Mathematical Logic

Part Two

Recap from Last Time

Recap So Far

- A ***propositional variable*** is a variable that is either true or false.
- The ***propositional connectives*** are as follows:
  - Negation: $\neg p$
  - Conjunction: $p \wedge q$
  - Disjunction: $p \vee q$
  - Implication: $p \rightarrow q$
  - Biconditional: $p \leftrightarrow q$
  - True: $\top$
  - False: $\bot$

Take out a sheet of paper!

What's the truth table for the → connective?

What's the negation of $p \to q$?

New Stuff!

First-Order Logic

What is First-Order Logic?

- ***First-order logic*** is a logical system for reasoning about properties of objects.
- Augments the logical connectives from propositional logic with
  - ***predicates*** that describe properties of objects,
  - ***functions*** that map objects to one another, and
  - ***quantifiers*** that allow us to reason about multiple objects.

Some Examples

$$Likes(You, Eggs) \wedge Likes(You, Tomato) \rightarrow Likes(You, Shakshuka)$$

$$Learns(You, History) \vee ForeverRepeats(You, History)$$

$$DrinksTooMuch(Me) \wedge IsAnIssue(That) \wedge IsOkay(Me)$$

$$Likes(You, Eggs) \wedge Likes(You, Tomato) \rightarrow Likes(You, Shakshuka)$$

$$Learns(You, History) \vee ForeverRepeats(You, History)$$

$$DrinksTooMuch(Me) \wedge IsAnIssue(That) \wedge IsOkay(Me)$$

These blue terms are called *constant symbols*. Unlike propositional variables, they refer to *objects*, not *propositions*.

$$Likes(You, Eggs) \wedge Likes(You, Tomato) \rightarrow Likes(You, Shakshuka)$$

$$Learns(You, History) \vee ForeverRepeats(You, History)$$

$$DrinksTooMuch(Me) \wedge IsAnIssue(That) \wedge IsOkay(Me)$$

The red things that look like function calls are called *predicates*. Predicates take objects as arguments and evaluate to true or false.

$$Likes(You, Eggs) \wedge Likes(You, Tomato) \rightarrow Likes(You, Shakshuka)$$

$$Learns(You, History) \vee ForeverRepeats(You, History)$$

$$DrinksTooMuch(Me) \wedge IsAnIssue(That) \wedge IsOkay(Me)$$

What remains are traditional propositional connectives. Because each predicate evaluates to true or false, we can connect the truth values of predicates using normal propositional connectives.

Reasoning about Objects

- To reason about objects, first-order logic uses ***predicates***.
- Examples:
  - *IsCute*(*Quokka*)
  - *ArgueIncessantly*(*Democrats*, *Republicans*)
- Applying a predicate to arguments produces a proposition, which is either true or false.

First-Order Sentences

- Sentences in first-order logic can be constructed from predicates applied to objects:

$$Cute(a) \rightarrow Dikdik(a) \vee Kitty(a) \vee Puppy(a)$$

$$Succeeds(You) \leftrightarrow Practices(You)$$

$$x < 8 \rightarrow x < 137$$

The less-than sign is just another predicate. Binary predicates are sometimes written in *infix notation* this way.

Numbers are not "built in" to first-order logic. They're constant symbols just like "You" and "a" above.

Equality

- First-order logic is equipped with a special predicate **=** that says whether two objects are equal to one another.
- Equality is a part of first-order logic, just as → and ¬ are.
- Examples:

$$TomMarvoloRiddle = LordVoldemort$$

$$MorningStar = EveningStar$$

- Equality can only be applied to ***objects***; to state that two ***propositions*** are equal, use ↔.

Let's see some more examples.

$$FavoriteMovieOf(You) \neq FavoriteMovieOf(Her) \wedge StarOf(FavoriteMovieOf(You)) = StarOf(FavoriteMovieOf(Her))$$

$$FavoriteMovieOf(You) \neq FavoriteMovieOf(Her) \wedge StarOf(FavoriteMovieOf(You)) = StarOf(FavoriteMovieOf(Her))$$

These purple terms are *functions*. Functions take objects as input and produce objects as output.

$$FavoriteMovieOf(You) \neq FavoriteMovieOf(Her) \wedge$$
$$StarOf(FavoriteMovieOf(You)) = StarOf(FavoriteMovieOf(Her))$$

Functions

- First-order logic allows ***functions*** that return objects associated with other objects.
- Examples:

$$ColorOf(Money)$$

$$MedianOf(x, y, z)$$

$$x + y$$

- As with predicates, functions can take in any number of arguments, but always return a single value.
- Functions evaluate to ***objects***, not ***propositions***.

Objects and Predicates

- When working in first-order logic, be careful to keep objects (actual things) and predicates (true or false) separate.
- You cannot apply connectives to objects:

  ⚠ $Venus \rightarrow TheSun$ ⚠

- You cannot apply functions to propositions:

  ⚠ $StarOf(IsRed(Sun) \wedge IsGreen(Mars))$ ⚠

- Ever get confused? *Just ask!*

One last (and major) change

"For any natural number n,
n is even iff n^2 is even"

$$\forall n.\ (n \in \mathbb{N} \rightarrow (Even(n) \leftrightarrow Even(n^2)))$$

∀ is the ***universal quantifier***
and says "for any choice of n,
the following is true."

The Universal Quantifier

- A statement of the form

  $\forall x.$ *some-formula*

  is true if, for every choice of x, the statement ***some-formula*** is true when x is plugged into it.

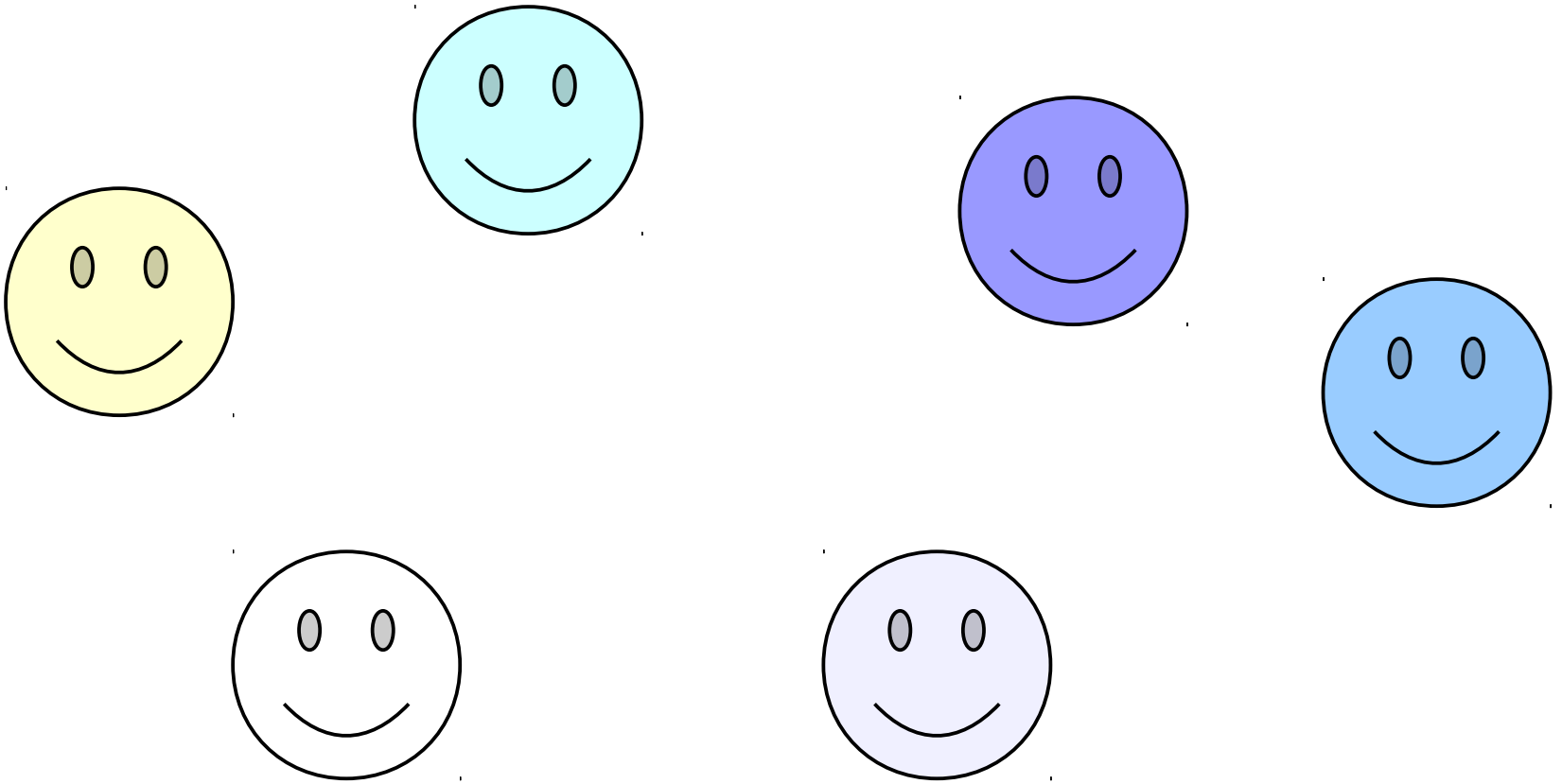
- Examples:

