First-Order Logic

Part One

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: $p \to q$
  - 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

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

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

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

$$In(MyHeart, Havana) \wedge TookBackTo(Him, Me, EastAtlanta)$$

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

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

$$In(MyHeart, Havana) \wedge TookBackTo(Him, Me, EastAtlanta)$$

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

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

$$In(MyHeart, Havana) \wedge TookBackTo(Him, Me, EastAtlanta)$$

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

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

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

$$In(MyHeart, Havana) \wedge TookBackTo(Him, Me, EastAtlanta)$$

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

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

$$In(MyHeart, Havana) \wedge TookBackTo(Him, Me, EastAtlanta)$$

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

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

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

$$In(MyHeart, Havana) \wedge TookBackTo(Him, Me, EastAtlanta)$$

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

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

$$In(MyHeart, Havana) \wedge TookBackTo(Him, Me, EastAtlanta)$$

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:

$$Cute(Quokka)$$

$$ArgueIncessantly(Democrats, Republicans)$$

- Applying a predicate to arguments produces a proposition, which is either true or false.
- Typically, when you're working in FOL, you'll have a list of predicates, what they stand for, and how many arguments they take. It'll be given separately than the formulas you write.

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.

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

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

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

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

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

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

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

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

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

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 propositions (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!*

The Type-Checking Table

	… operate on …	… and produce
Connectives (↔, ∧, etc.) …	propositions	a proposition
Predicates (=, etc.) …	objects	a proposition
Functions …	objects	an object

One last (and major) change

Some muggle is intelligent.

Some muggle is intelligent.

$\exists m.\ (\mathit{Muggle}(m) \wedge \mathit{Intelligent}(m))$

Some muggle is 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.

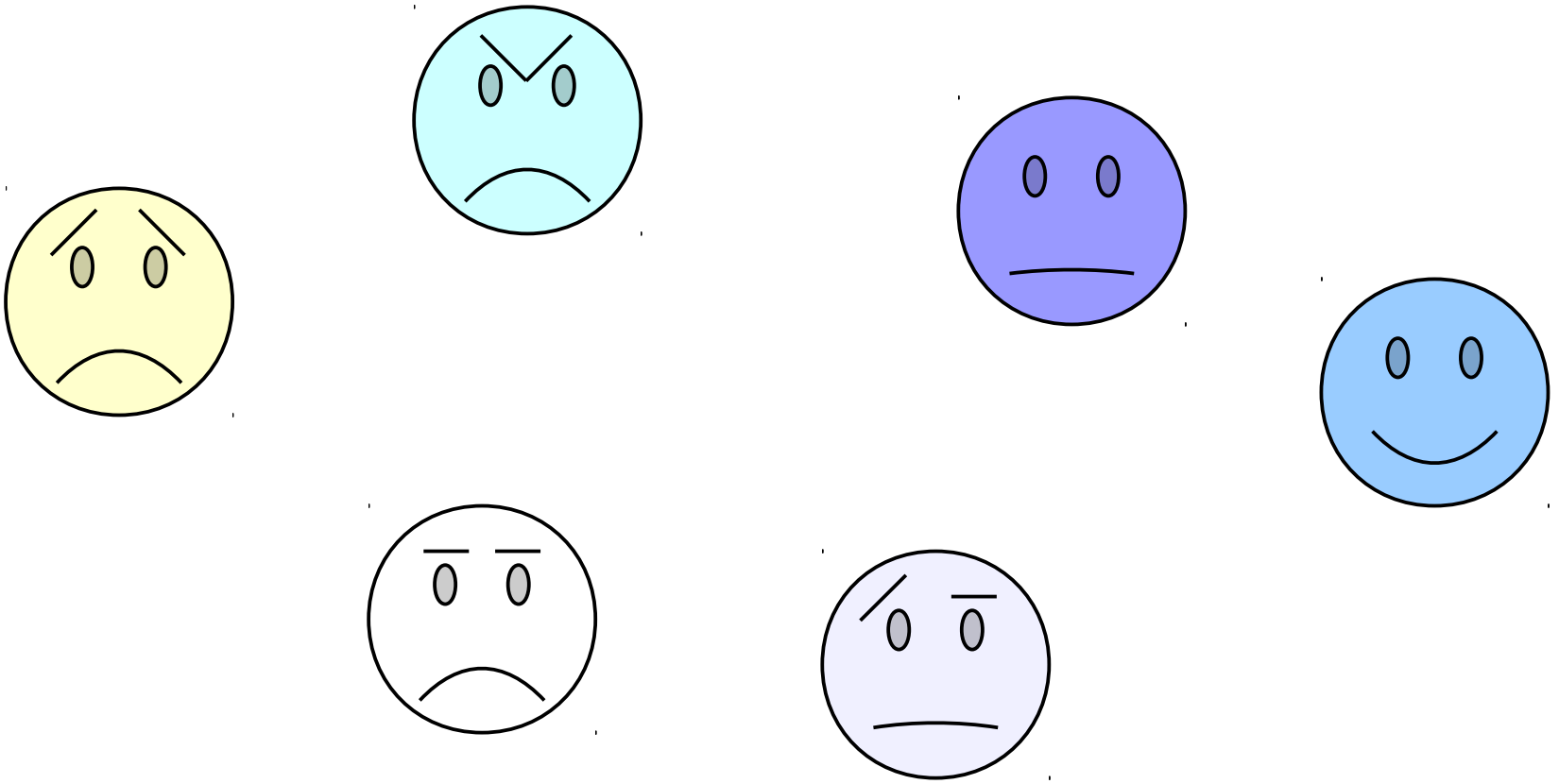
- Examples:

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

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

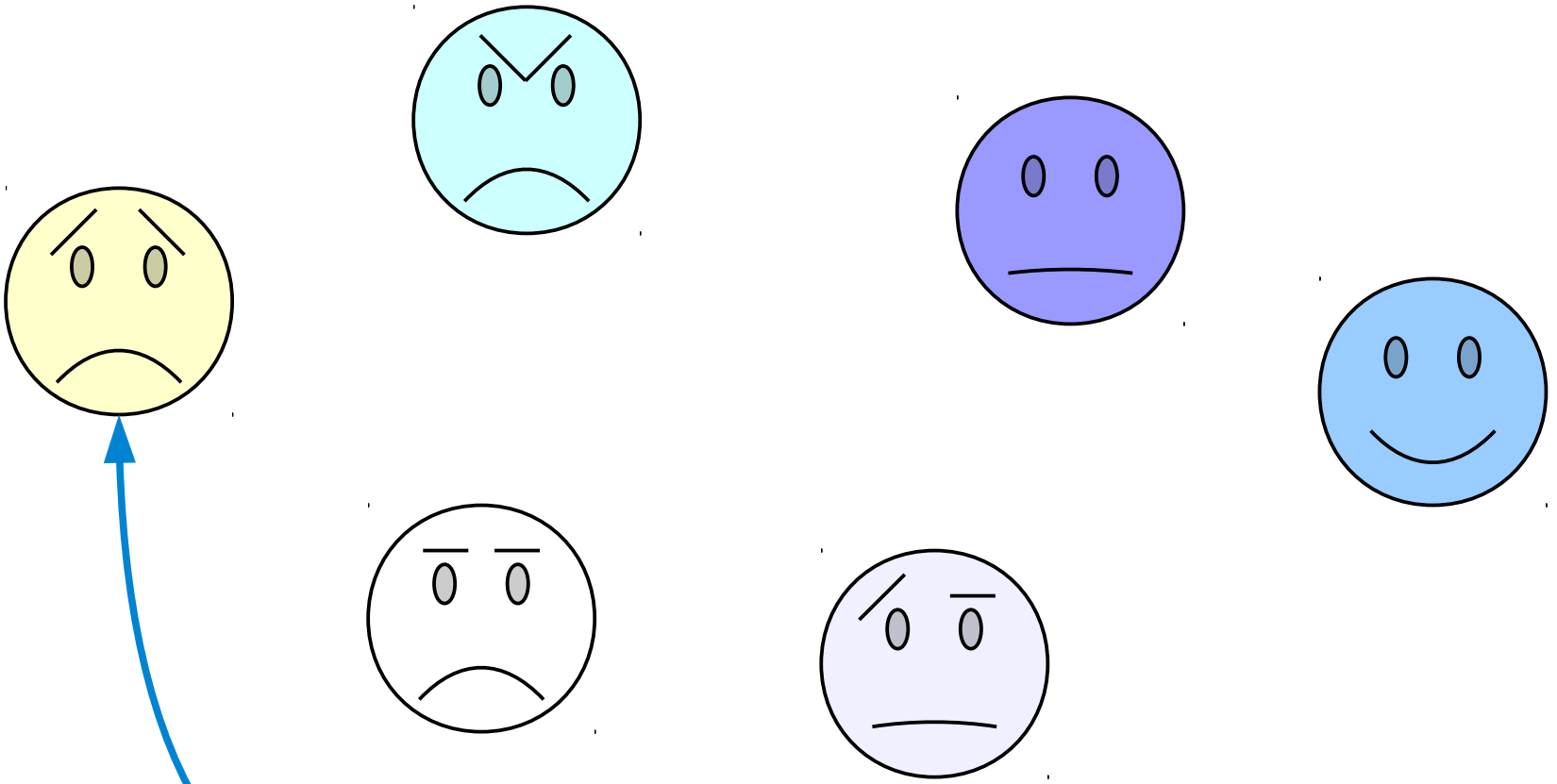
  $(\exists w.\ Will(w)) \rightarrow (\exists x.\ Way(x))$

