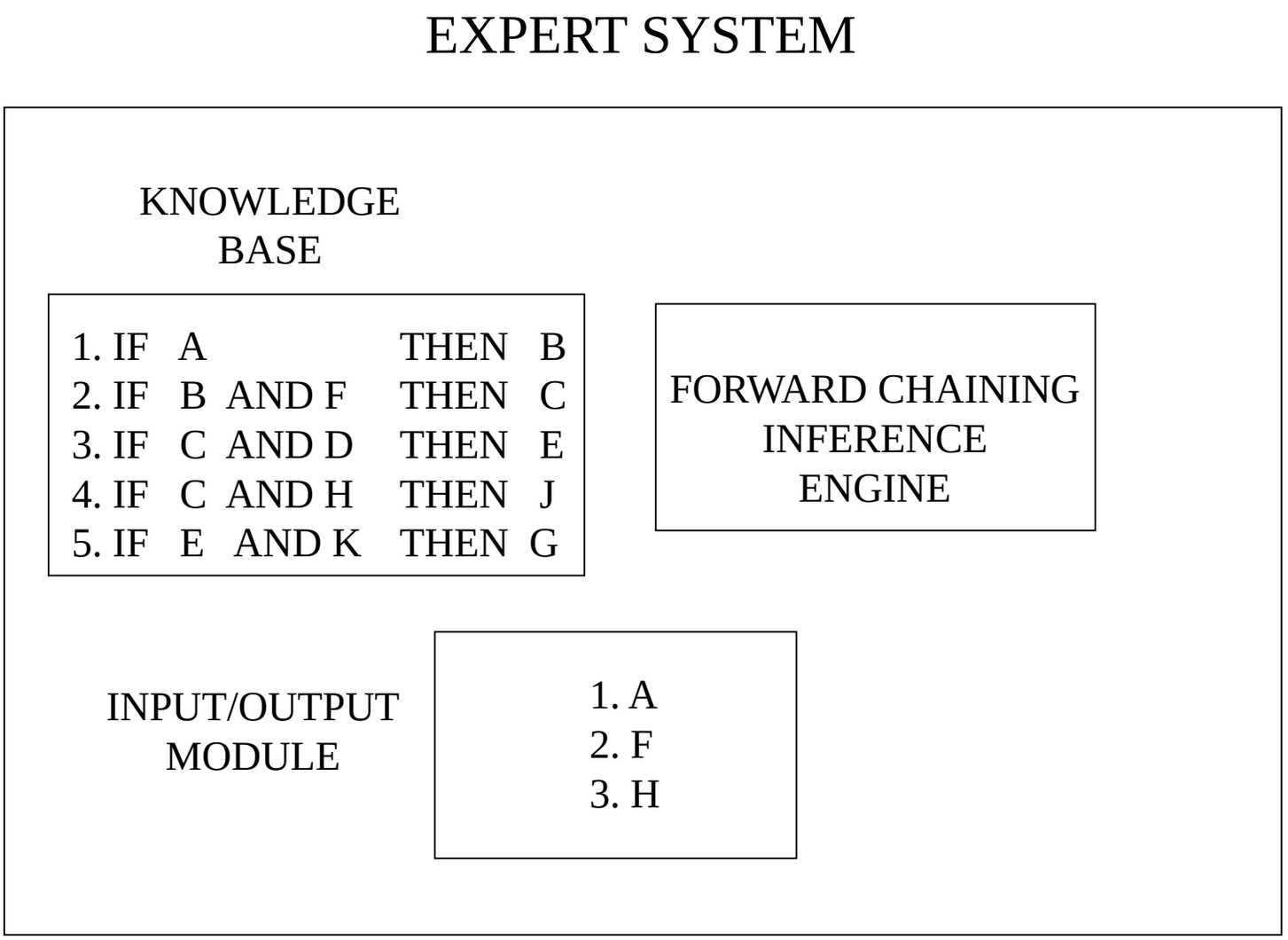
Most Common Types of Inference Engines

Used in Rule-Based Systems

FORWARD CHAINING

BACKWARD CHAINING

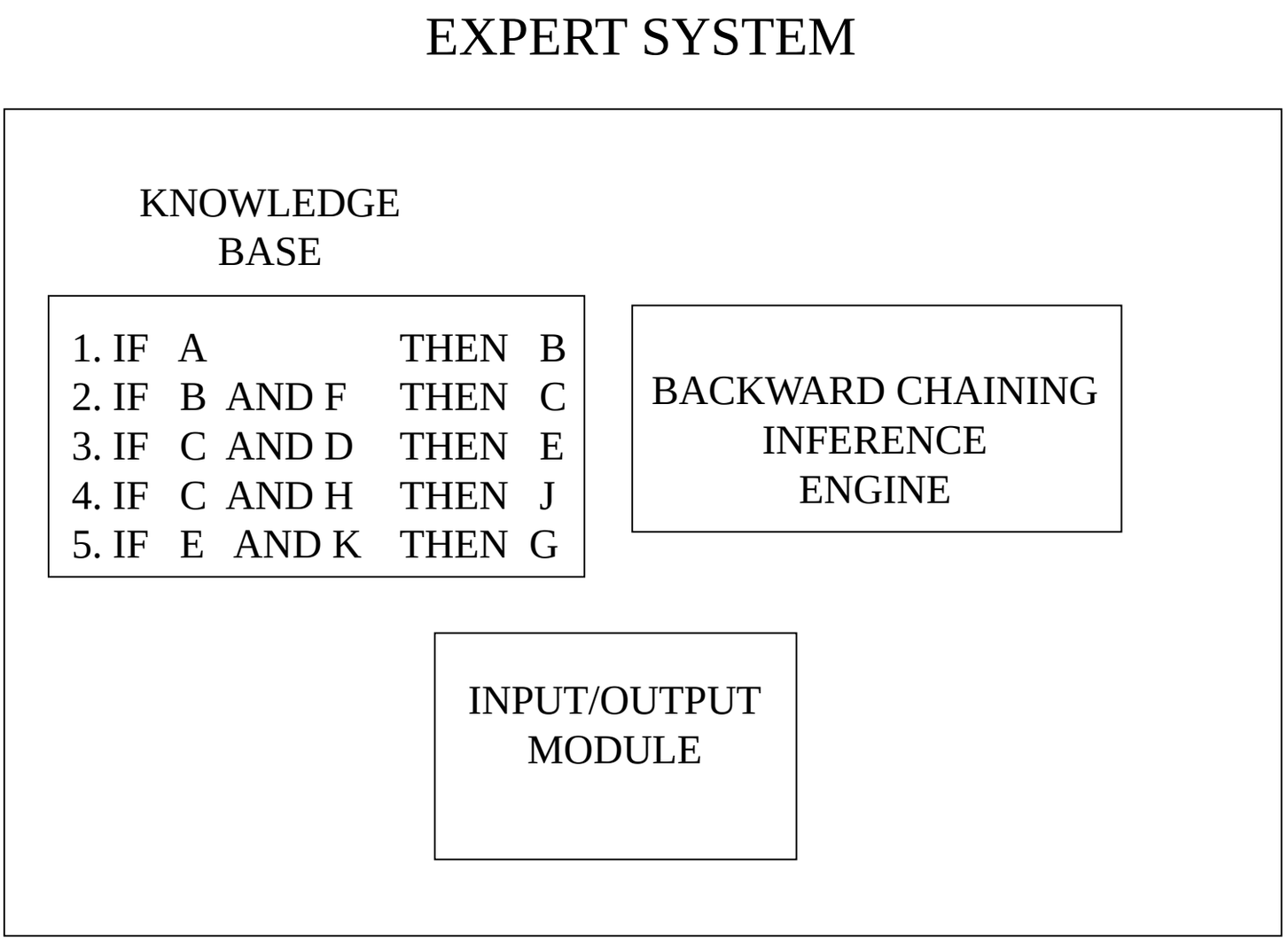
Forward Chaining



Forward Chaining Inference Engine works from known facts to conclude:

B, C, J ARE TRUE

Backward Chaining



Backward Chaining Inference Engine tries to prove that a specific conclusion (or goal) is true:

Is G true ?

It will ask (via the I/O module) if A is true. If so then it will ask if F is true. If so, then it will ask if D is true. If so it will ask if K is true.

Explanation Capabilities

Why is it advantageous to be able to explain conclusions reached by expert systems?

KNOWLEDGE BASE

1. IF	A		THEN	B
2. IF	B	AND F	THEN	C
3. IF	C	AND D	THEN	E
4. IF	C	AND H	THEN	J
5. IF	E	AND K	THEN	G

Why is G true in this particular instance?

Answer: Because E is true and K is true

Continue:

Why is E true in this particular instance?