  $\forall p.\ (Puppy(p) \rightarrow Cute(p))$

  $\forall m.\ (IsMillennial(m) \rightarrow IsSpecial(m))$

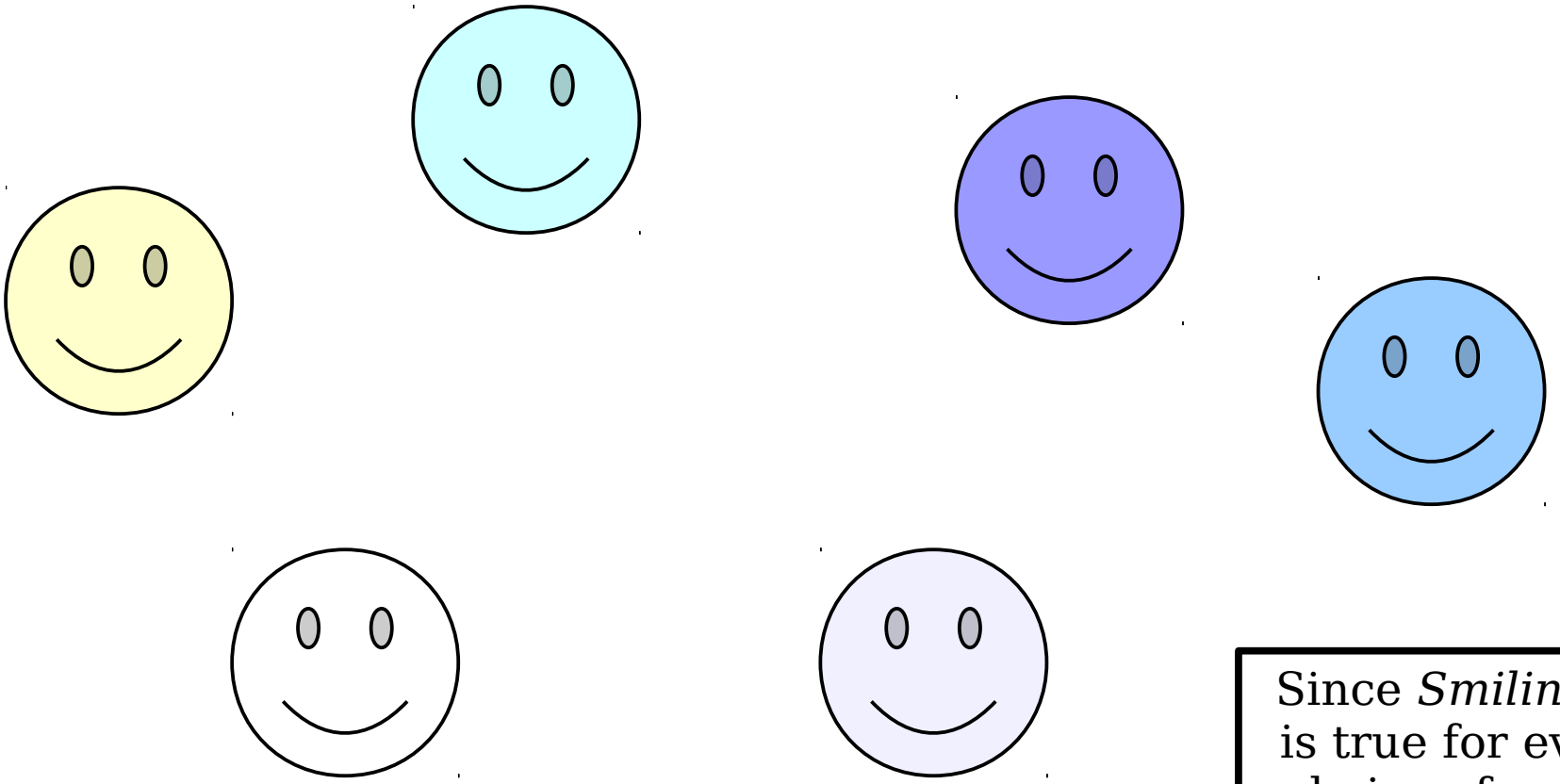
  $Tallest(SK) \rightarrow$
  $\quad \forall x.\ (SK \neq x \rightarrow ShorterThan(x, SK))$