The Existential Quantifier



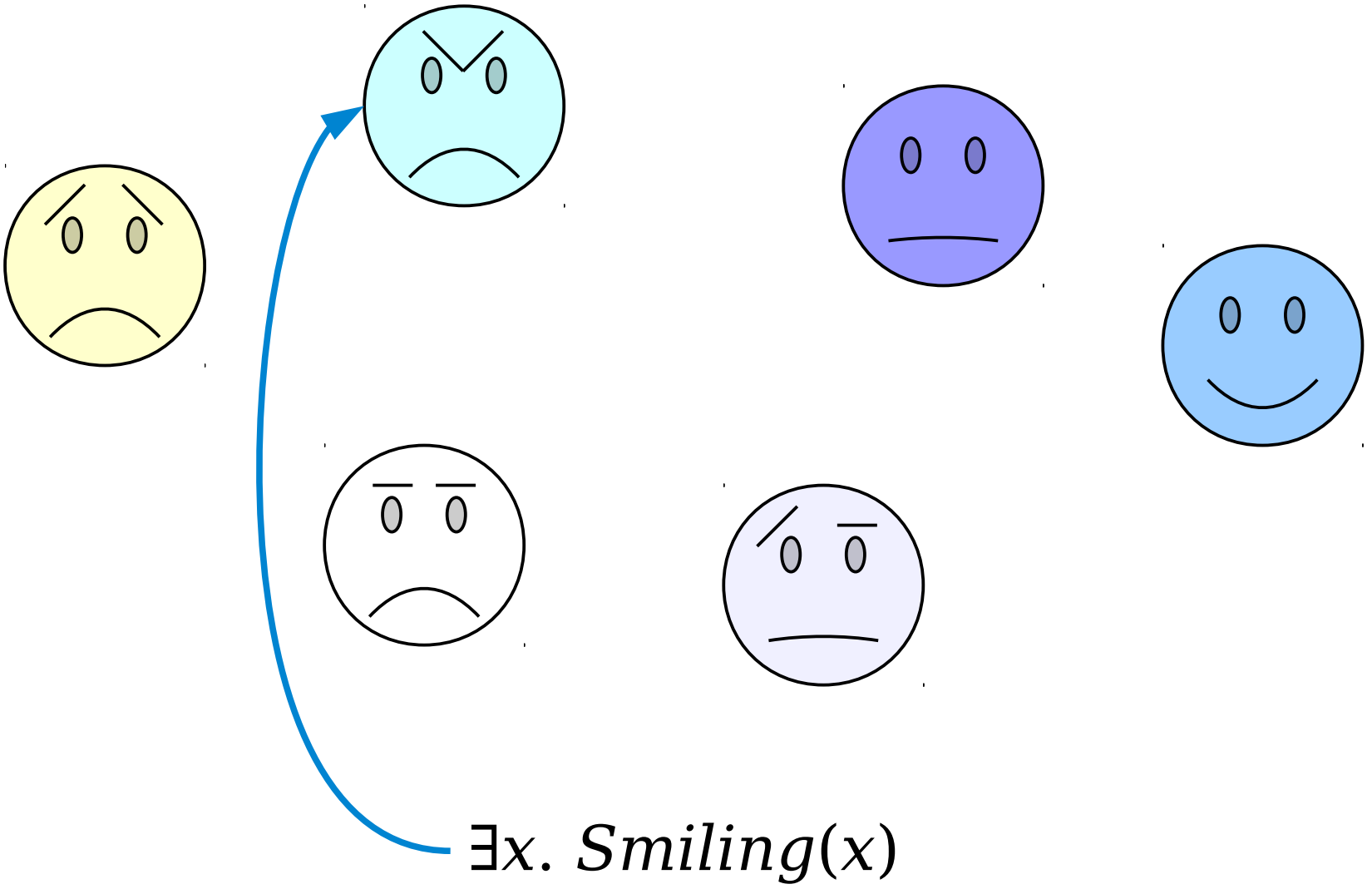
$\exists x.\ \mathit{Smiling}(x)$

The Existential Quantifier

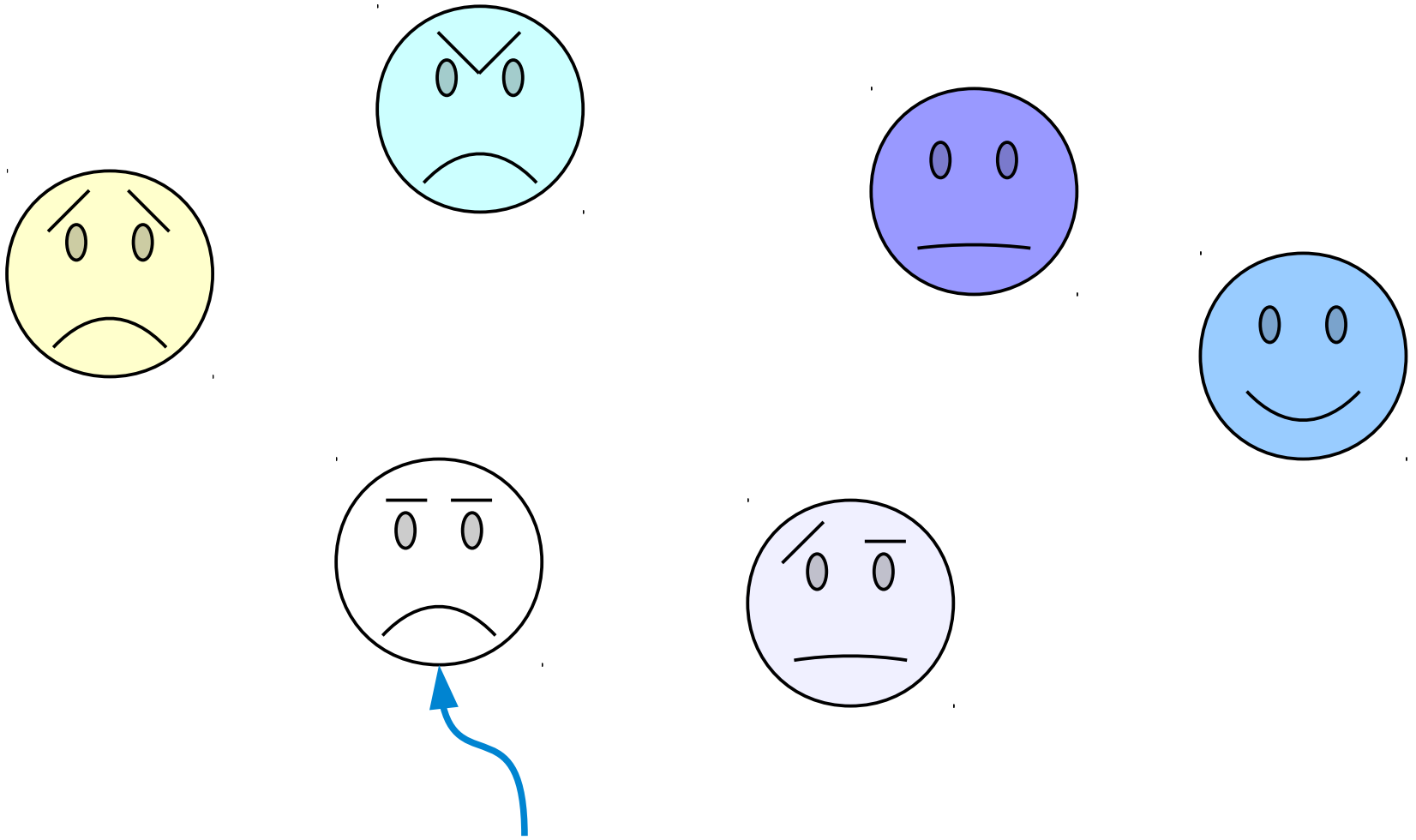


$\exists x.\ \mathit{Smiling}(x)$

The Existential Quantifier

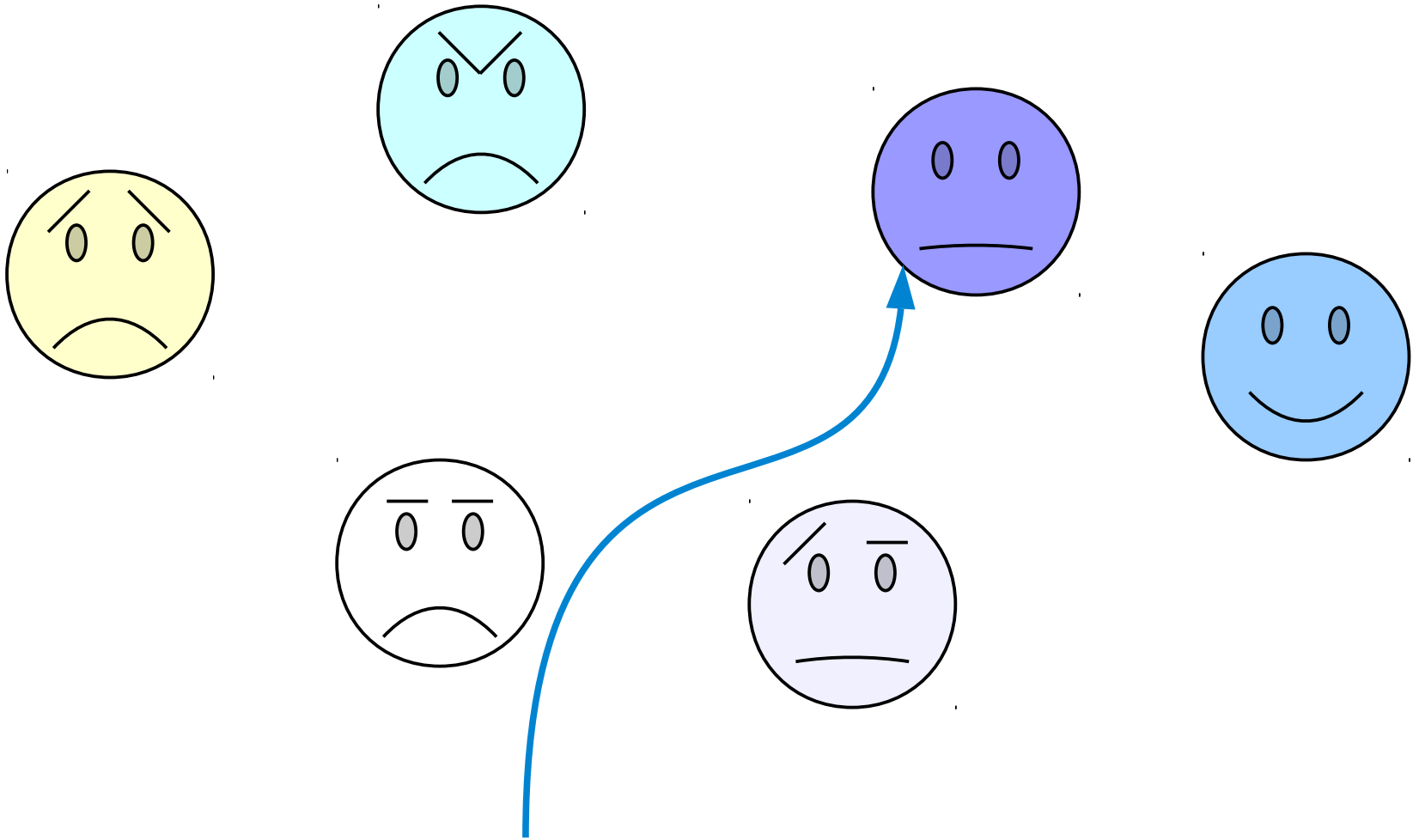


The Existential Quantifier



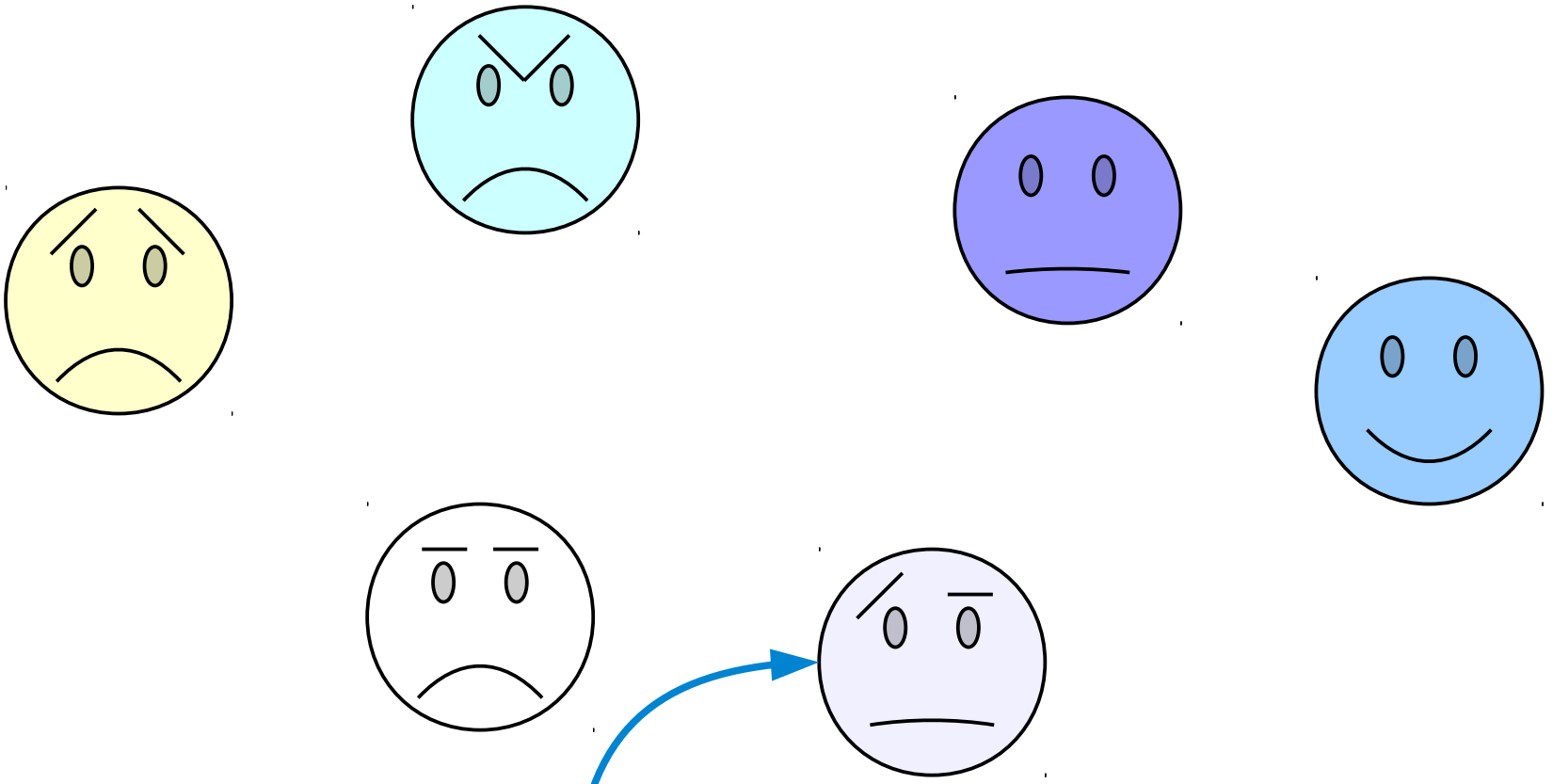
$\exists x.\ \mathit{Smiling}(x)$

The Existential Quantifier



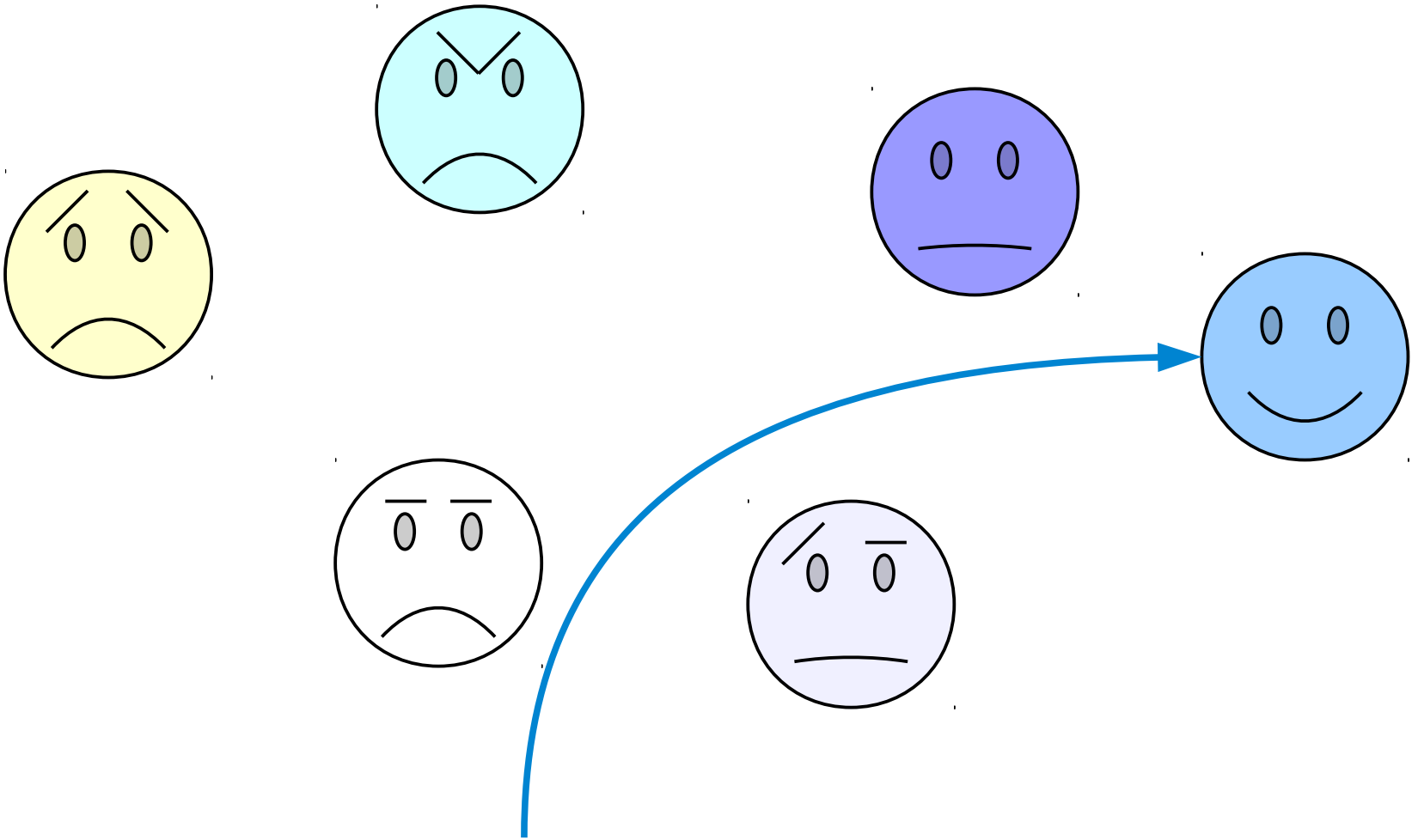
$\exists x.\ \mathit{Smiling}(x)$

The Existential Quantifier



