Contrast $5 > x$ to $\forall x\ 5 > x$ or $\exists x\ 5 > x$

x is free — occurs outside any quantifier over x

x is bound

A formula with no free variables is called a <u>sentence</u>.

Our models will make each sentence true or false, but do not interpret formulas with free variables.

Q2] What are the models of predicate logic?

A model specifies a domain D and a relational database over D interpreting each relation symbol. The model also provides a mapping from constant symbols to D.

Also, = , if present, must be the identity relation over D : $\{(x,x) \mid x \in D\}$

examplb model

$D = \{0, 1, 2\}$

$penguin(x) \equiv \{1, 2\}$

$human(x) \equiv \{0\}$

$teacher(x,y) \equiv \{(0,1), (0,2)\}$

$Bob \longrightarrow 0$

example model

$D = \mathbb{N}$

$penguin(x) = \mathbb{N}$

$human(x) \equiv$ odd numbers

$teacher(x,y) \equiv \{\}$

$Bob \mapsto 1$

$\exists x \exists y \exists z \; x \neq y \land y \neq z \land z \neq x \land Penguin(x) \land Penguin(y) \land Penguin(z)$

$\forall x \exists y \; Teacher(x,y)$

Q3] Which formulas (sentences) are true in which models?

$\Longrightarrow$ translate to English, ask if the English is true in the model.

There are three distinct penguins.

Everyone teaches someone.

Translate Bob ~~John~~ loves every beagle.

use constant ~~John~~ Bob, binary relation loves(x,y) and monadic/unary ~~...~~ relation (symbol) Beagle(x)

$\forall x$ Beagle(x) $\wedge$ Loves(Bob, x) ✗

$\forall x$ Beagle(x) $\rightarrow$ Loves(Bob, x)

$\forall x$ Loves(Bob, x) ✗

essentially $\forall$ limited to beagles.

Bob loves some beagle.

$\exists x$ Loves (Bob, x) ✗

$\exists x$ Beagle(x) $\rightarrow$ Loves(Bob, x) ✗

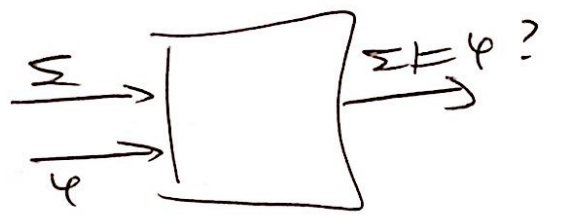
$\exists x$(Beagle(x) $\wedge$ Loves(Bob, x))

$\exists$ limited to Beagles

When can we conclude φ from premises Σ?

(in p.c.)

$\Sigma \models \varphi$ when every model of Σ (making Σ all true) is also a model of φ. (makes φ true)



(for Boolean logic this box can be implemented w/ exponential worst case runtime. Better may be possible ⇒ open?)

<u>Thm</u> Predicate logic entailment is <u>undecidable</u>.

↳ no computer program can guarantee termination with a correct answer.