The Universal Quantifier



$\forall x.\ \mathit{Smiling}(x)$

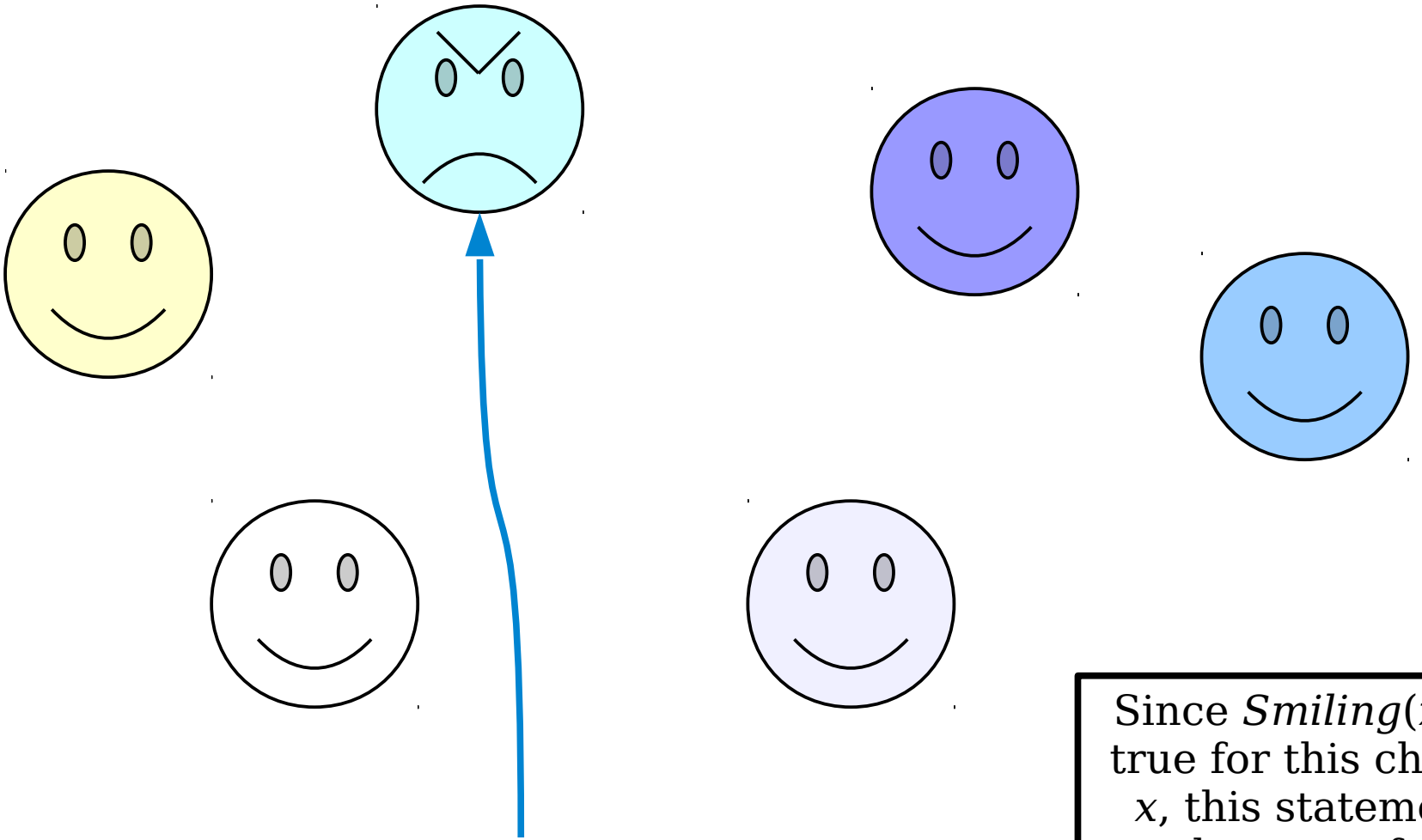
The Universal Quantifier



$\forall x.\ Smiling(x)$

Since $Smiling(x)$ is true for every choice of x, this statement evaluates to true.

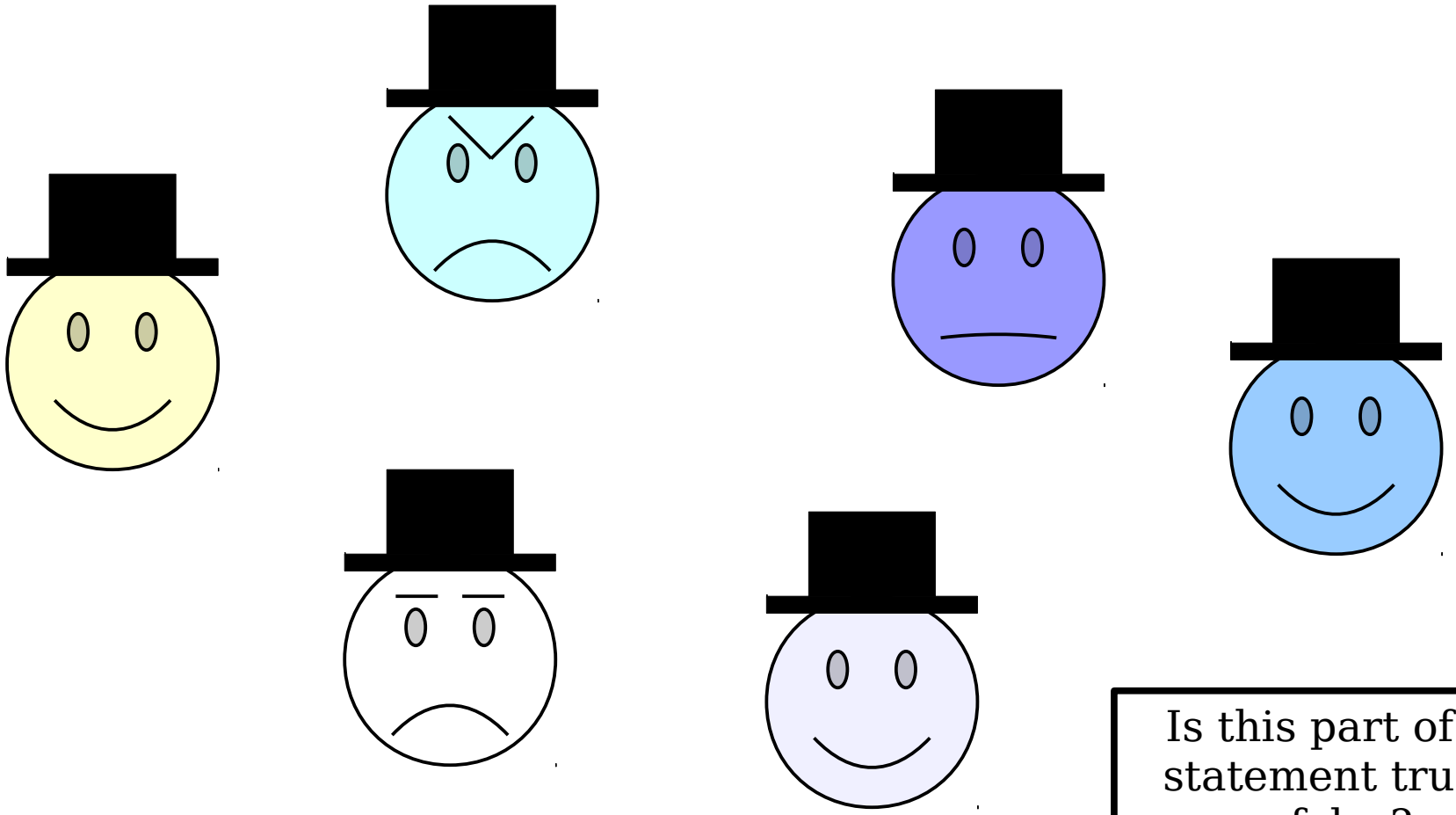
The Universal Quantifier



~~$\forall x.\ Smiling(x)$~~

Since $Smiling(x)$ is true for this choice x, this statement evaluates to false.