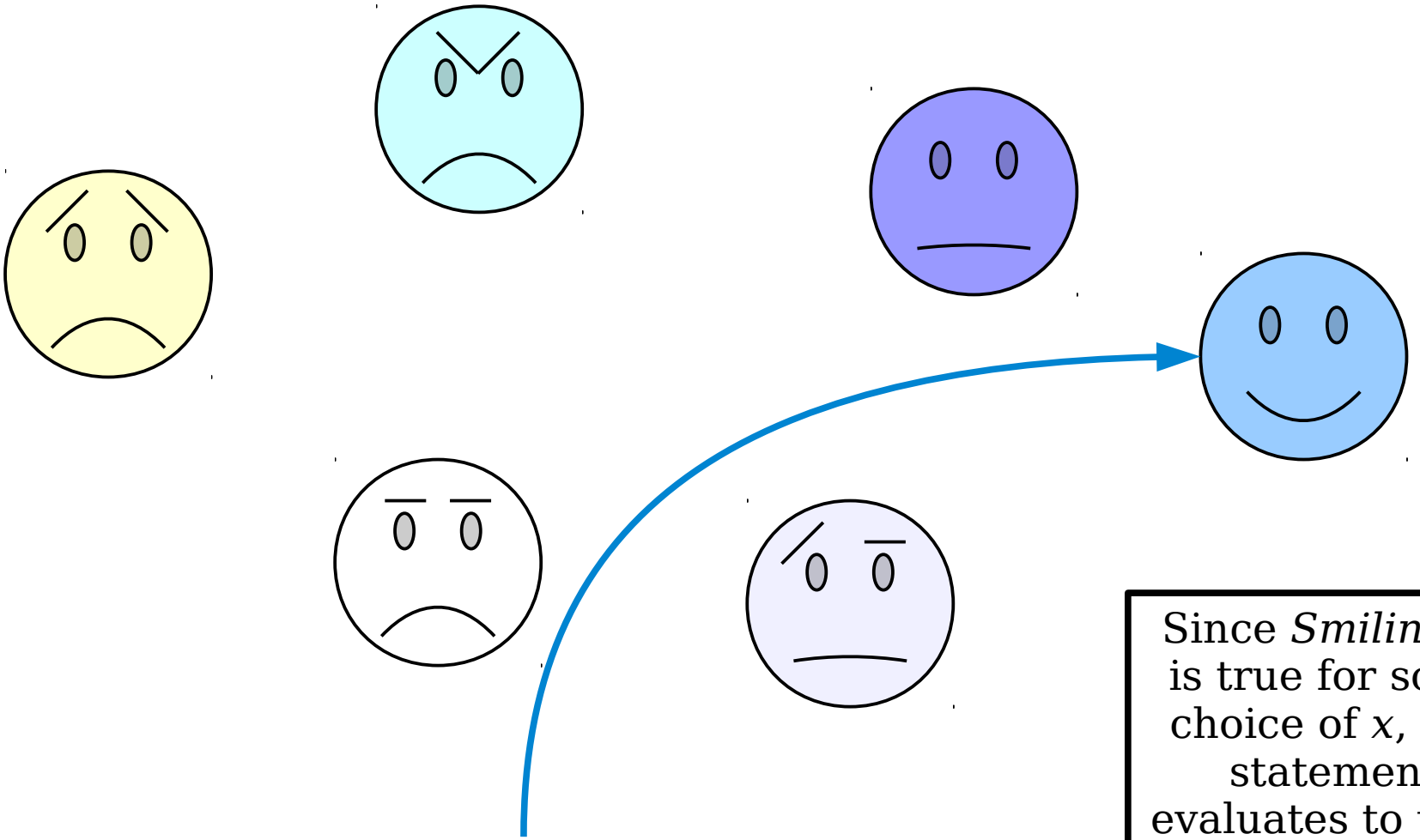
$\exists x.\ Smiling(x)$

The Existential Quantifier



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

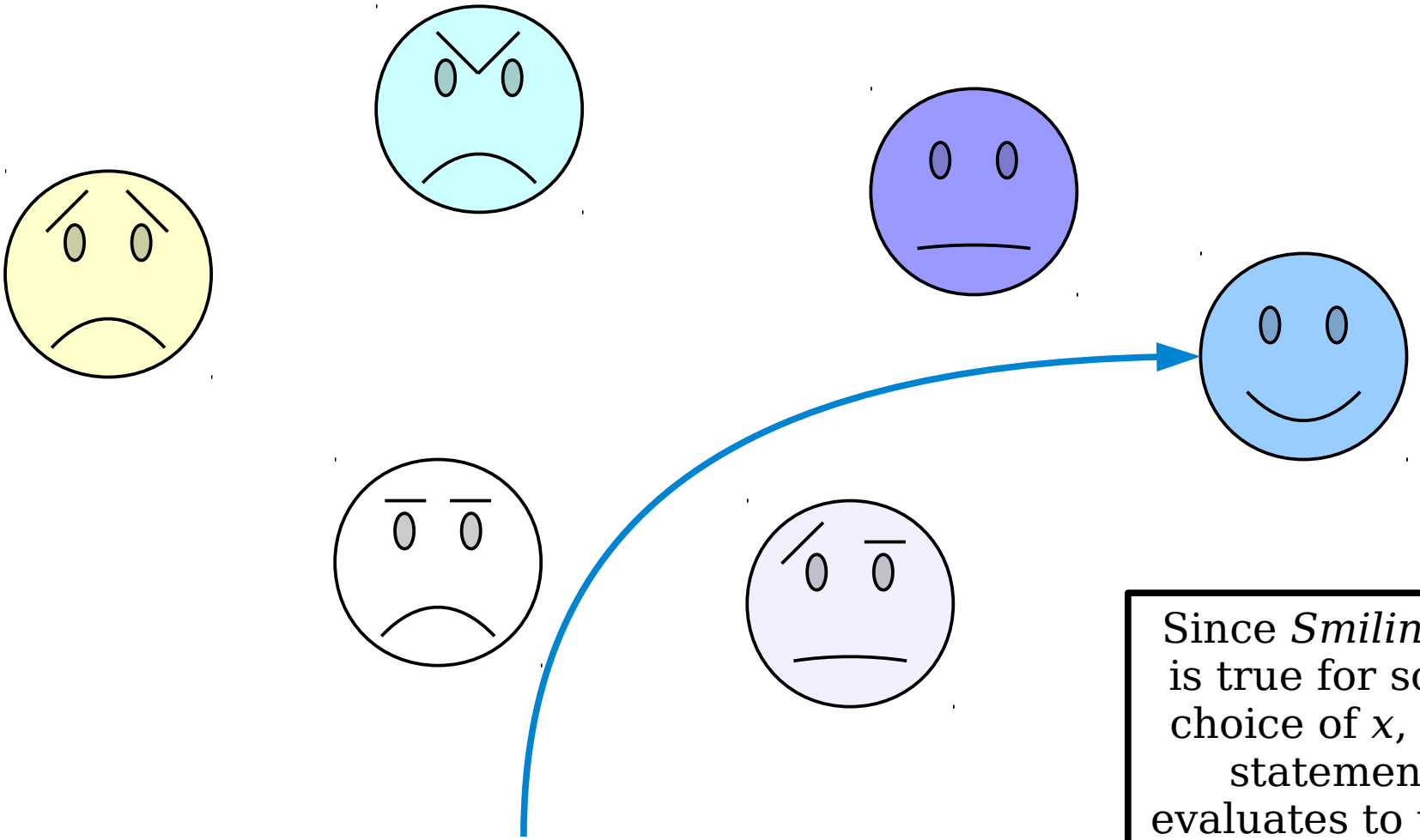
The Existential Quantifier



$\exists x.\ Smiling(x)$

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

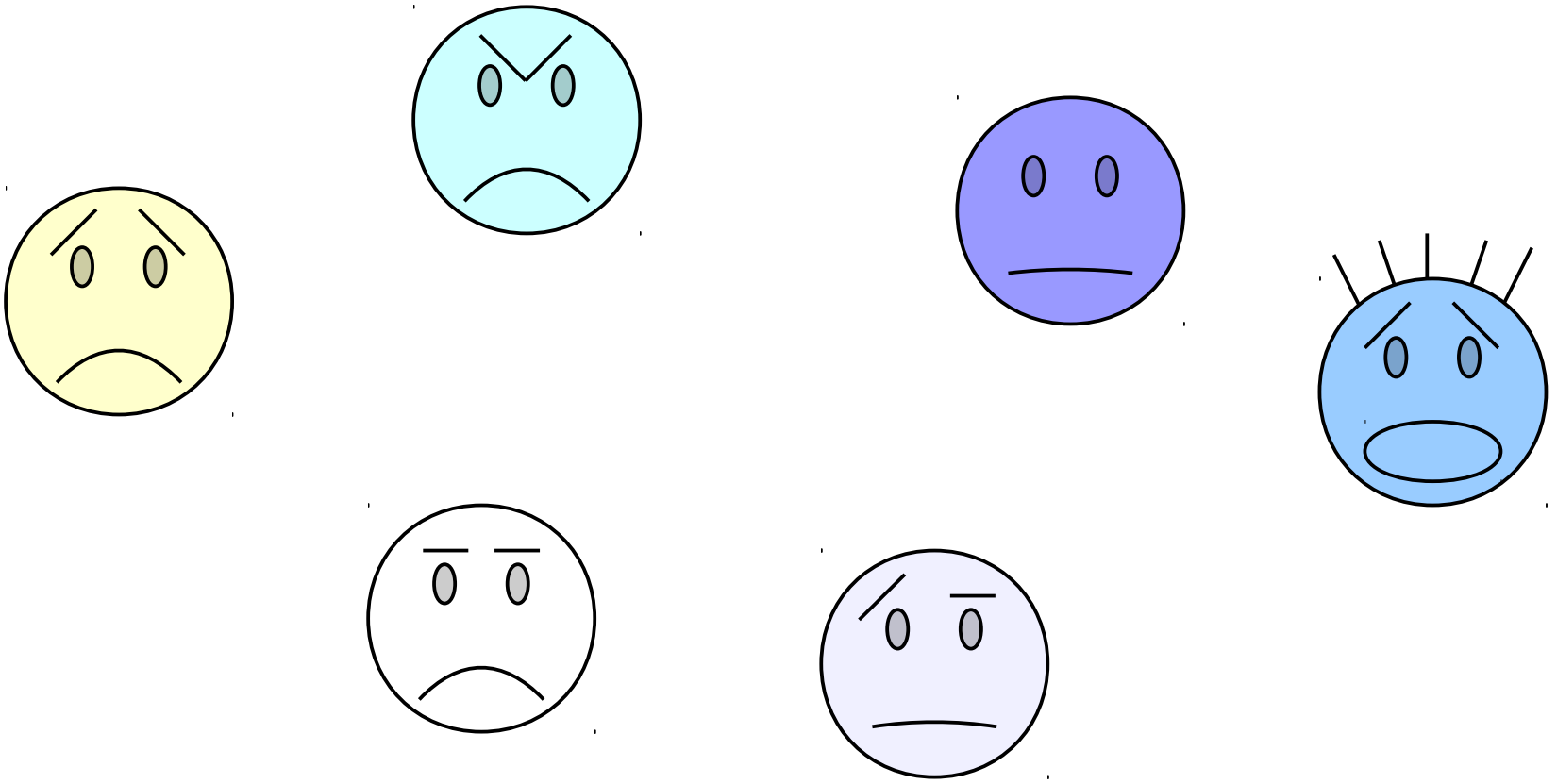
The Existential Quantifier



$\exists x.\ Smiling(x)$

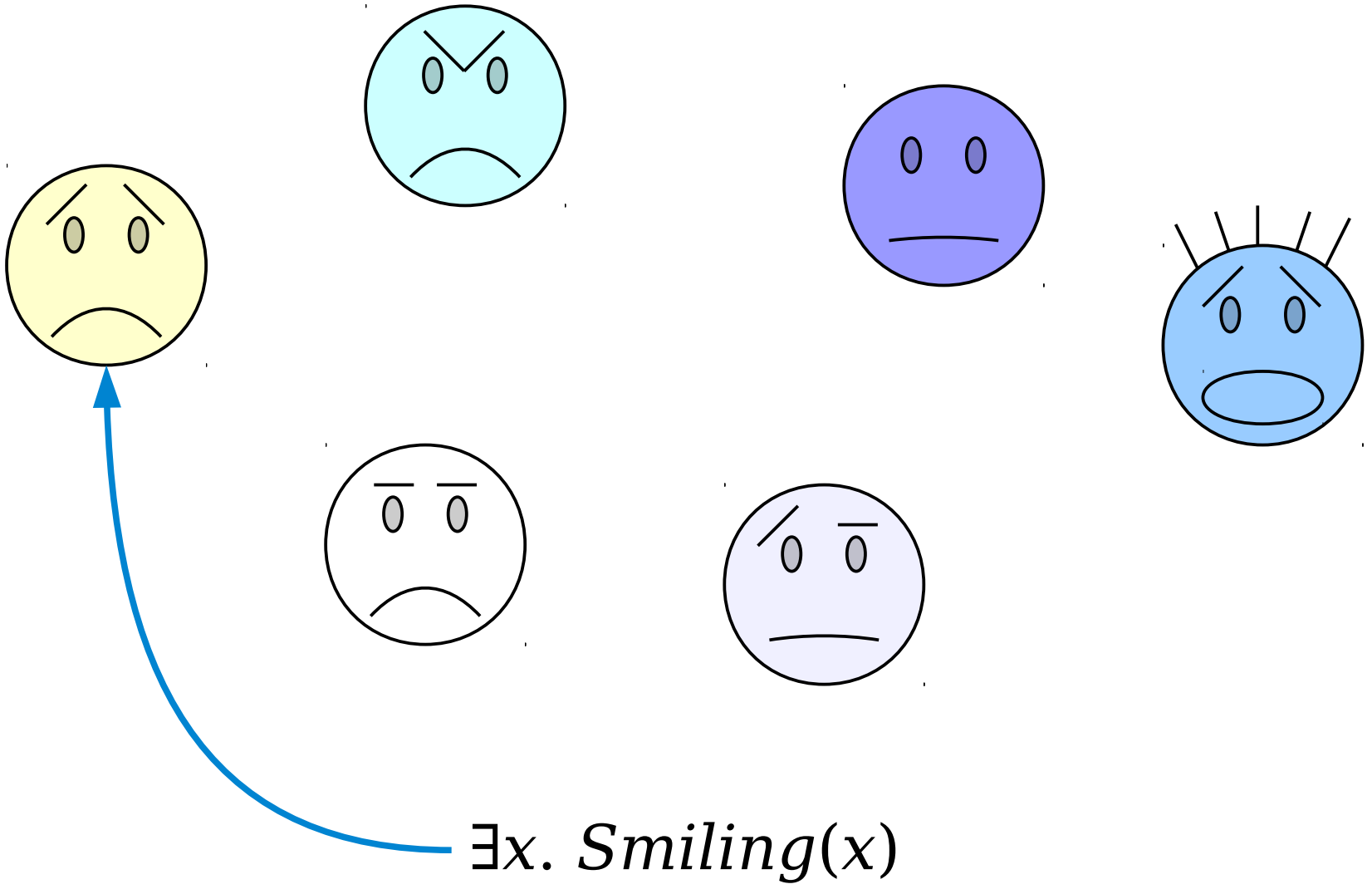
Since $Smiling(x)$ is true for some choice of x, this statement evaluates to true.