Answer: Because C is true and D is true

Continue:

Why is C true in this particular instance?

Answer: Because B is true and F is true

Continue:

Why is B true in this particular instance?

Answer: Because A is true

What makes it possible to explain why a specific conclusion was reached?

Expert Systems Shells

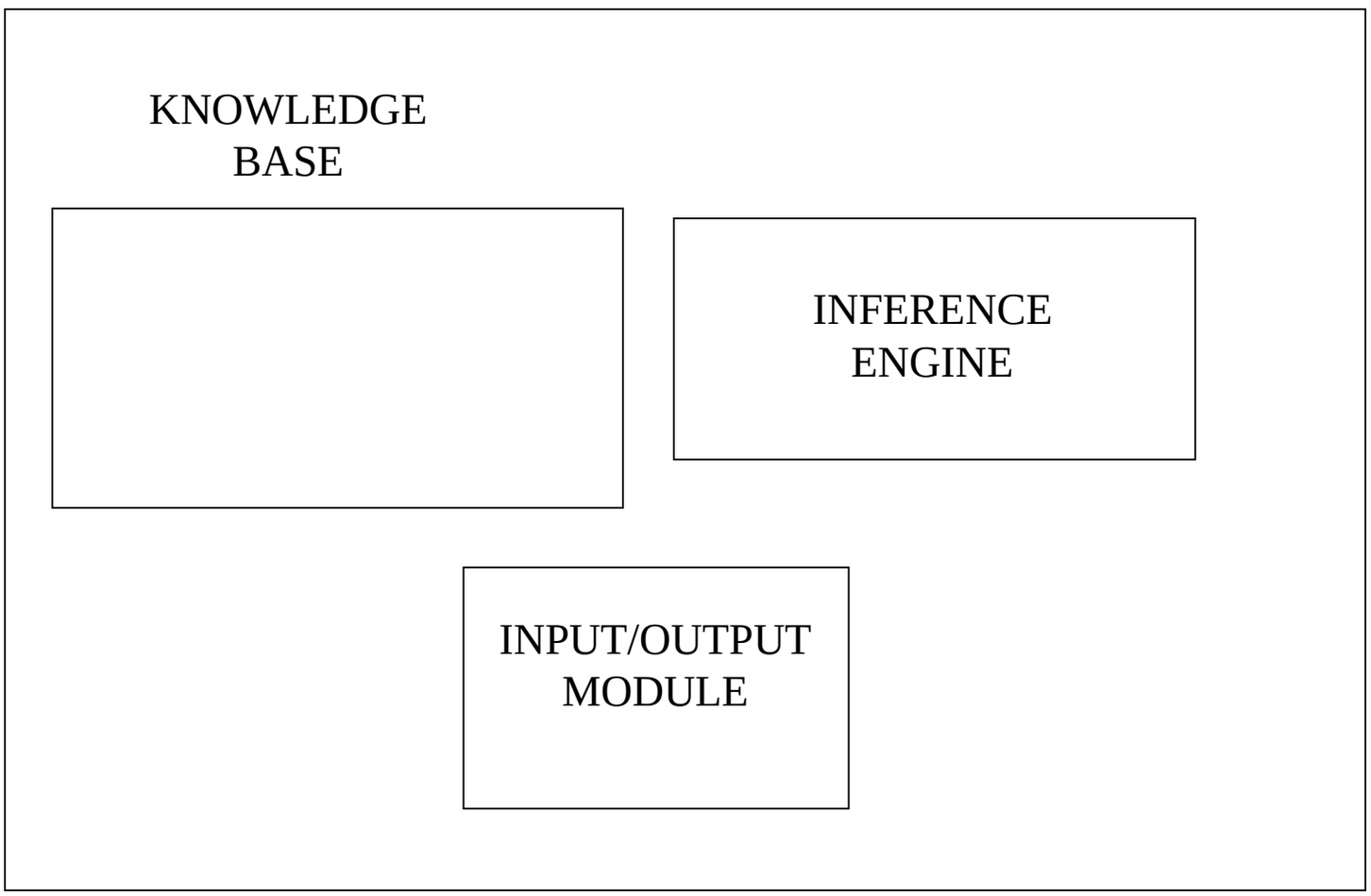
A software tool to build expert systems

Includes 1. A language to capture the knowledge

2. An inference engine to reason

3. A way to build a GUI – may include explanation feature

4. A deployment mechanism for the system



A Simple Set of Rules

if has_sauce = yes and
sauce = spicy
then wine_body = full

if has_sauce = no and
main_component = fish
then wine_body = light

if wine_body = light and
preferred_color = white
then wine = riesling

if wine_body = full and
preferred_color = red
then wine = cabernet_sauvignon

if wine_body = light and
preferred_color = red
then wine = zinfandel

if has_sauce = yes and
main_component = pasta
then wine = chianti

NOTICE:

Syntax

How Knowledge is represented

implied objects

limitations

Backward Chaining
With the Set of Rules

Work **from the goal** of finding a wine **to the data** needed

goal → data

goal = wine

if has_sauce = yes and
sauce = spicy
then wine_body = full

if has_sauce = no and
main_component = fish
then wine_body = light

if wine_body = light and
preferred_color = white
then wine = riesling

if wine_body = full and
preferred_color = red
then wine = cabernet_sauvignon

if wine_body = light and
preferred_color = red
then wine = zinfandel

if has_sauce = yes and
main_component = pasta
then wine = chianti

NOTICE:

How search is done

Need for pattern matching

When to ask user questions

When to Stop Searching

Forward Chaining
With the Set of Rules

Work **from the data** given **to the conclusions** that can be reached

data → conclusions

main_component = fish
preferred_color = white

has_sauce = no

if has_sauce = yes and
sauce = spicy
then wine_body = full

if has_sauce = no and
main_component = fish
then wine_body = light

if wine_body = light and
preferred_color = white
then wine = riesling

if wine_body = full and
preferred_color = red
then wine = cabernet_sauvignon

if wine_body = light and
preferred_color = red
then wine = zinfandel

if has_sauce = yes and
main_component = pasta
then wine = chianti

NOTICE:

How search is done

Need for pattern matching

No questions asked

When to Stop Searching

Some Observations on Forward and Backward Chaining

Backward Chaining is used when:

A goal has been established and you need to find if the data can show that the goal is true

Most problems involving “diagnosis” are backward chaining – example: a physician does not collect “all possible” data from you but only the data needed to support a diagnosis

Forward Chaining is used when:

All of the relevant data is available beforehand example: Theorem proving using resolution – you know a group of facts are true but you don’t know what can be proven from them

Contrast Between Rule-Based Expert Systems and Conventional (Algorithmic) Programs

Rule-Based System	Algorithmic Program
program consists of rules plus inference engine	program consists of statements in a specified sequence
knowledge about the problem is in the rules	knowledge about the problem is in the statements AND in which sequence the statements are placed
inference engine selects which rules to apply	programmer specifies sequence explicitly
rules can be added or deleted anywhere in the list of rules	adding or deleting statements must carefully consider their position in the sequence

Create a Set of Rules
for a simple problem domain

Select a Domain

Select an Expert

Create the rules

Test them

Create a Set of Rules
for a simple problem domain

1. If X has hair
   Then X-species = mammal

2. If X gives milk
   Then X-species = mammal

3. If X has feathers
   Then X-species = bird

4. If X flyes AND
   X lays eggs
   Then X-species = bird

5. If X-species = mammal AND
   X eats meat
   Then X-mammal-type = carnivore

6. If X-species = mammal AND
   X has pointed teeth AND
   X has claws AND
   X has forward-pointed eyes
   Then X-mammal-type = carnivore

7. If X-species = mammal AND
   X has hooves
   Then X-mammal-type = ungulate

Create a Set of Rules
for a simple problem domain

8. If X-species = mammal AND
   X chews cud
   Then X-mammal-type = ungulate

9. If X-mammal-type = carnivore AND
   X has tawny color AND
   X has dark spots
   Then X = cheetah

10. If X-mammal-type = carnivore AND
    X has tawny color AND
    X has black strips
    Then X = tiger

11. If X-mammal-type = ungulate AND
    X has long legs AND
    X has long neck AND
    X has tawny color AND
    X has dark spots
    Then X = giraffe

12. If X-mammal-type = ungulate AND
    X has white color AND
    X has black stripes
    Then X = zebra

Create a Set of Rules
for a simple problem domain

13. If X-species = bird AND
X does not fly AND
X has long legs AND
X has long neck AND
X is black and white
Then X = ostrich

14. If X-species = bird AND
X does not fly AND
X swims AND
X is black and white
Then X = penguin

15. If X-species = bird AND
X is a good flyer
Then X = albatross