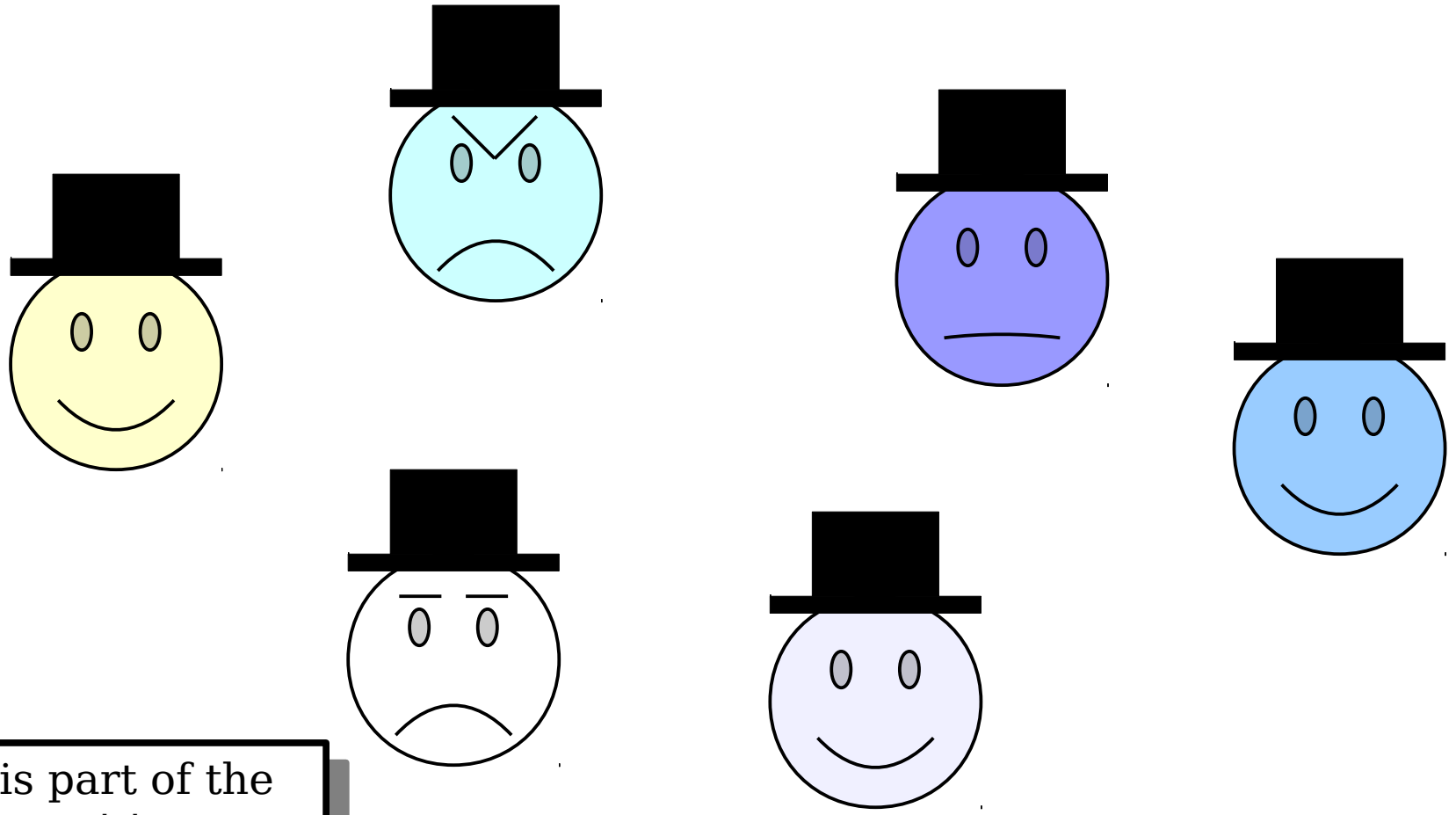
The Universal Quantifier



Is this part of the statement true or false?

$$(\forall x.\ \mathit{Smiling}(x)) \rightarrow (\forall y.\ \mathit{WearingHat}(y))$$

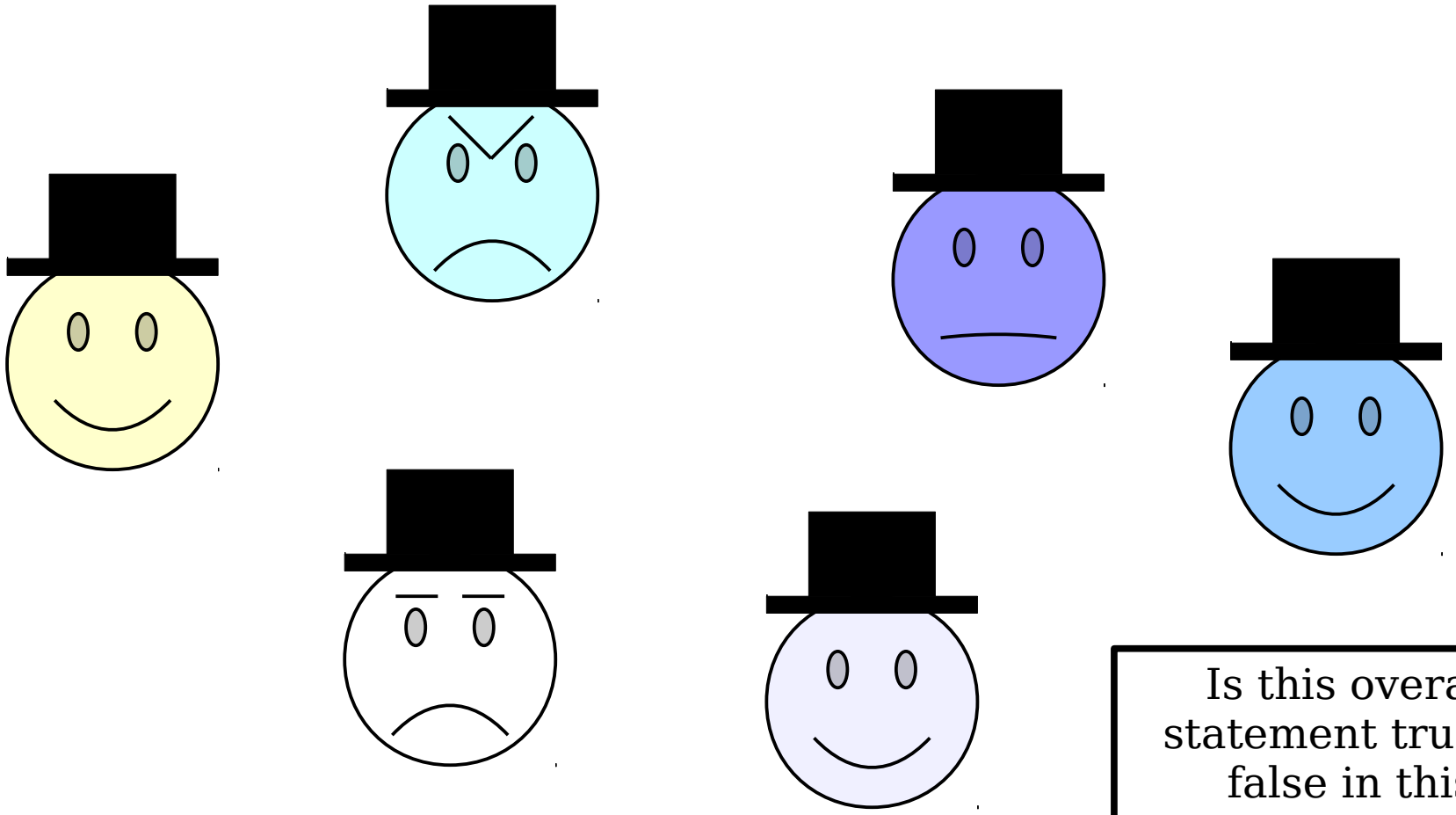
The Universal Quantifier



Is this part of the statement true or false?

$$\cancel{(\forall x.\ \mathit{Smiling}(x))} \rightarrow (\forall y.\ \mathit{WearingHat}(y))$$

The Universal Quantifier



Is this overall statement true or false in this scenario?

$(\forall x.\ \mathit{Smiling}(x)) \rightarrow (\forall y.\ \mathit{WearingHat}(y))$

Some muggles are intelligent.

$$\exists m.\ (Muggle(m) \wedge Intelligent(m))$$

∃ is the ***existential quantifier*** and says "for some choice of m, the following is true."

The Existential Quantifier

- A statement of the form

  $\exists x.$ *some-formula*

  is true if, for *some* choice of x, the statement ***some-formula*** is true when that x is plugged into it.