The Existential Quantifier

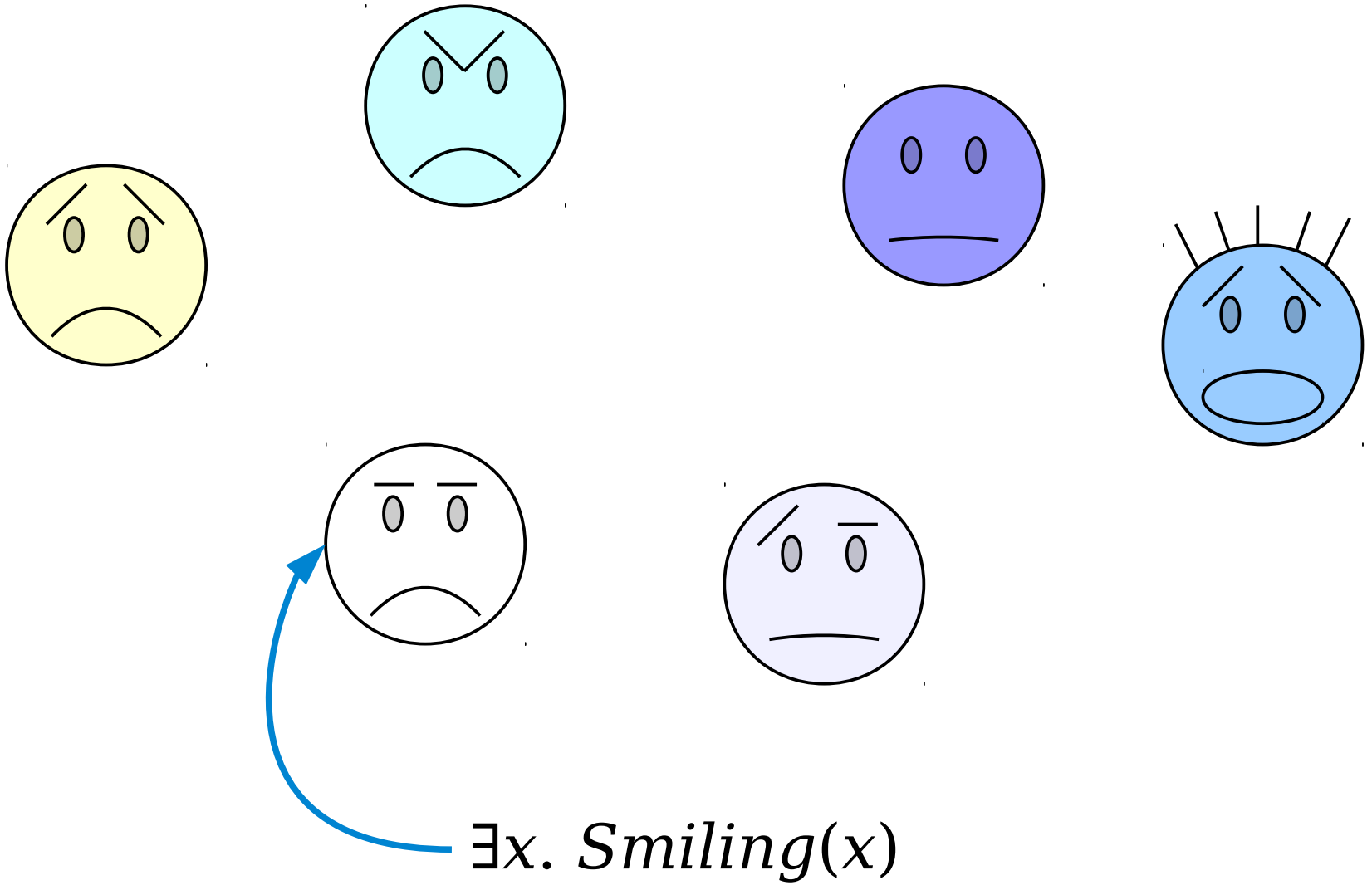


$\exists x.\ \mathit{Smiling}(x)$

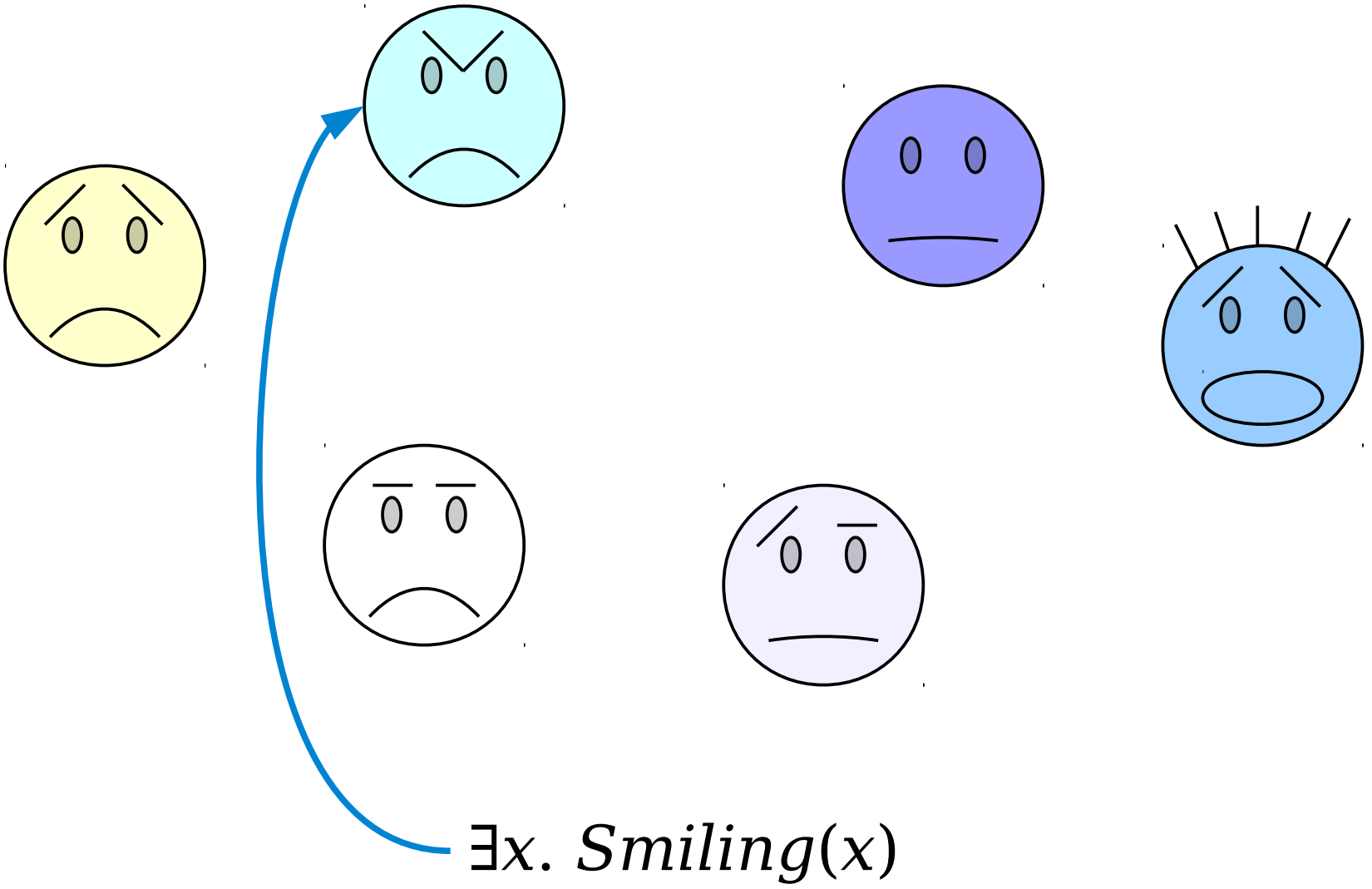
The Existential Quantifier



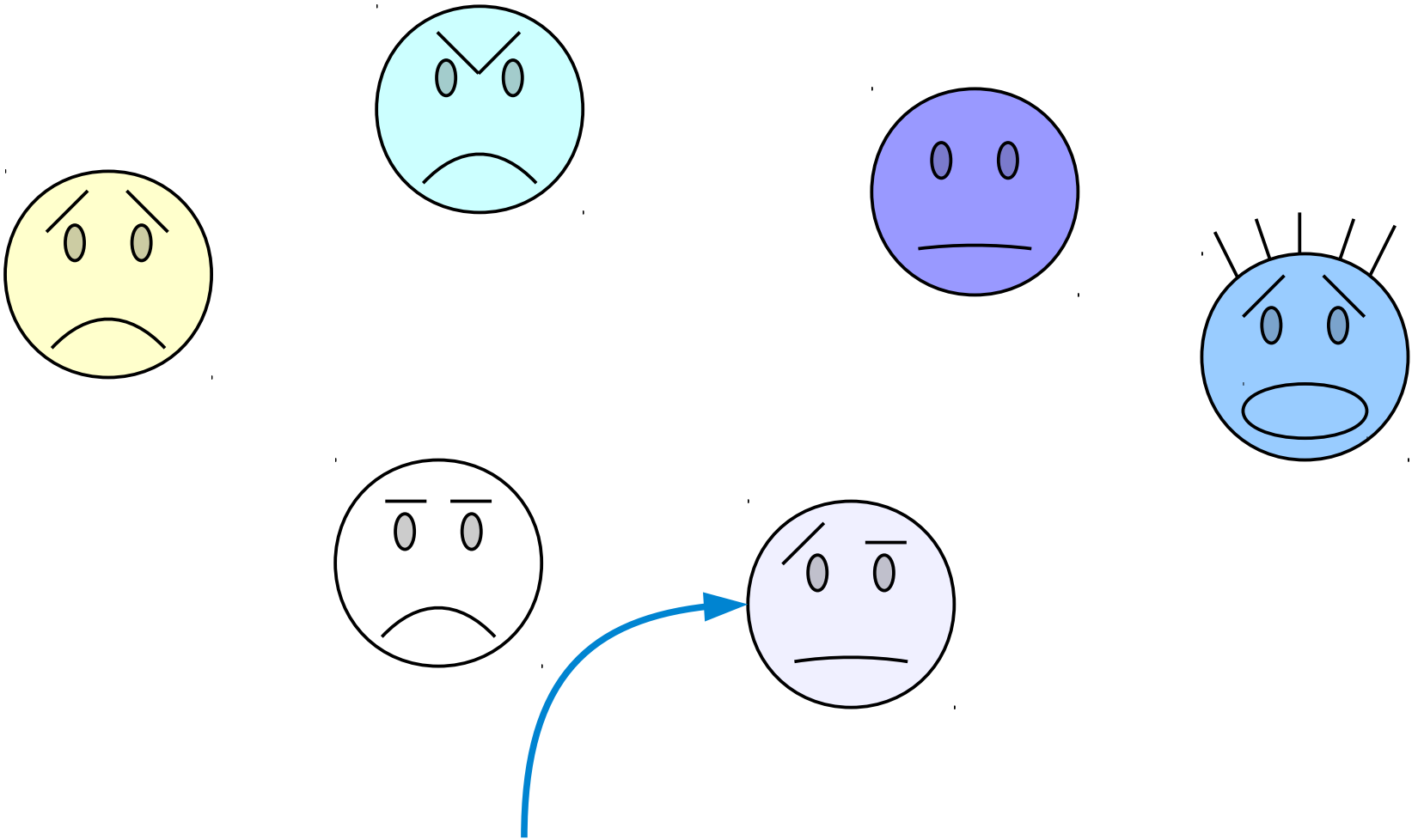
The Existential Quantifier



The Existential Quantifier

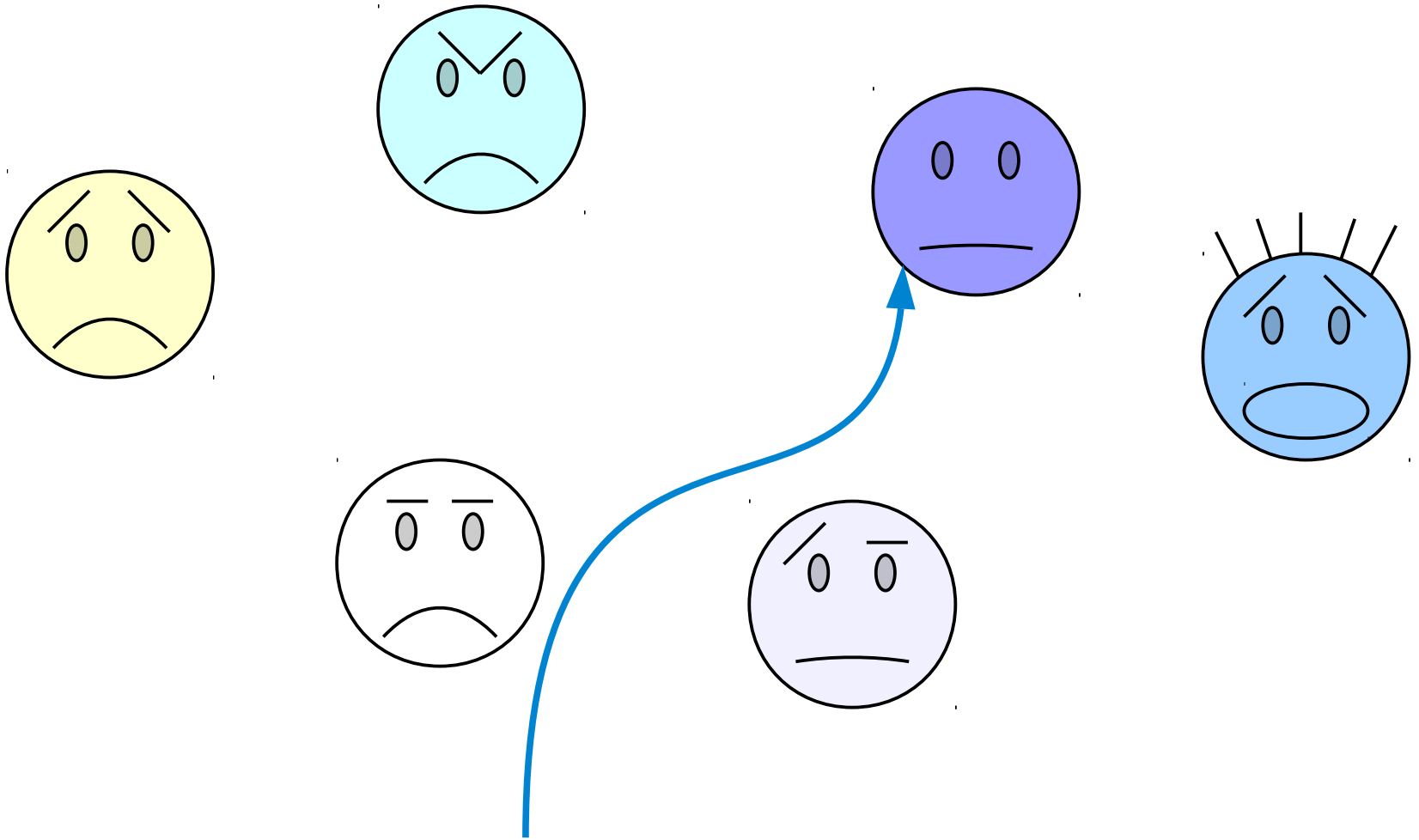


The Existential Quantifier



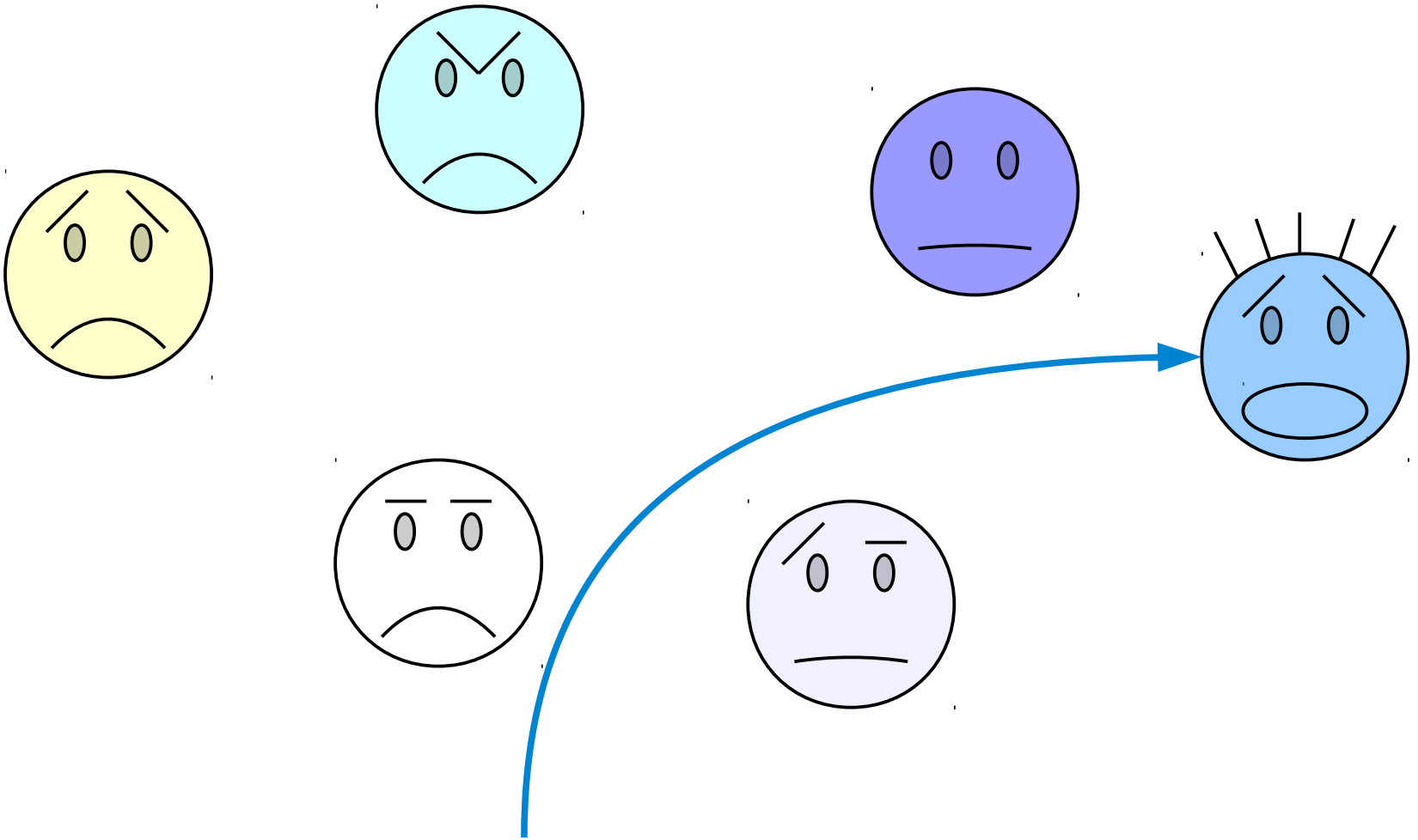
$\exists x.\ \mathit{Smiling}(x)$

The Existential Quantifier



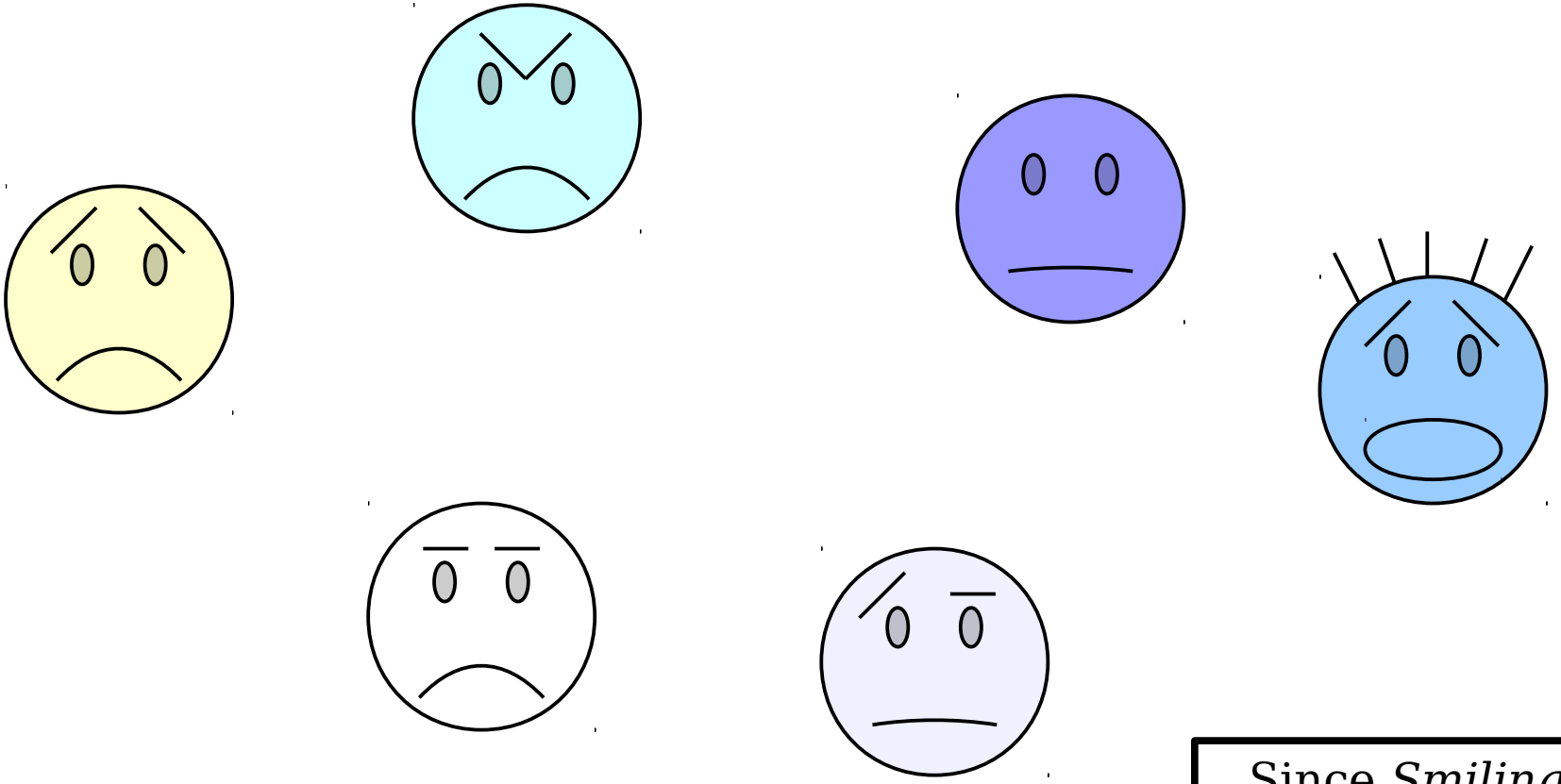
$\exists x.\ \mathit{Smiling}(x)$

The Existential Quantifier



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

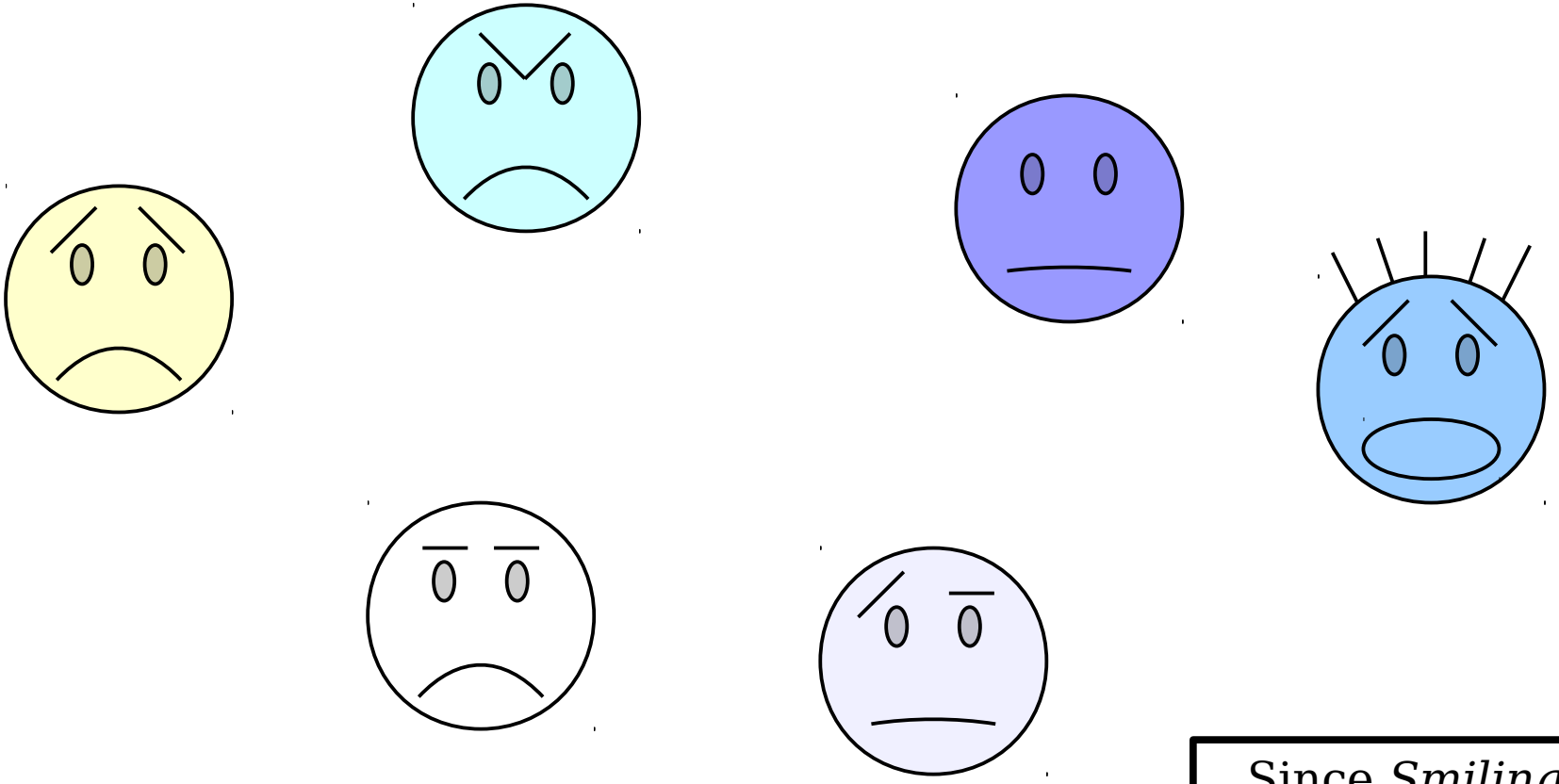
The Existential Quantifier