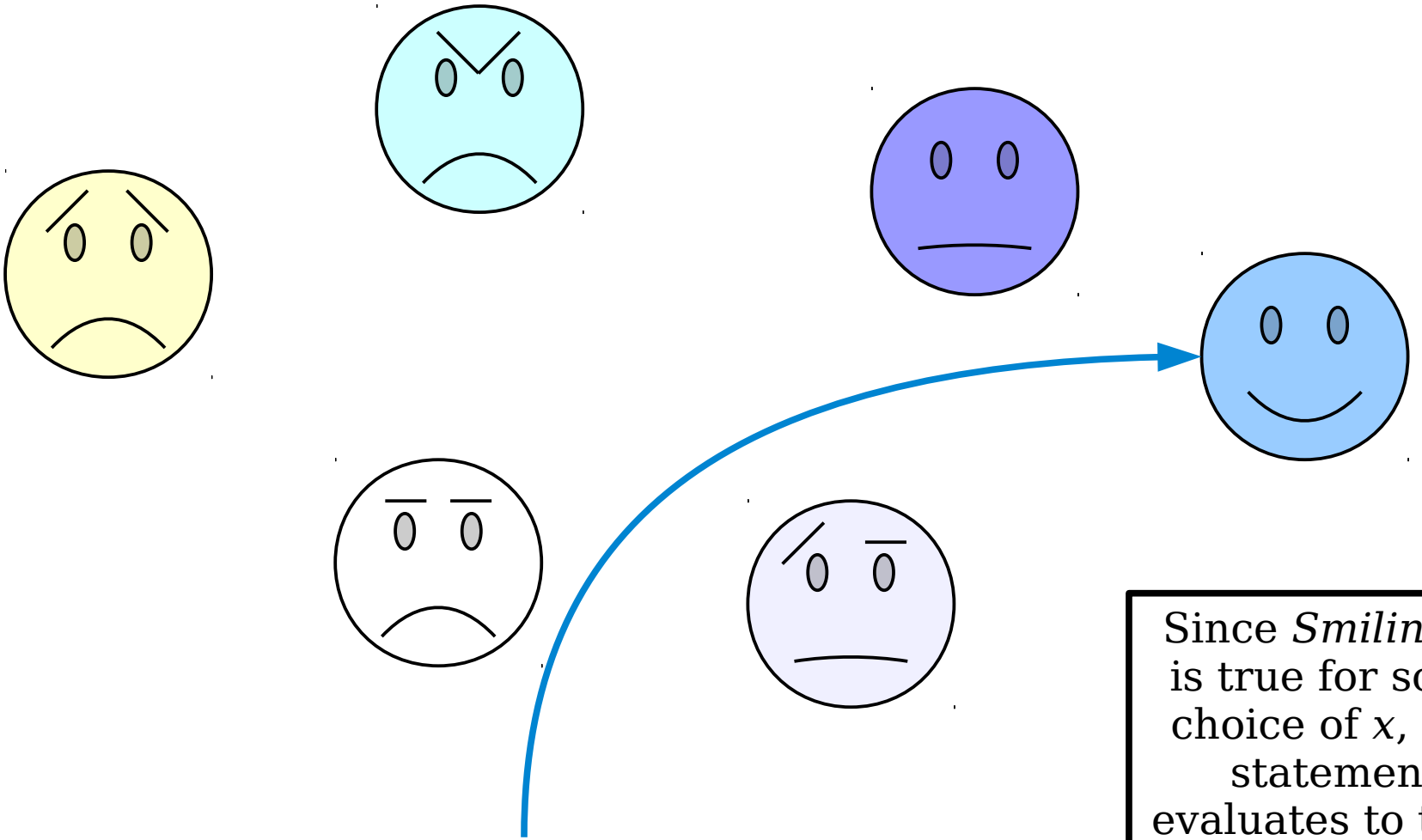
- Examples:

  $\exists x.\ (Even(x) \wedge Prime(x))$

  $\exists x.\ (TallerThan(x, me) \wedge LighterThan(x, me))$

  $(\exists x.\ Appreciates(x, me)) \rightarrow Happy(me)$

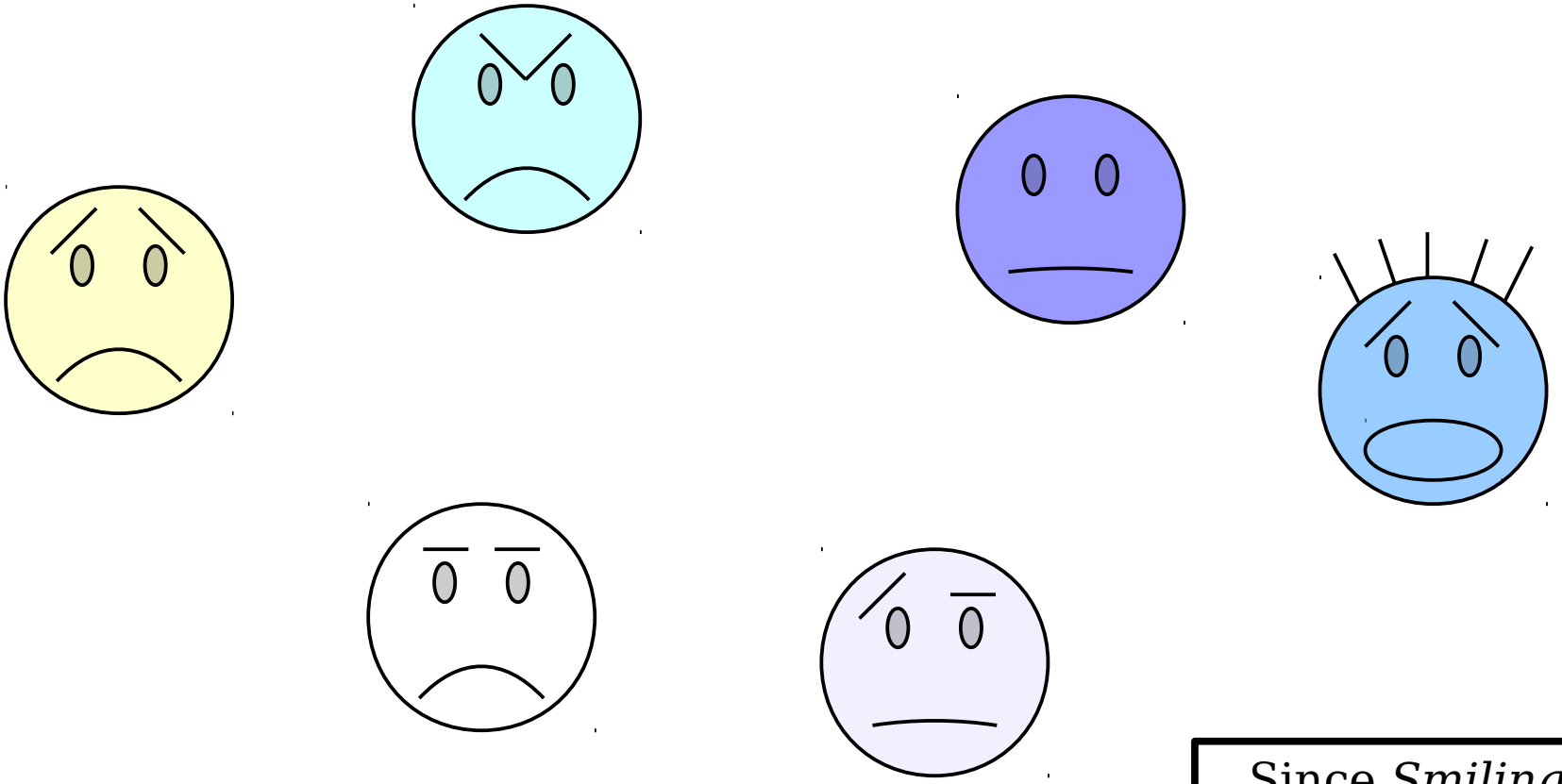
The Existential Quantifier



$\exists x.\ Smiling(x)$

Since $Smiling(x)$ is true for some choice of x, this statement evaluates to true.

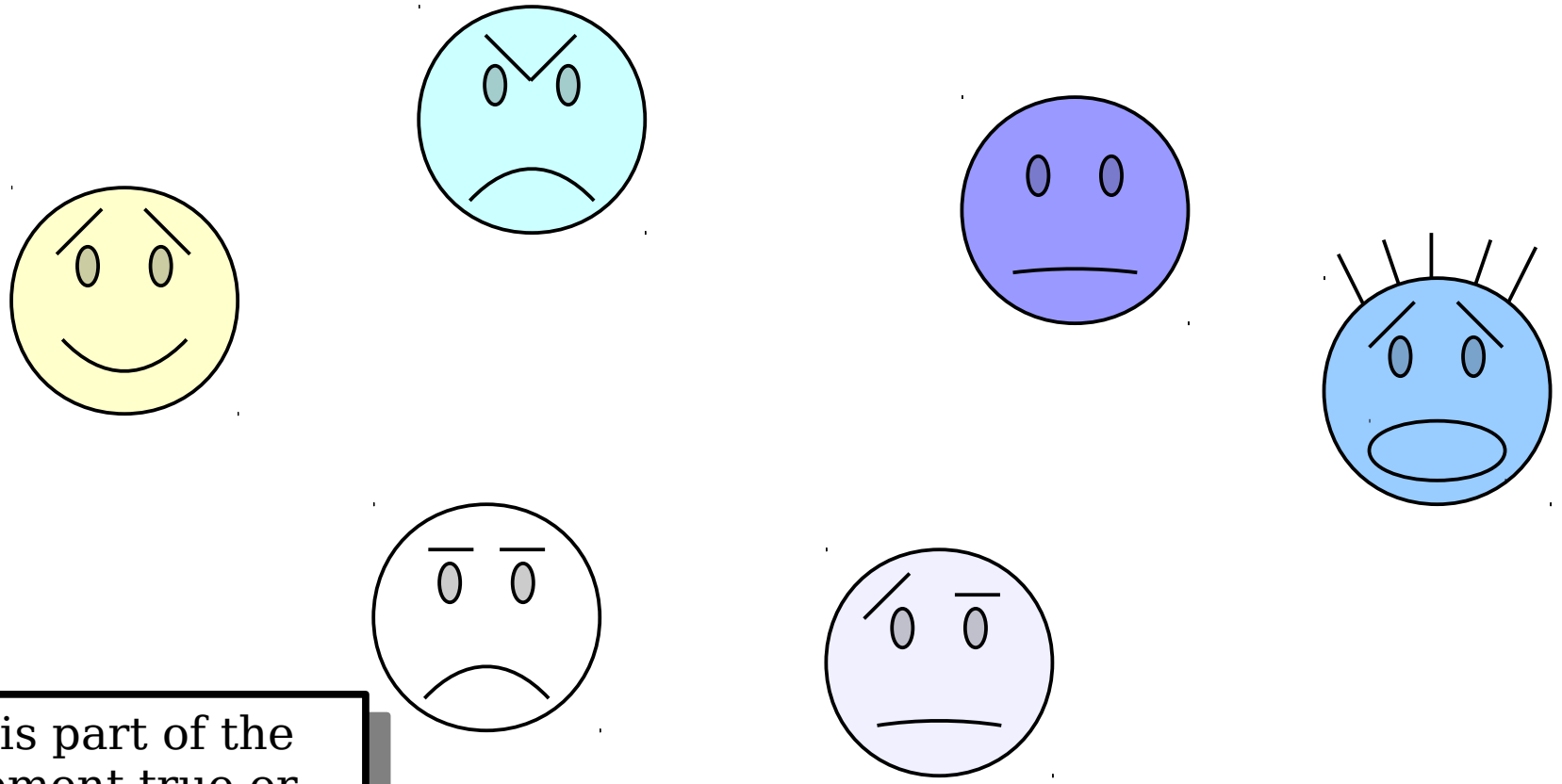
The Existential Quantifier



~~$\exists x.\ Smiling(x)$~~

Since $Smiling(x)$ is not true for any choice of x, this statement evaluates to false.

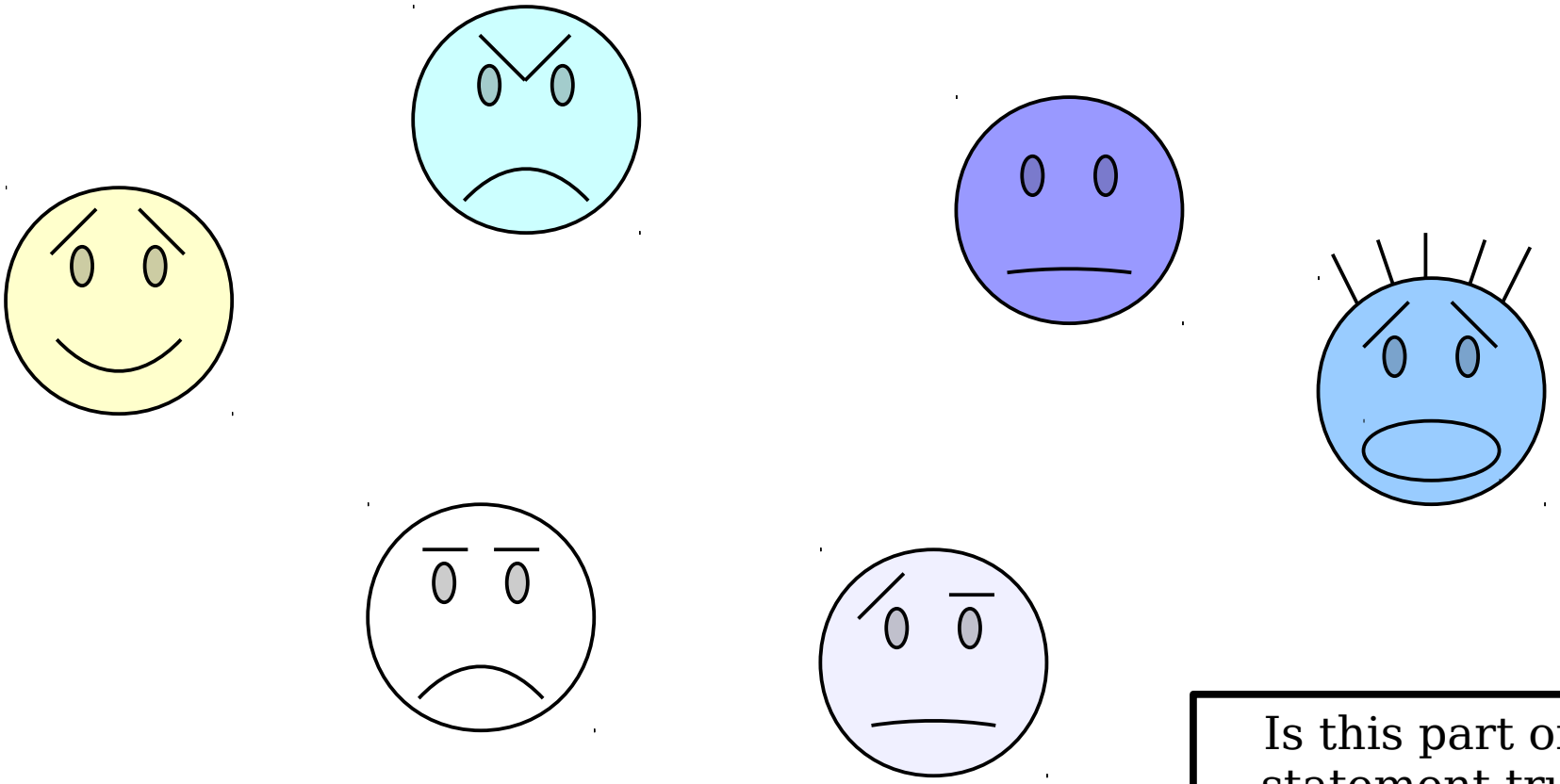
The Existential Quantifier



Is this part of the statement true or false?