$\exists x.\ Smiling(x)$

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

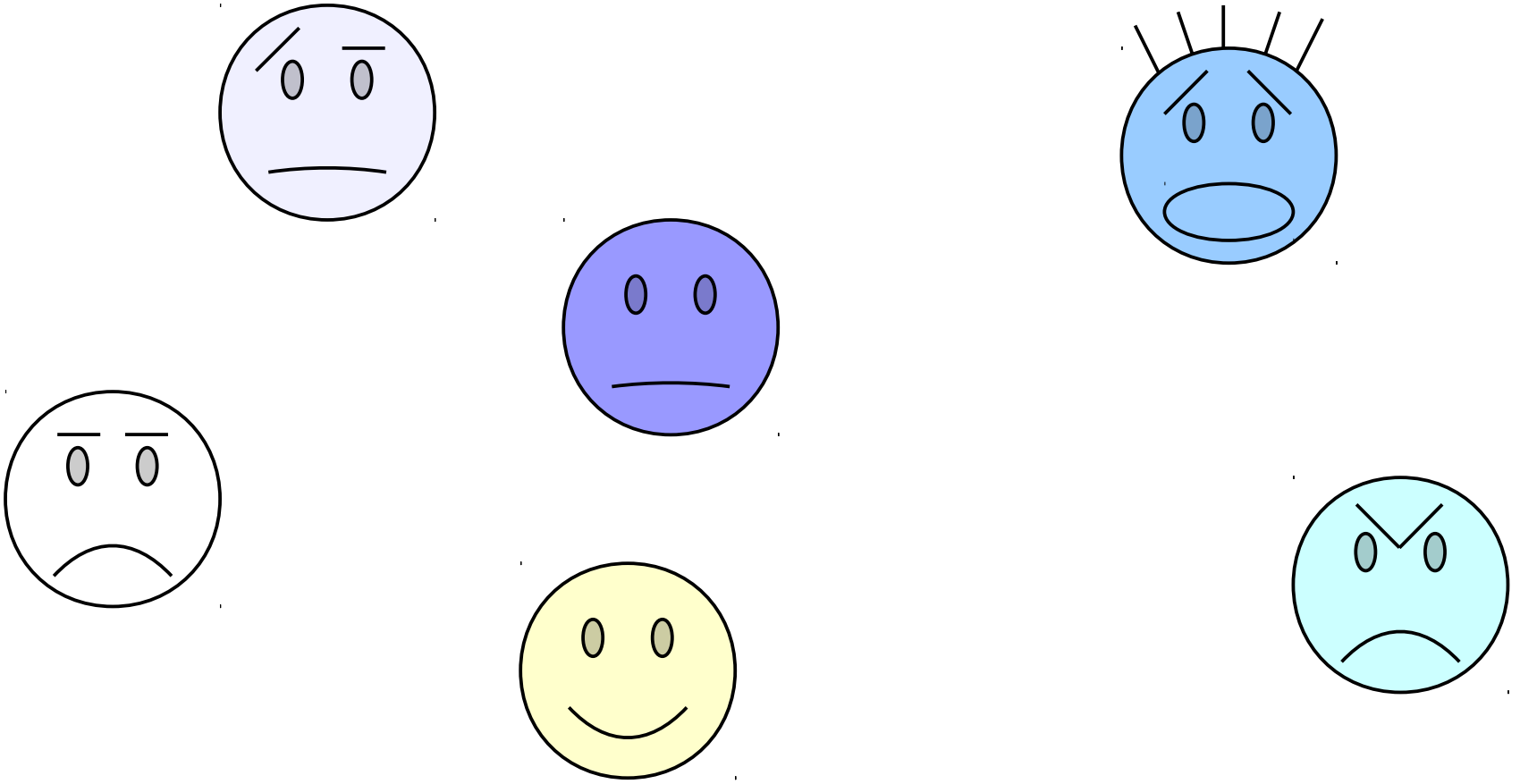
The Existential Quantifier



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

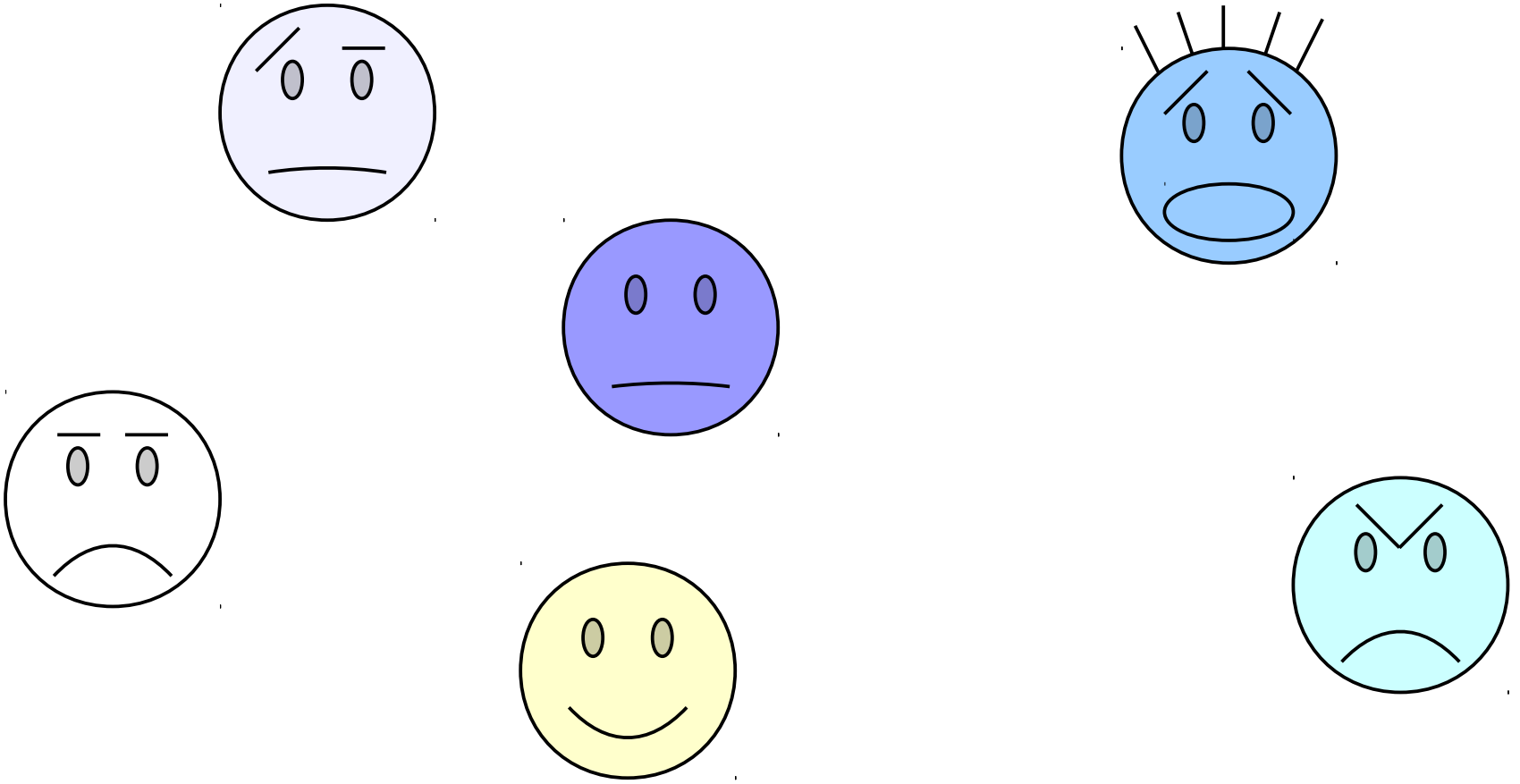
Since $Smiling(x)$ is not true for any choice of x, this statement evaluates to false.

The Existential Quantifier



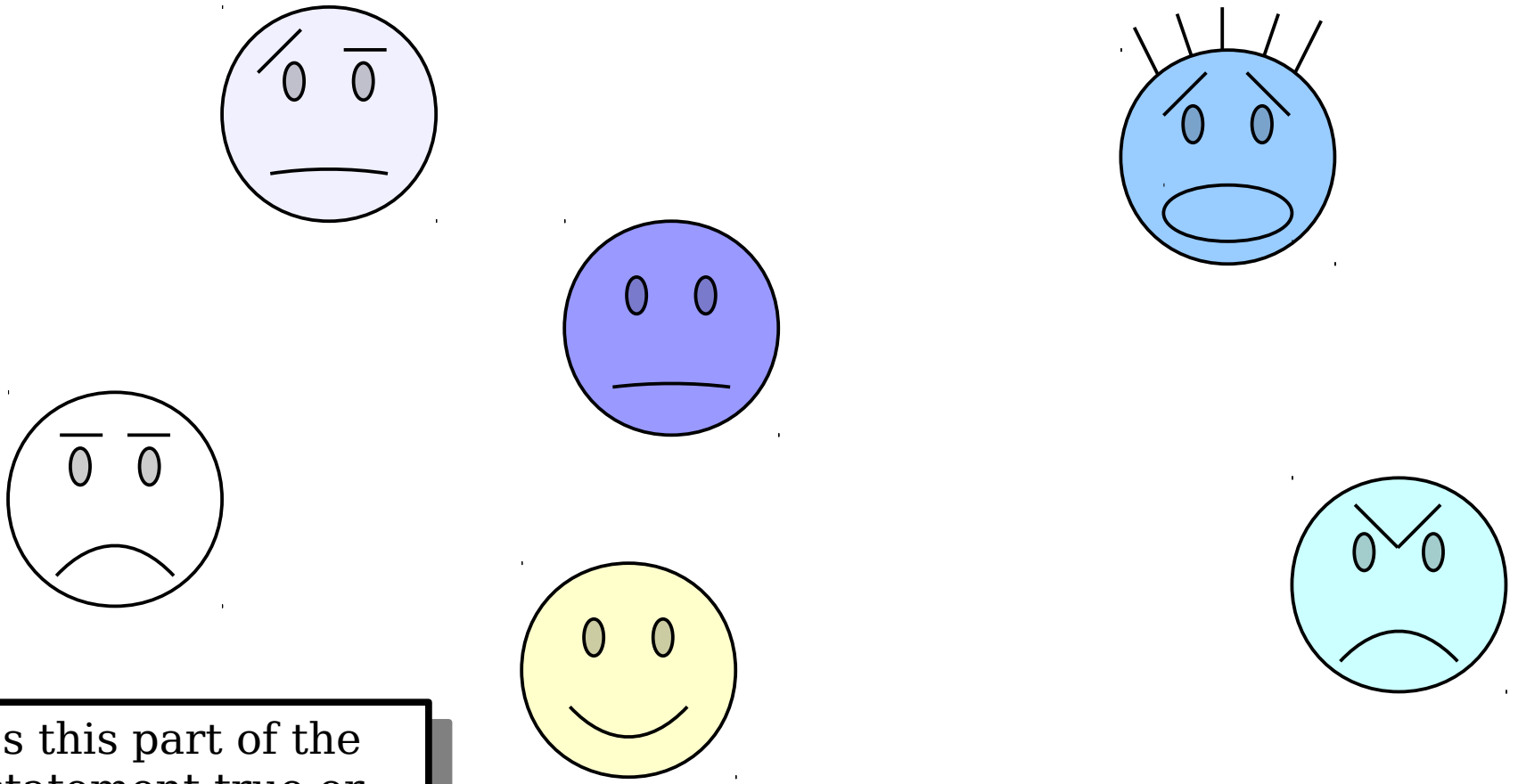
$(\exists x.\ \mathit{Smiling}(x)) \rightarrow (\exists y.\ \mathit{WearingHat}(y))$

The Existential Quantifier



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

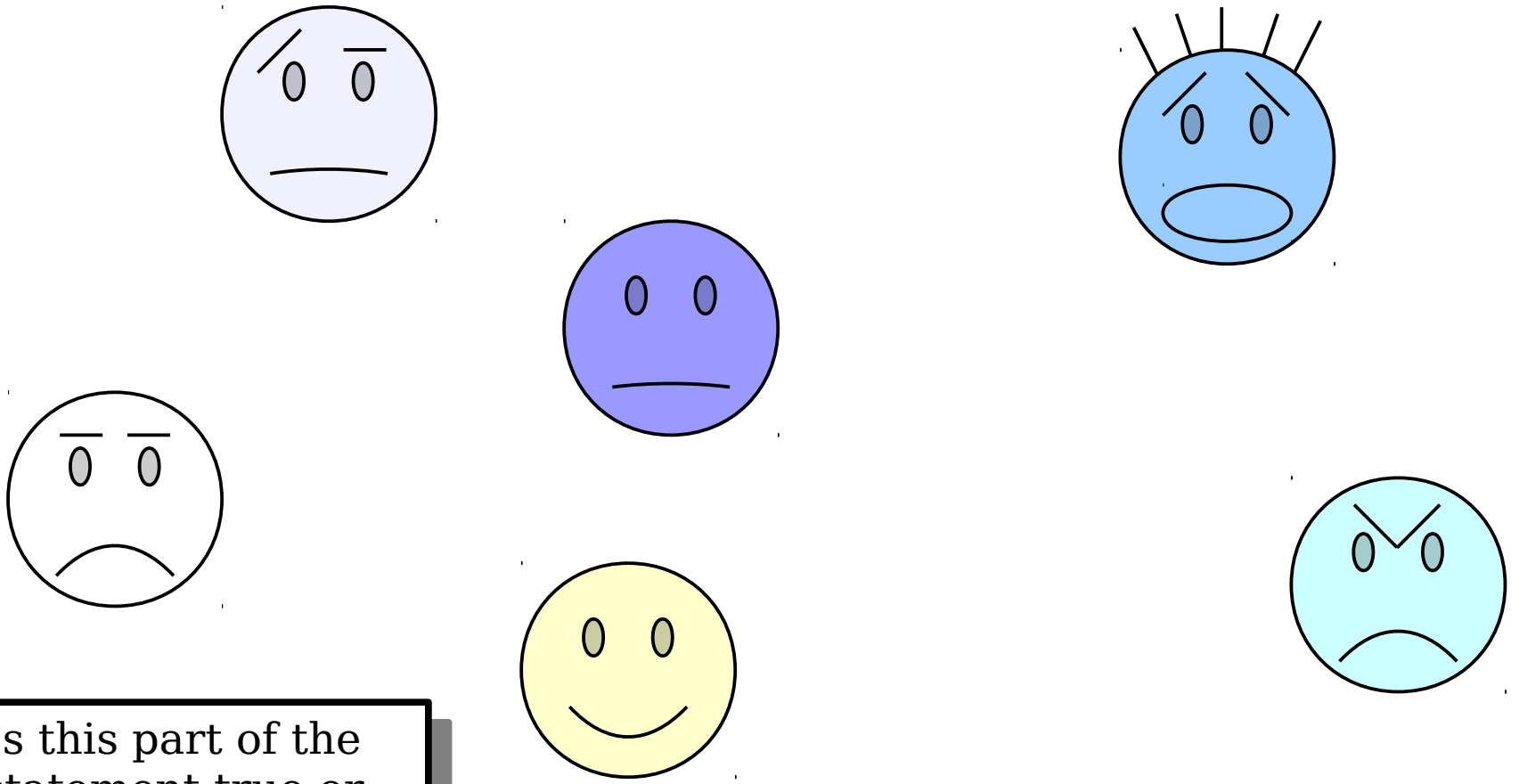
The Existential Quantifier



Is this part of the statement true or false?