$(\exists x.\ \mathit{Smiling}(x)) \rightarrow (\exists y.\ \mathit{WearingHat}(y))$

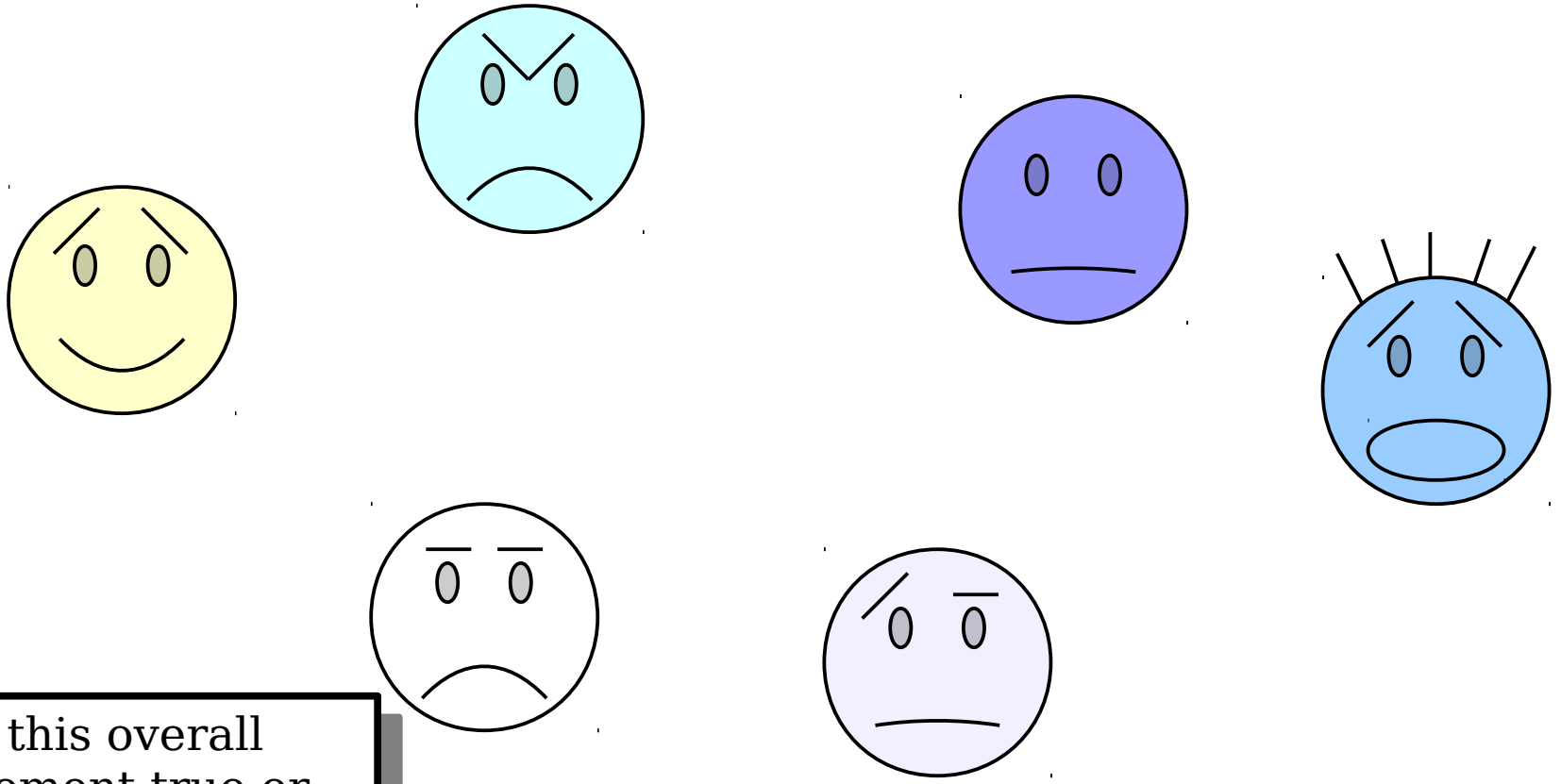
The Existential Quantifier



Is this part of the statement true or false?

$(\exists x.\ \mathit{Smiling}(x)) \rightarrow \sout{(\exists y.\ \mathit{WearingHat}(y))}$

The Existential Quantifier



Is this overall statement true or false?

~~$(\exists x.\ \mathit{Smiling}(x)) \rightarrow (\exists y.\ \mathit{WearingHat}(y))$~~

Some Technical Details

Variables and Quantifiers

- Each quantifier has two parts:
  - the variable that is introduced, and
  - the statement that's being quantified.
- The variable introduced is scoped just to the statement being quantified.

$$(\forall x.\ \mathit{Loves}(\mathit{You}, x)) \to (\forall y.\ \mathit{Loves}(y, \mathit{You}))$$

The variable x just lives here.

The variable y just lives here.

Variables and Quantifiers

- Each quantifier has two parts:
  - the variable that is introduced, and
  - the statement that's being quantified.
- The variable introduced is scoped just to the statement being quantified.

$$(\forall x.\ \mathit{Loves}(\mathit{You}, x)) \rightarrow (\forall x.\ \mathit{Loves}(x, \mathit{You}))$$

The variable x just lives here.

A different variable, also named x, just lives here.

Operator Precedence (Again)

- When writing out a formula in first-order logic, the quantifiers ∀ and ∃ have precedence just below ¬.
- The statement

$$\forall x.\ P(x) \vee R(x) \to Q(x)$$

  is parsed like this:

$$\big(\ (\forall x.\ P(x)\ )\ \ \vee\ \ R(x)\ \big)\ \to\ Q(x)$$

- This is syntactically invalid because the variable x is out of scope in the back half of the formula.
- To ensure that x is properly quantified, explicitly put parentheses around the region you want to quantify:

$$\forall x.\ (P(x) \vee R(x) \to Q(x))$$

Time-Out for Announcements!

Checkpoints Graded

- The PS1 checkpoint assignment has been graded.
- Review your feedback on GradeScope. Contact the staff (via Piazza or by stopping by office hours) if you have any questions.
- Some notes:
  - ***Make sure to list your partner through GradeScope.*** There's a space to list your partner when you submit the assignment. If you forget to do this, *they won't get credit for the assignment*!
  - ***Make sure to check your grade ASAP.*** For the reason listed above, make sure you have a grade recorded. If not, contact the course staff. Plus, that way, you can take our feedback into account when writing up your answers to the rest of the problem set questions.
- Best of luck on the rest of the problem set!

Proofwriting Checklist

- We've just released Handout 11, which contains a checklist of five good proofwriting tips.
- We'll be looking at these five specific points when grading your problem sets.
- ***Good idea:*** Before submitting, reread all of your proofs and make sure that you adhere to the conventions.
  - Already submitted? No worries! You can resubmit and we'll grade the second version.

Your Questions

“Industry or research? PhD or not to PhD?”

These are both really good options – it's more a matter of what you're interested in working on and what's a better fit for you.