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

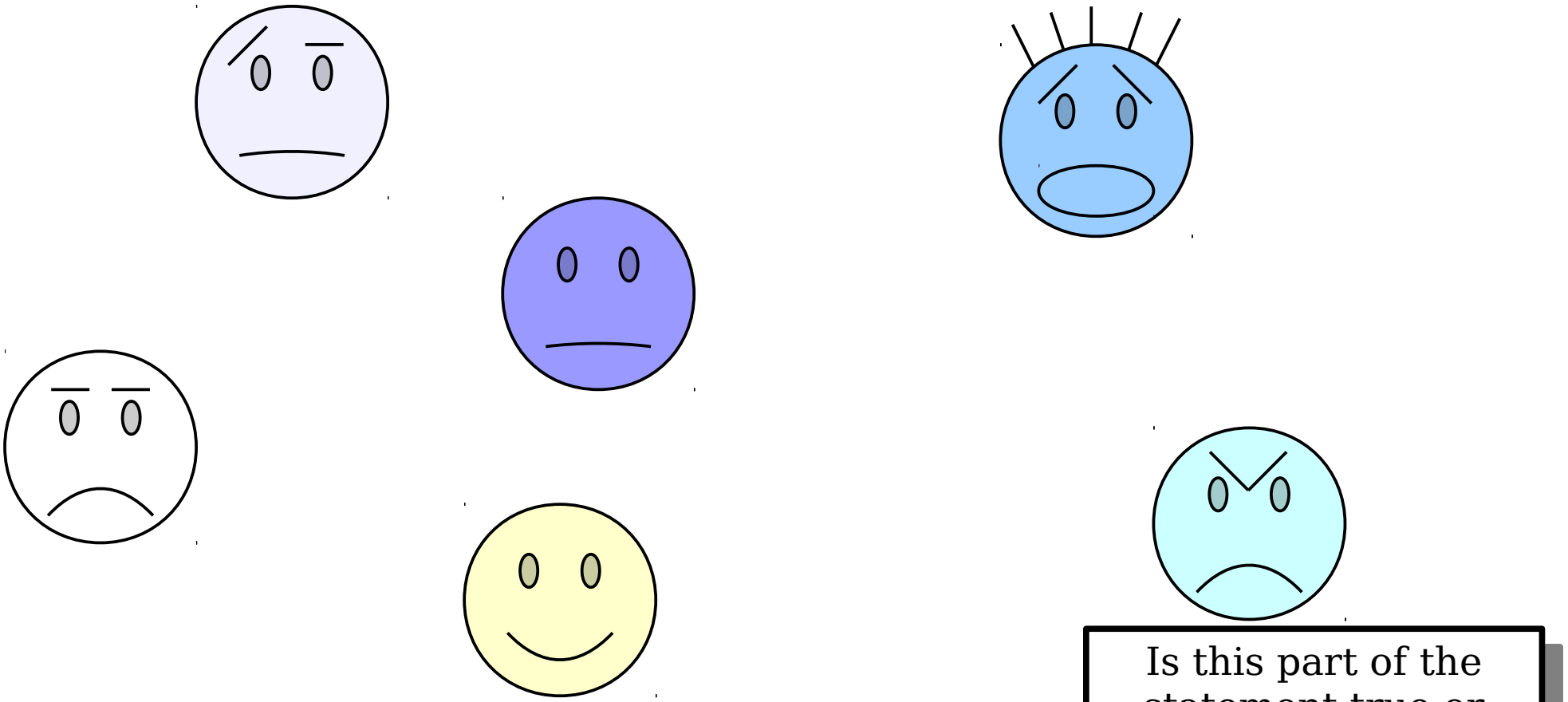
The Existential Quantifier



Is this part of the statement true or false?

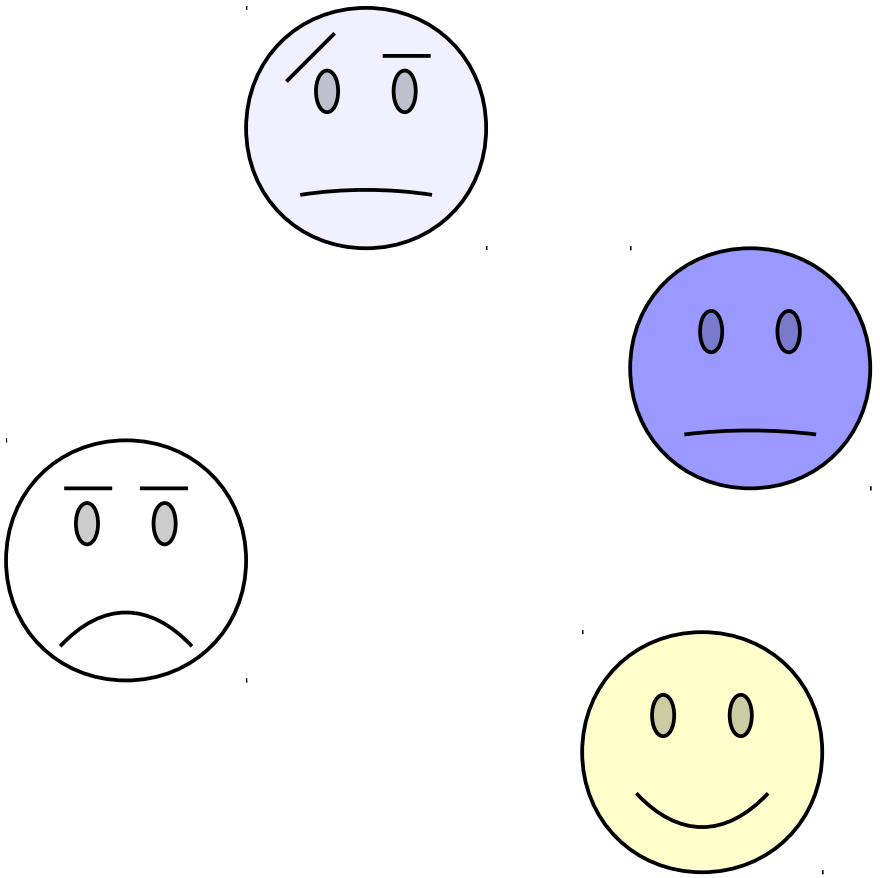
$(\exists x.\ \mathit{Smiling}(x)) \rightarrow (\exists y.\ \mathit{WearingHat}(y))$

The Existential Quantifier



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

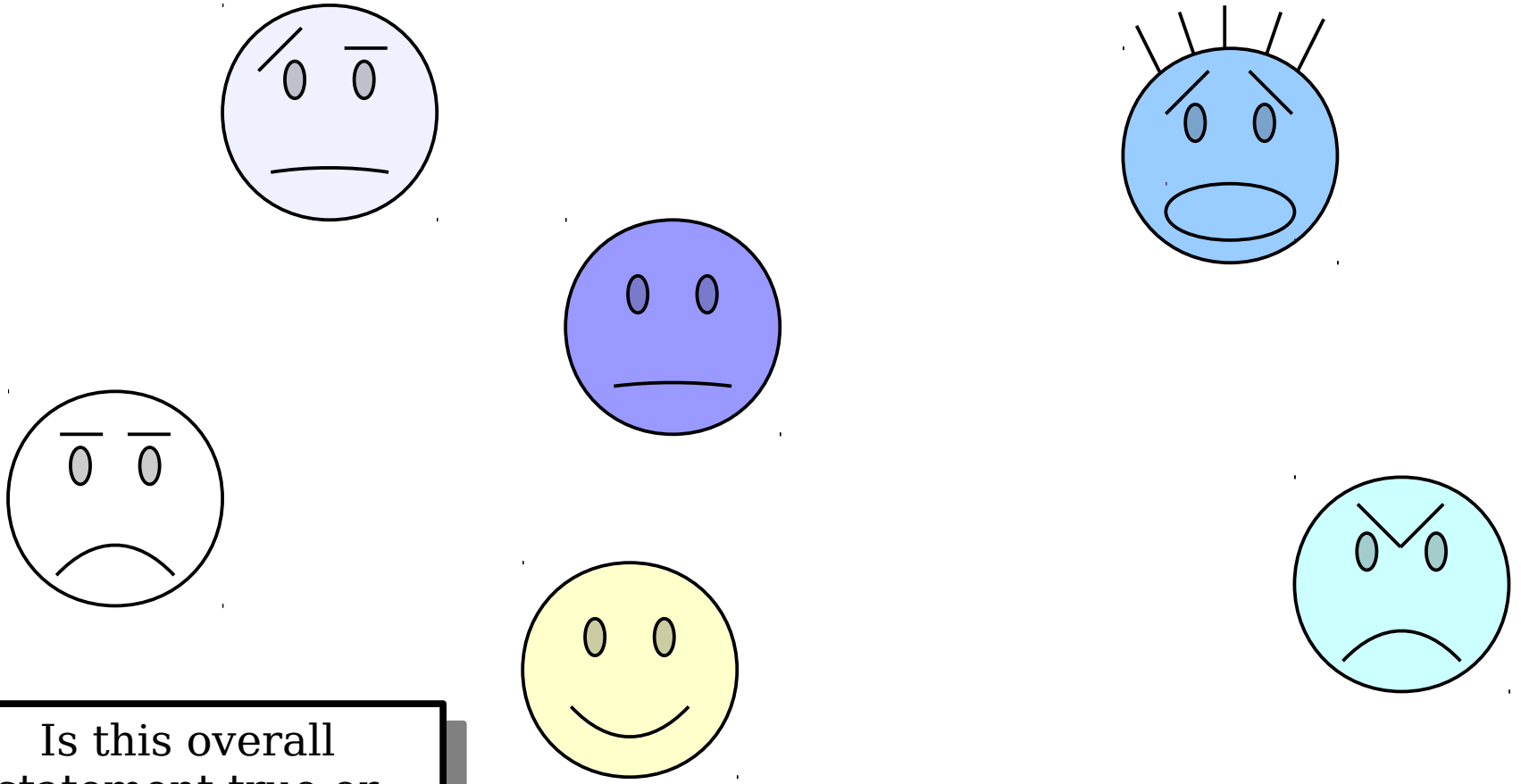
The Existential Quantifier



Is this part of the statement true or false?

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

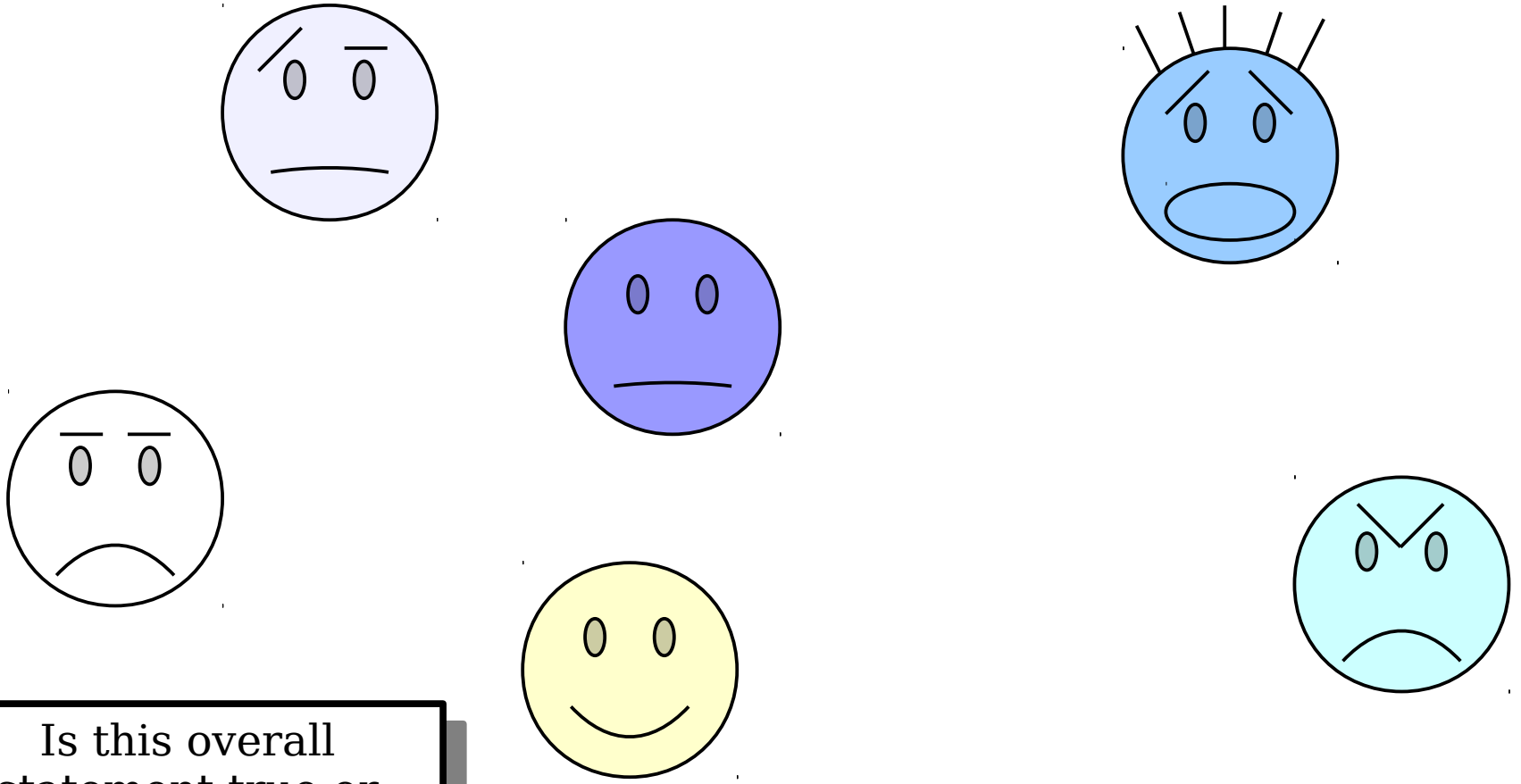
The Existential Quantifier



Is this overall statement true or false?

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

The Existential Quantifier



Is this overall statement true or false?

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

Fun with Edge Cases

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

Fun with Edge Cases

Existentially-quantified statements are false in an empty world, since nothing exists, period!

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

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.

$$(\exists x.\ \mathit{Loves}(\mathit{You}, x)) \wedge (\exists y.\ \mathit{Loves}(y, \mathit{You}))$$

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.

$$(\exists x.\ \mathit{Loves}(\mathit{You}, x)) \wedge (\exists y.\ \mathit{Loves}(y, \mathit{You}))$$

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.

$$(\exists x.\ \mathit{Loves}(\mathit{You}, x)) \wedge (\exists y.\ \mathit{Loves}(y, \mathit{You}))$$

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.

$$(\exists x.\ \mathit{Loves}(\mathit{You}, x)) \wedge (\exists x.\ \mathit{Loves}(x, \mathit{You}))$$

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.

$$(\exists x.\ \mathit{Loves}(\mathit{You}, x)) \wedge (\exists x.\ \mathit{Loves}(x, \mathit{You}))$$

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, quantifiers have precedence just below ¬.
- The statement

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

  is parsed like this:

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

- 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:

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

“For any natural number n,
n is even if and only if n^2 is even”

“For any natural number n,
n is even if and only if n^2 is even”

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

"For any natural number n,
n is even if and only if 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

  $$\forall x.\ \textit{some-formula}$$

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

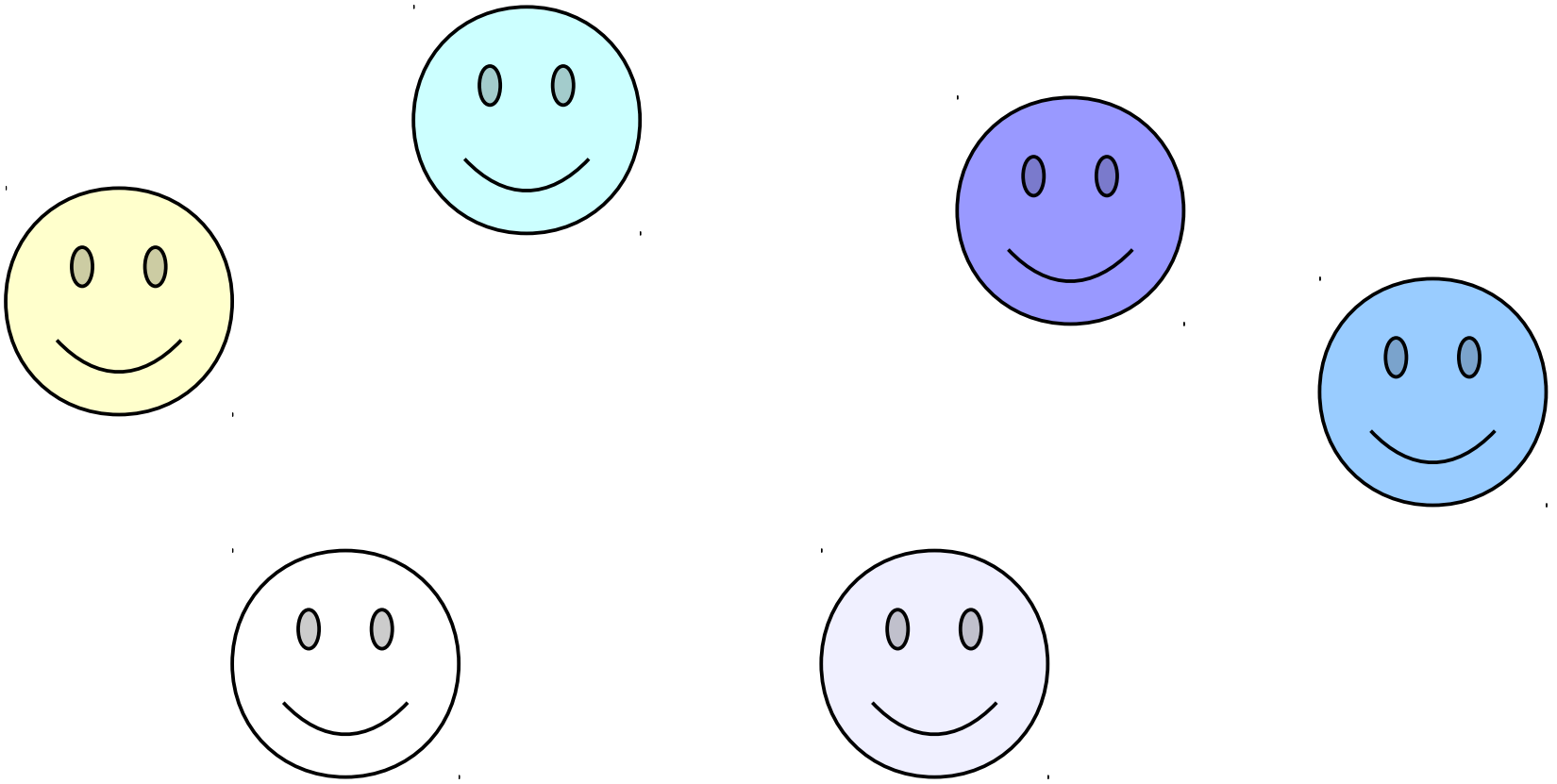
- Examples:

  $\forall p.\ (\mathit{Puppy}(p) \to \mathit{Cute}(p))$

  $\forall a.\ (\mathit{EatsPlants}(a) \vee \mathit{EatsAnimals}(a))$

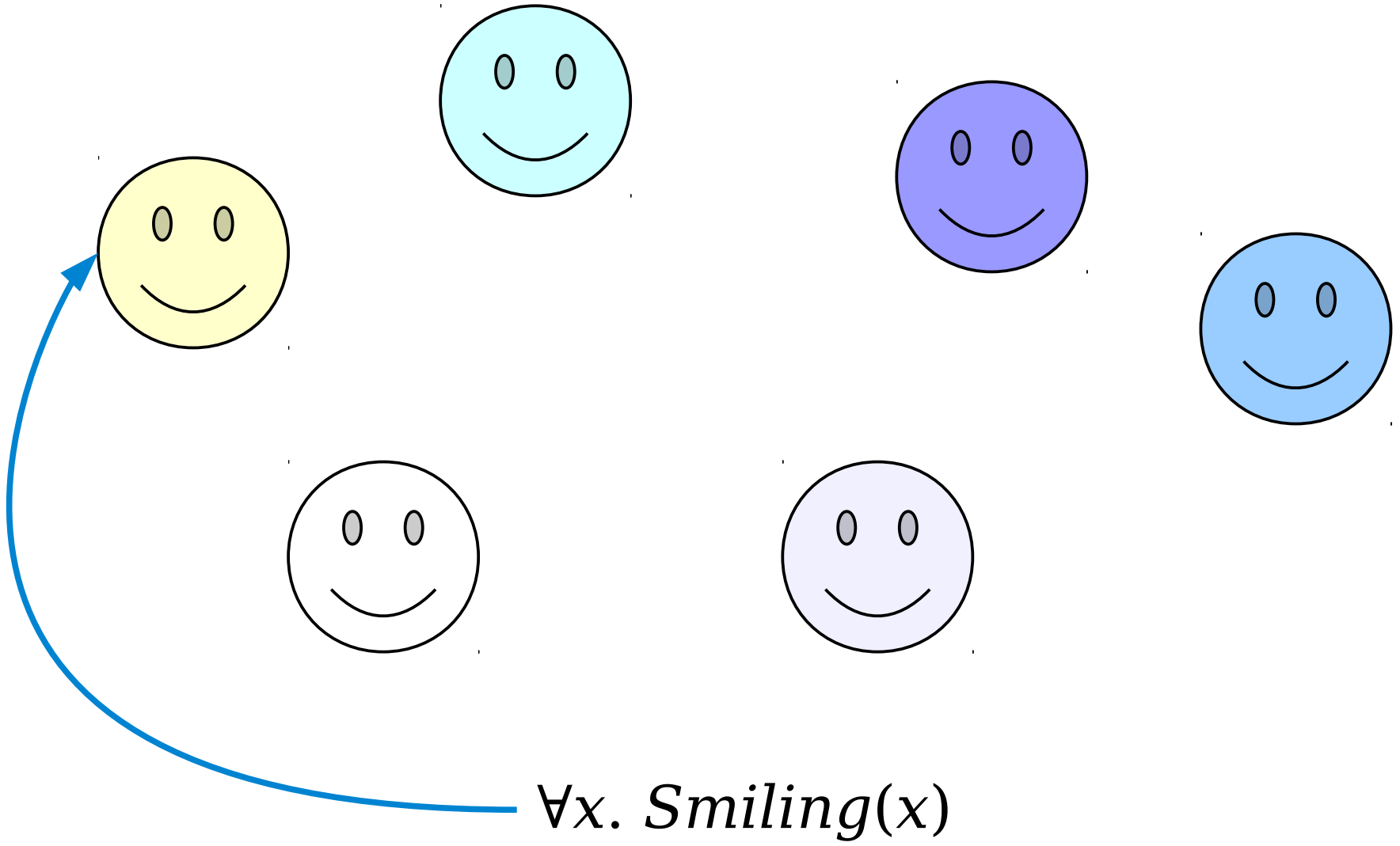
  $\mathit{Tallest}(\mathit{SultanKösen}) \to$
  $\quad \forall x.\ (\mathit{SultanKösen} \neq x \to \mathit{ShorterThan}(x, \mathit{SultanKösen}))$