Working on a Ph.D gives you a chance to work on a lot of problems you can't typically work on in industry. You'll get a thorough understanding of a particular problem domain and will learn how to work on new problems independently. Post-graduation, you can go into academia (there are a ton of openings!) or industry, and you'll have a lot of flexibility.

If you're thinking that you want to do a PhD, I'd strongly recommend trying to get involved in research in the CS department. Keep an eye open for CURIS for next year, and feel free to stop by professors' office hours to ask them about their work and how to get involved. Take classes outside the CS core, if you can, since that's another great way to meet professors.

“When have you felt inadequate?”

I'll take this one in class rather than writing things down for posterity's sake, but the short answer is “all the time.”

As for “what should you do about it?,” I don't think there's a one-size-fits all answer, and again I'll take this one in class.

“What's your favorite game?”

A tie between Codenames, Bananagrams, Wise and Otherwise, and One Night Werewolf. I love games you can play in a group where you can lose terribly and still have a really good time.

Back to CS103!

Translating into First-Order Logic

Translating Into Logic

- First-order logic is an excellent tool for manipulating definitions and theorems to learn more about them.
- Applications:
  - Determining the negation of a complex statement.
  - Figuring out the contrapositive of a tricky implication.

Translating Into Logic

- ***Translating statements into first-order logic is a lot more difficult than it looks.***
- There are a lot of nuances that come up when translating into first-order logic.
- We'll cover examples of both good and bad translations into logic so that you can learn what to watch for.
- We'll also show lots of examples of translations so that you can see the process that goes into it.

Using the predicates

- $Puppy(p)$, which states that p is a puppy, and
- $Cute(x)$, which states that x is cute,

write a sentence in first-order logic that means “all puppies are cute.”

An Incorrect Translation

All puppies are cute!

$\forall x.\ (\mathit{Puppy}(x) \wedge \mathit{Cute}(x))$

An Incorrect Translation

All puppies are cute!

$\forall x.\ (Puppy(x) \wedge Cute(x))$

This should work for any choice of x, including things that aren't puppies.

An Incorrect Translation

All puppies are cute!

$\forall x.\ (\sout{Puppy(x) \wedge Cute(x)})$

This should work for any choice of x, including things that aren't puppies.

An Incorrect Translation

All puppies are cute!

~~$\forall x.\ (Puppy(x) \wedge Cute(x))$~~

A statement of the form

$\forall x.$ *something*

is true only when *something* is true for *every* choice of x.

An Incorrect Translation

All puppies are cute!

~~$\forall x. (Puppy(x) \wedge Cute(x))$~~

This first-order statement is false even though the English statement is true. Therefore, it can't be a correct translation.

An Incorrect Translation

All puppies are cute!

$\forall x.\ (\mathit{Puppy}(x) \wedge \mathit{Cute}(x))$

The issue here is that this statement asserts that everything is a puppy. That's too strong of a claim to make.

A Better Translation

All puppies are cute!

$\forall x.\ (Puppy(x) \rightarrow Cute(x))$

This should work for any choice of x, including things that aren't puppies.

A Better Translation

All puppies are cute!

$\forall x.\ (\cancel{Puppy(x)} \rightarrow Cute(x))$

This should work for any choice of x, including things that aren't puppies.

A Better Translation

All puppies are cute!

$\forall x.\ (Puppy(x) \rightarrow Cute(x))$

A statement of the form

$\forall x.$ *something*

is true only when *something* is true for *every* choice of x.

“All *P*'s are *Q*'s”

translates as

$$\forall x.\ (P(x) \rightarrow Q(x))$$

Useful Intuition:

Universally-quantified statements are true unless there's a counterexample.

$$\forall x.\ (P(x) \to Q(x))$$

If x is a counterexample, it must have property P but not have property Q.

Using the predicates

- *Blobfish*(b), which states that b is a blobfish, and
- *Cute*(x), which states that x is cute,

write a sentence in first-order logic that means “some blobfish is cute.”

An Incorrect Translation

Some blobfish is cute.

$\exists x.\ (\mathit{Blobfish}(x) \rightarrow \mathit{Cute}(x))$

An Incorrect Translation

Some blobfish is cute.

$\exists x.\ (\mathit{Blobfish}(x) \rightarrow \mathit{Cute}(x))$

A statement of the form

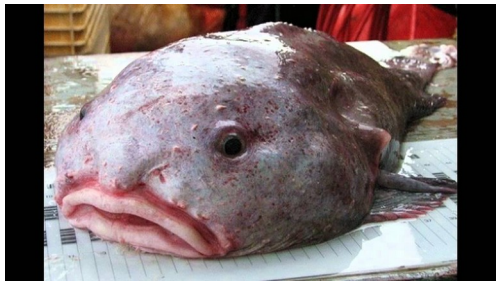
$\exists x.$ ***something***

is true only when *something* is true for at least one choice of x.

An Incorrect Translation

Some blobfish is cute.

$\exists x.\ (Blobfish(x) \rightarrow Cute(x))$

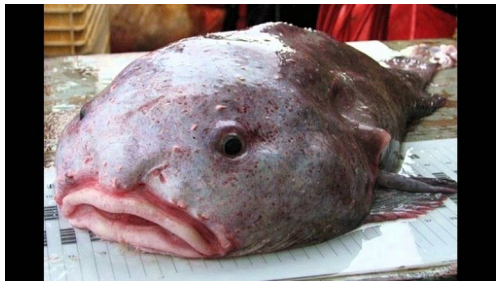


This first-order statement is true even though the English statement is false. Therefore, it can't be a correct translation.

An Incorrect Translation

Some blobfish is cute.

$\exists x.\ (\mathit{Blobfish}(x) \rightarrow \mathit{Cute}(x))$

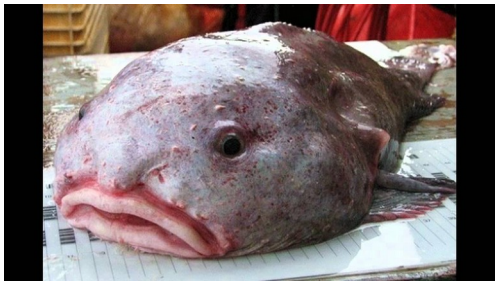


The issue here is that implications are true whenever the antecedent is false. This statement "accidentally" is true because of what happens when x isn't a blobfish.

A Correct Translation

Some blobfish is cute.

~~$\exists x.\ (Blobfish(x) \wedge Cute(x))$~~



A statement of the form

$\exists x.$ ***something***

is true only when *something* is true for at least one choice of x.

“Some *P* is a *Q*”

translates as

$\exists x.\ (P(x) \wedge Q(x))$

Useful Intuition:

Existentially-quantified statements are false unless there's a positive example.

$$\exists x.\ (P(x) \wedge Q(x))$$

If x is an example, it *must* have property P on top of property Q.

Good Pairings

- The ∀ quantifier *usually* is paired with →.

$$\forall x.\ (P(x) \rightarrow Q(x))$$

- The ∃ quantifier *usually* is paired with ∧.

$$\exists x.\ (P(x) \wedge Q(x))$$

- In the case of ∀, the → connective prevents the statement from being *false* when speaking about some object you don't care about.
- In the case of ∃, the ∧ connective prevents the statement from being *true* when speaking about some object you don't care about.

Next Time

- ***First-Order Translations***
  - How do we translate from English into first-order logic?
- ***Quantifier Orderings***
  - How do you select the order of quantifiers in first-order logic formulas?
- ***Negating Formulas***
  - How do you mechanically determine the negation of a first-order formula?
- ***Expressing Uniqueness***
  - How do we say there's just one object of a certain type?