The Universal Quantifier

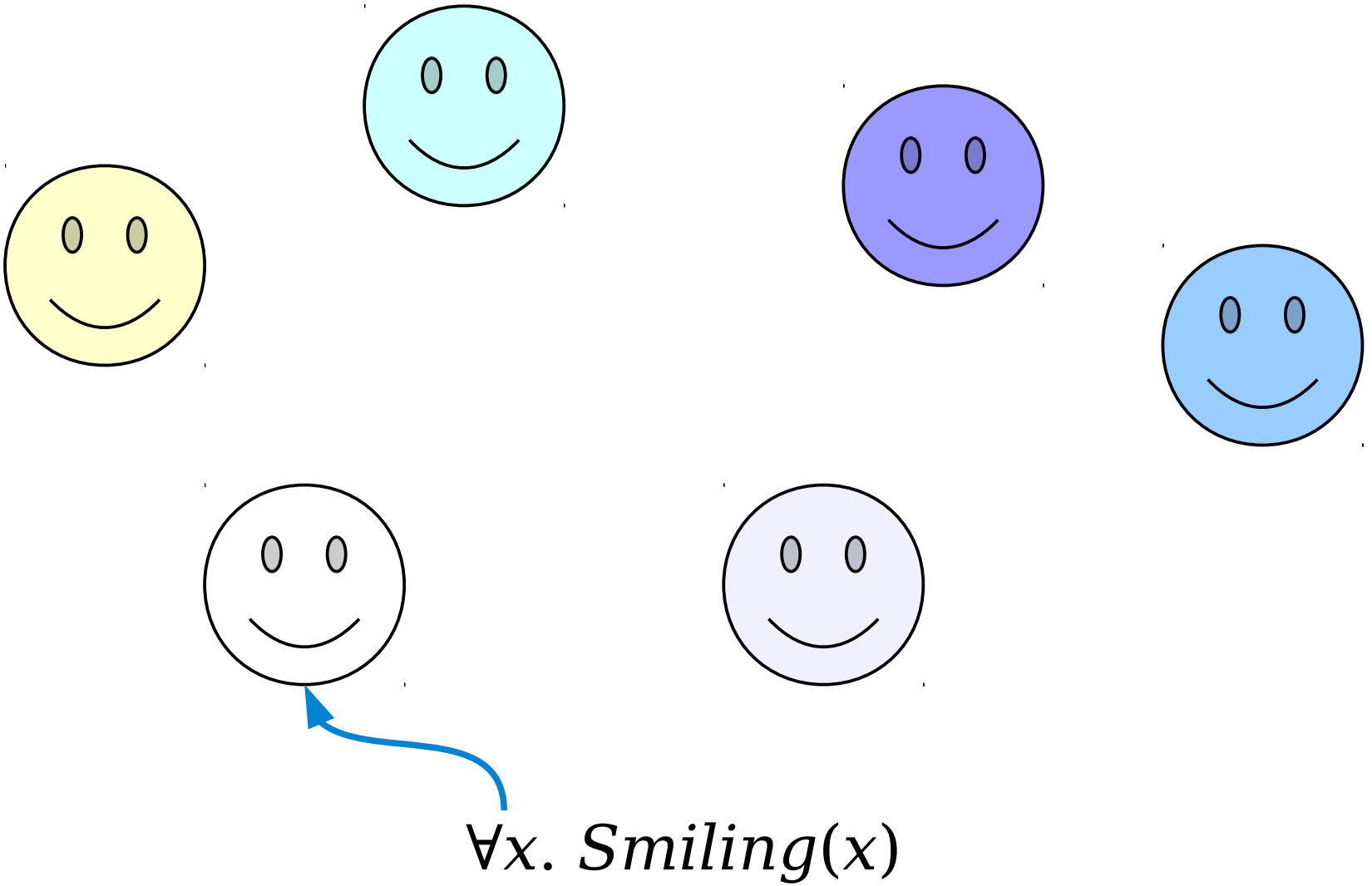


$\forall x.\ \mathit{Smiling}(x)$

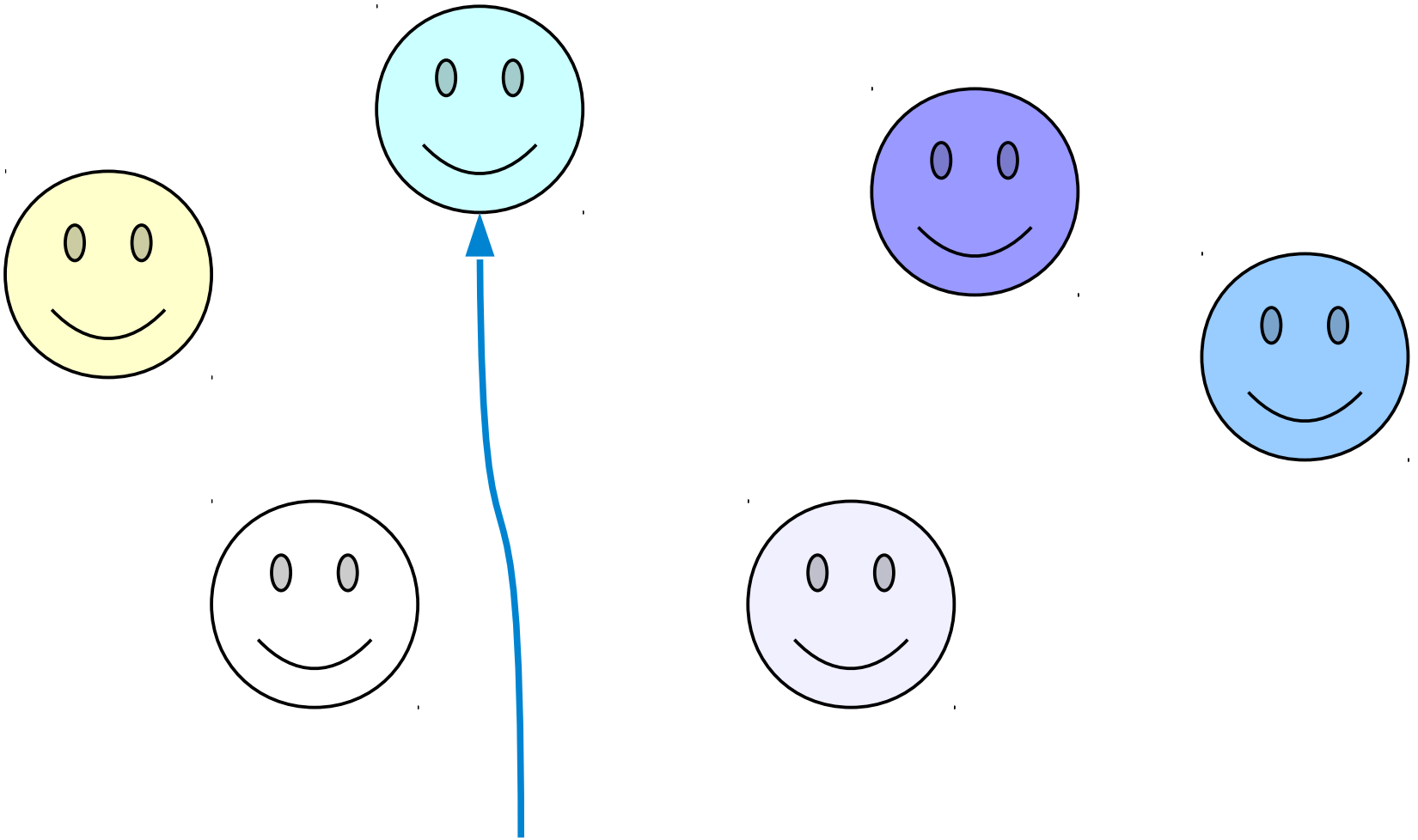
The Universal Quantifier



The Universal Quantifier

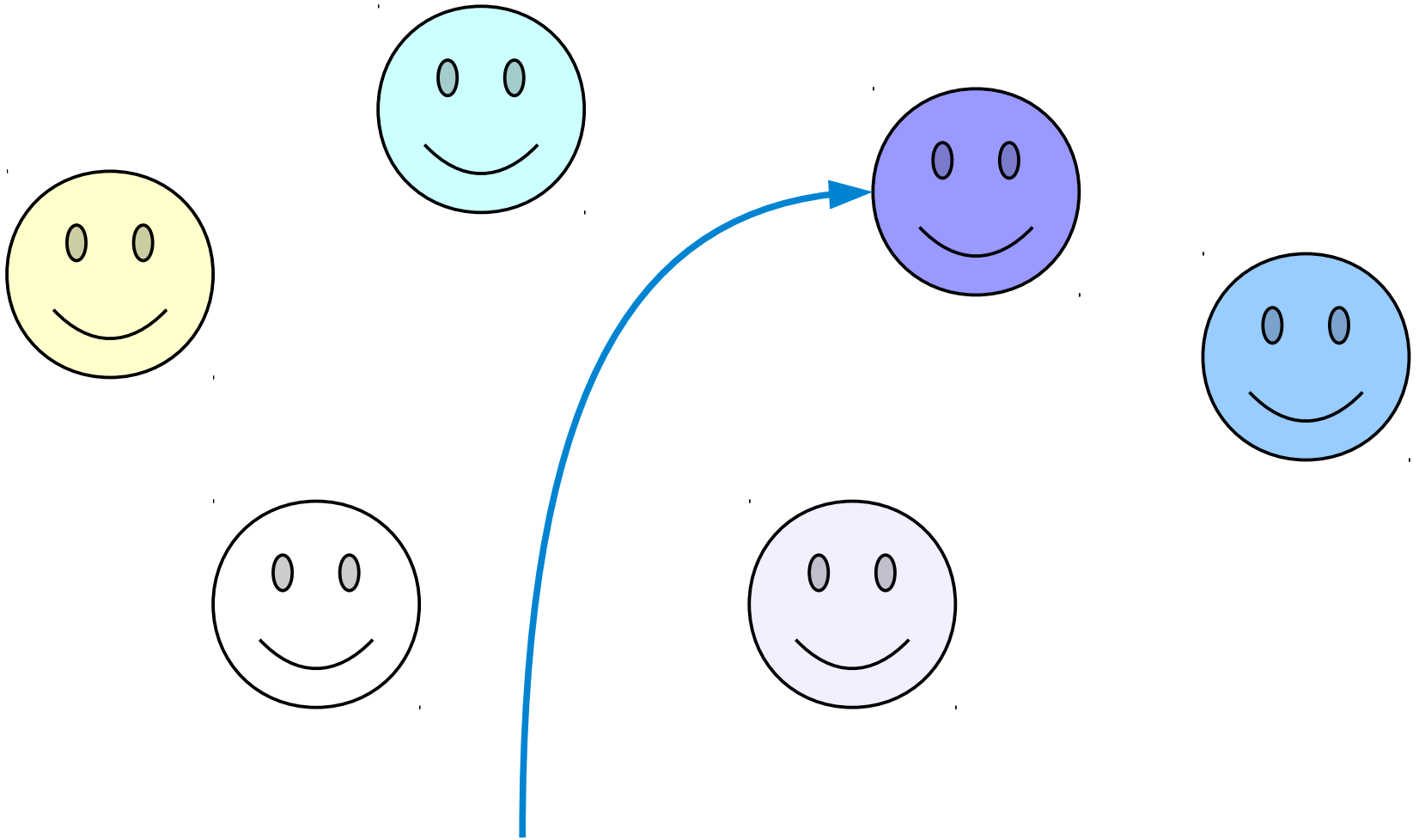


The Universal Quantifier

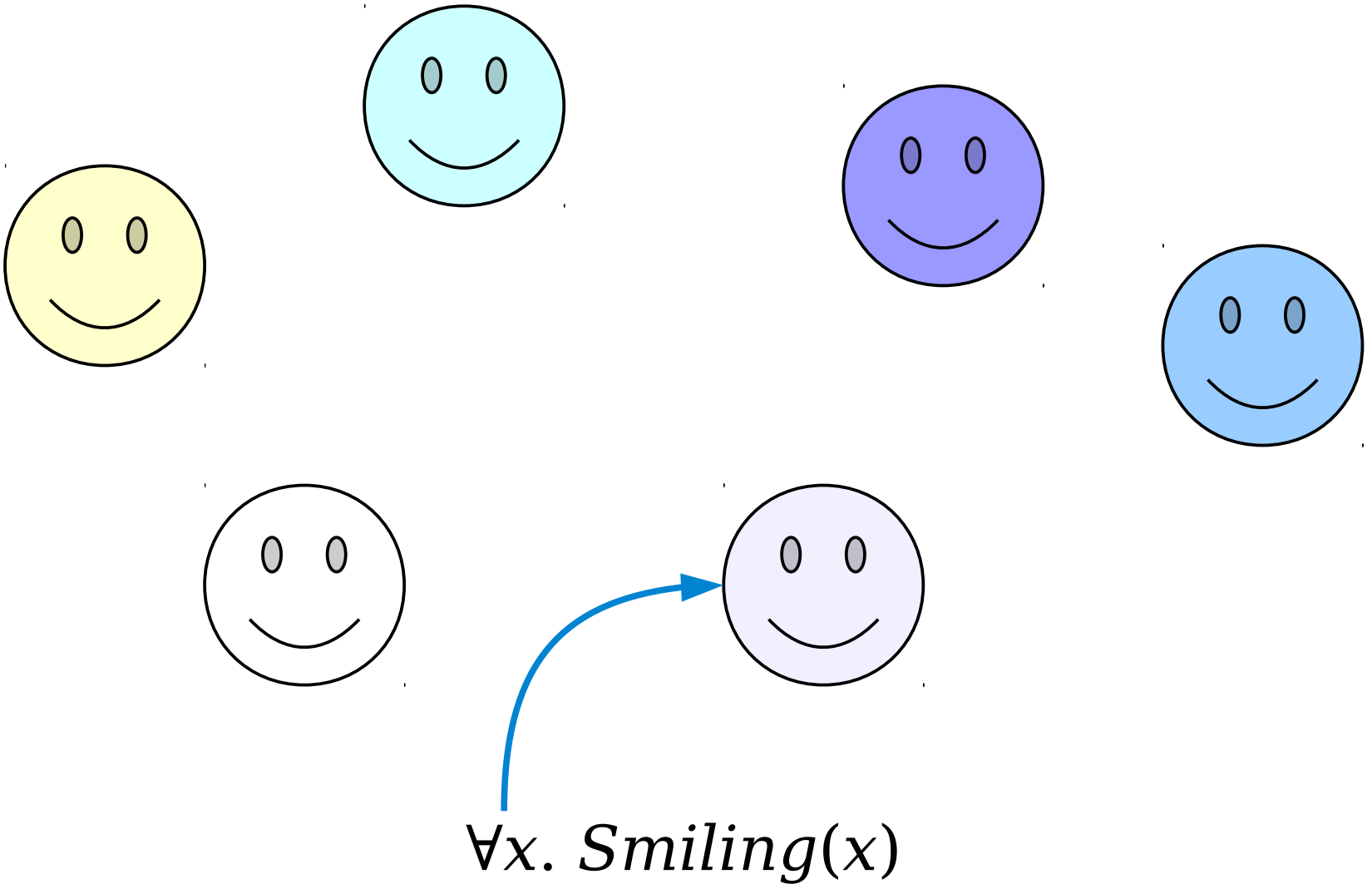


$\forall x.\ \mathit{Smiling}(x)$

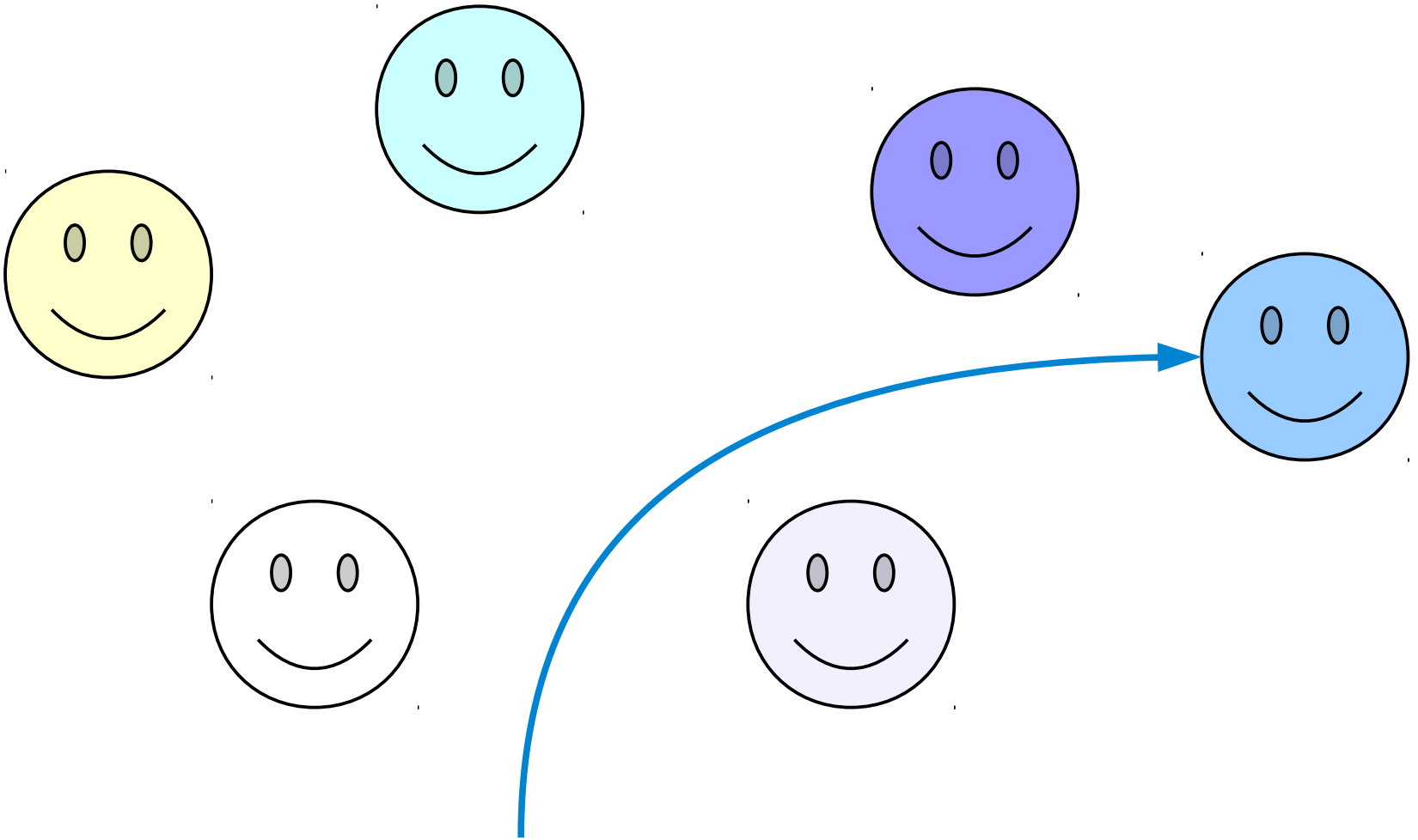
The Universal Quantifier



The Universal Quantifier

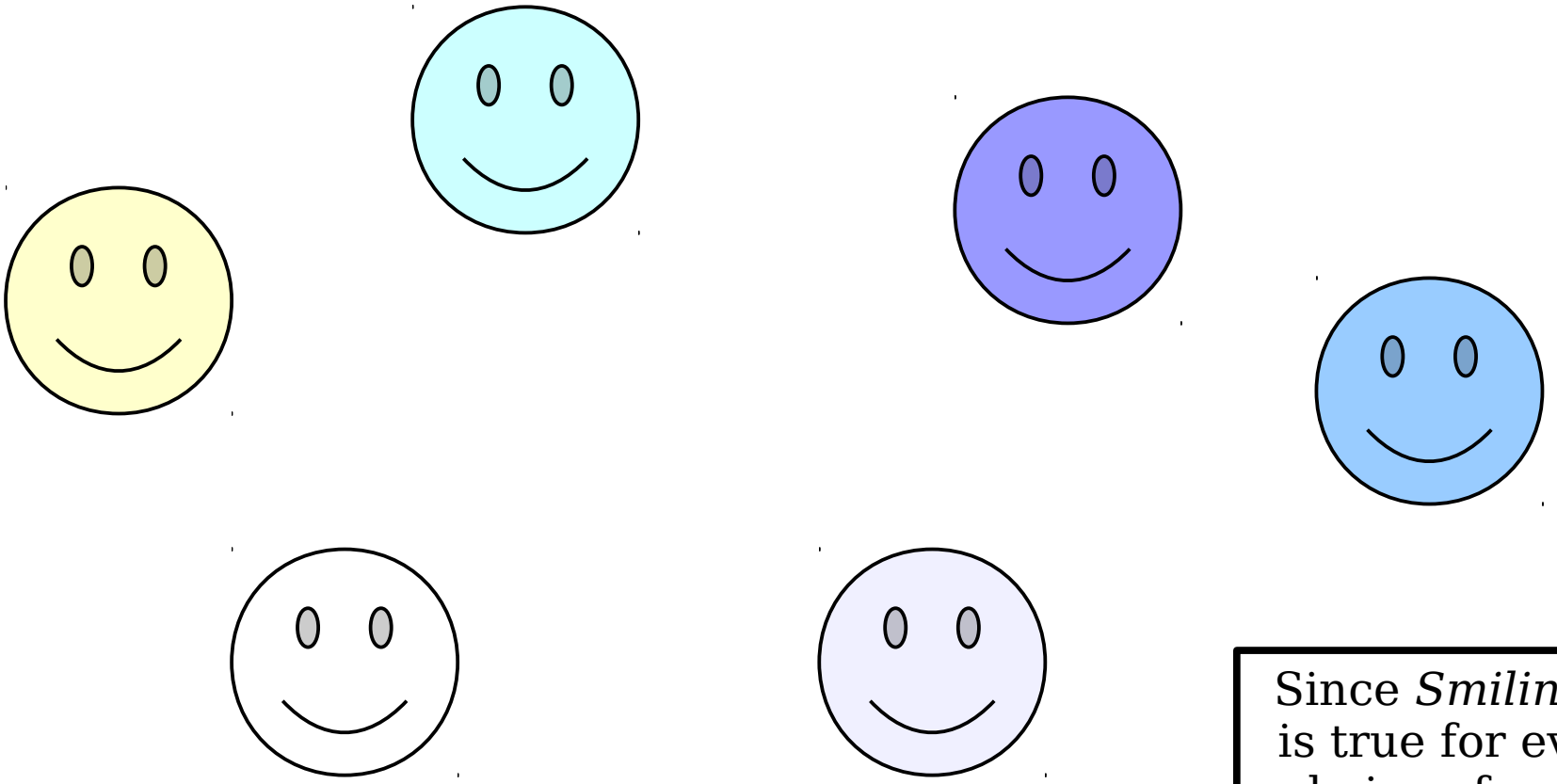


The Universal Quantifier



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

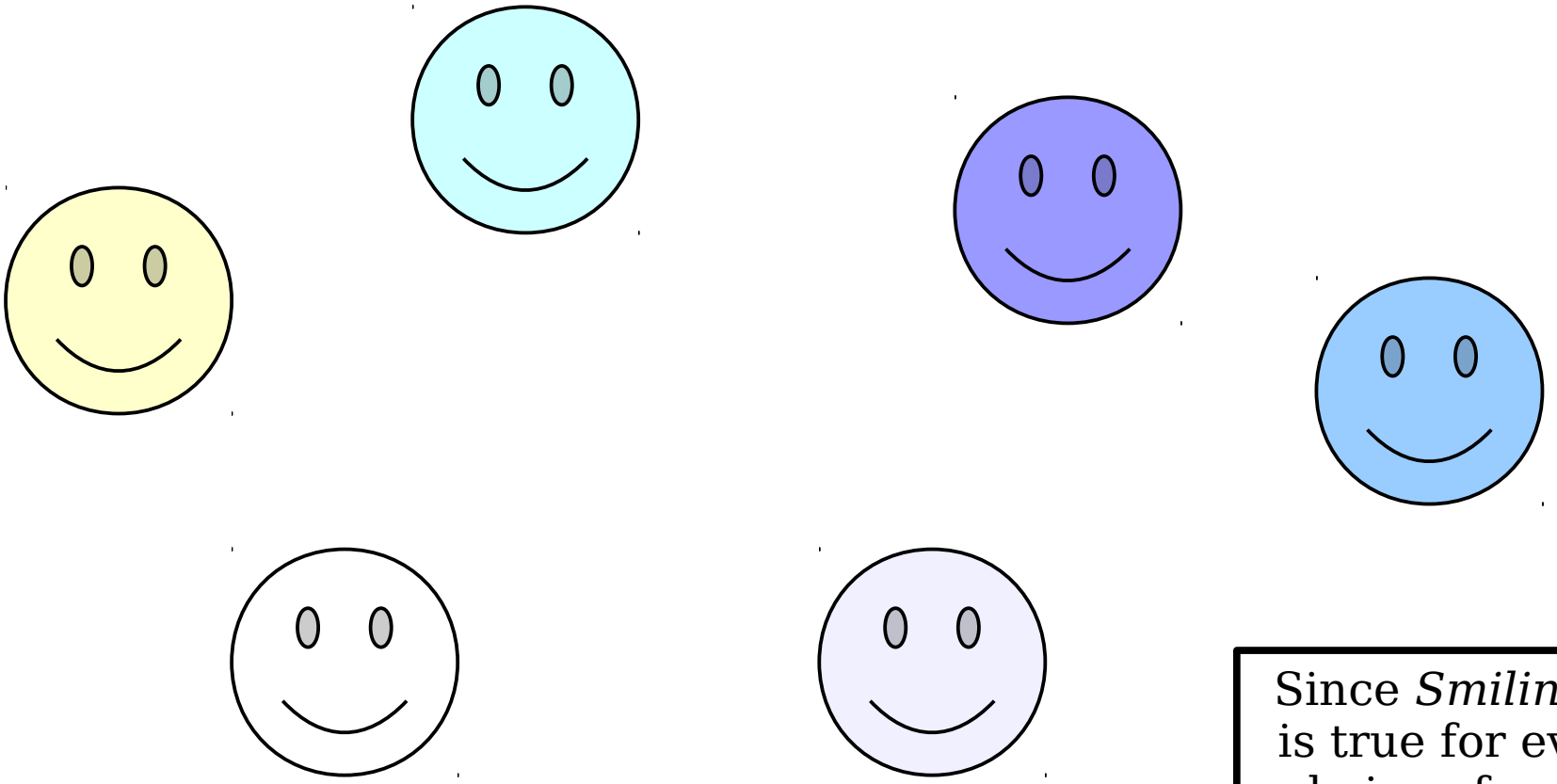
The Universal Quantifier



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

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

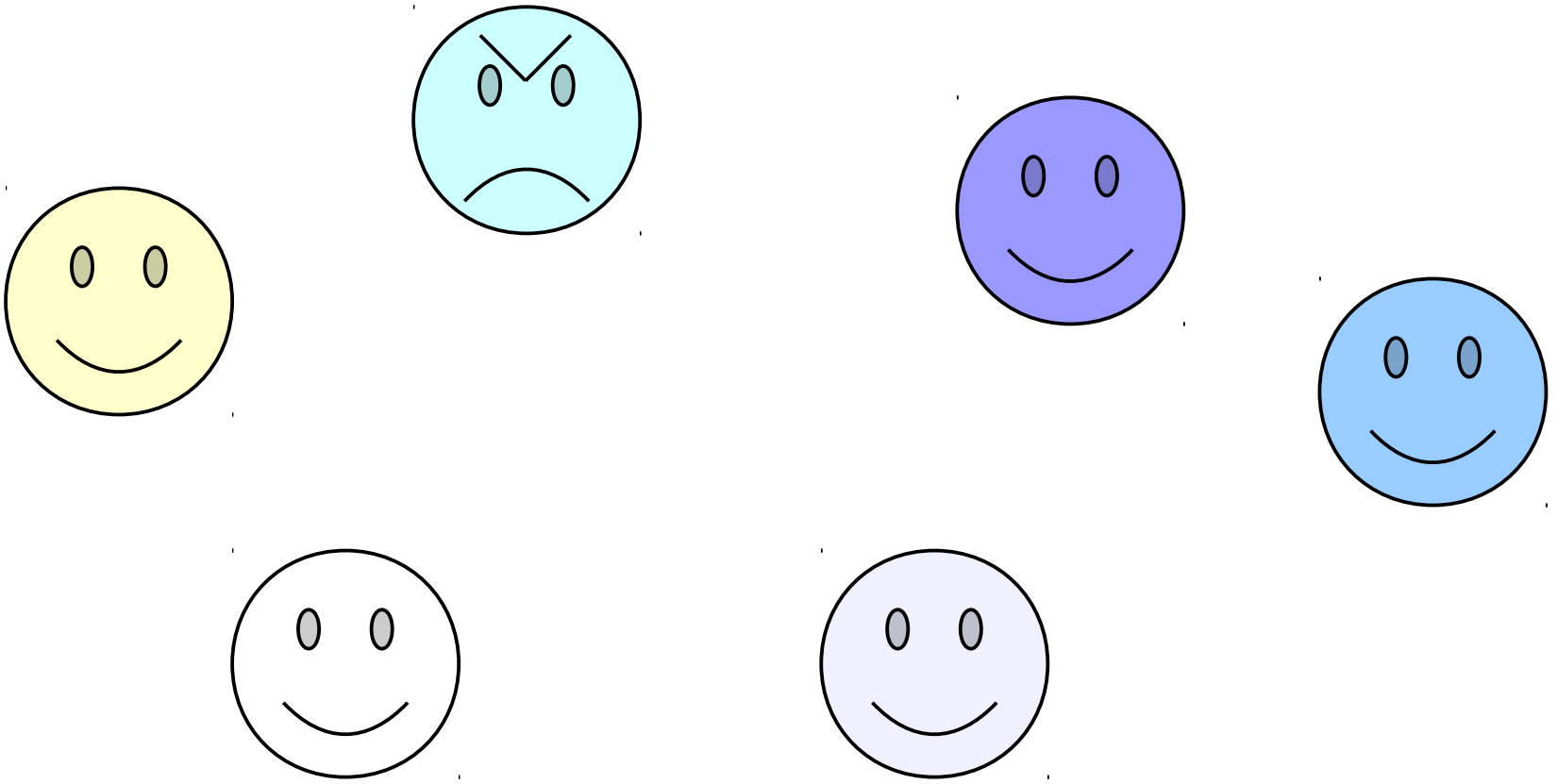
The Universal Quantifier



$\forall x.\ Smiling(x)$

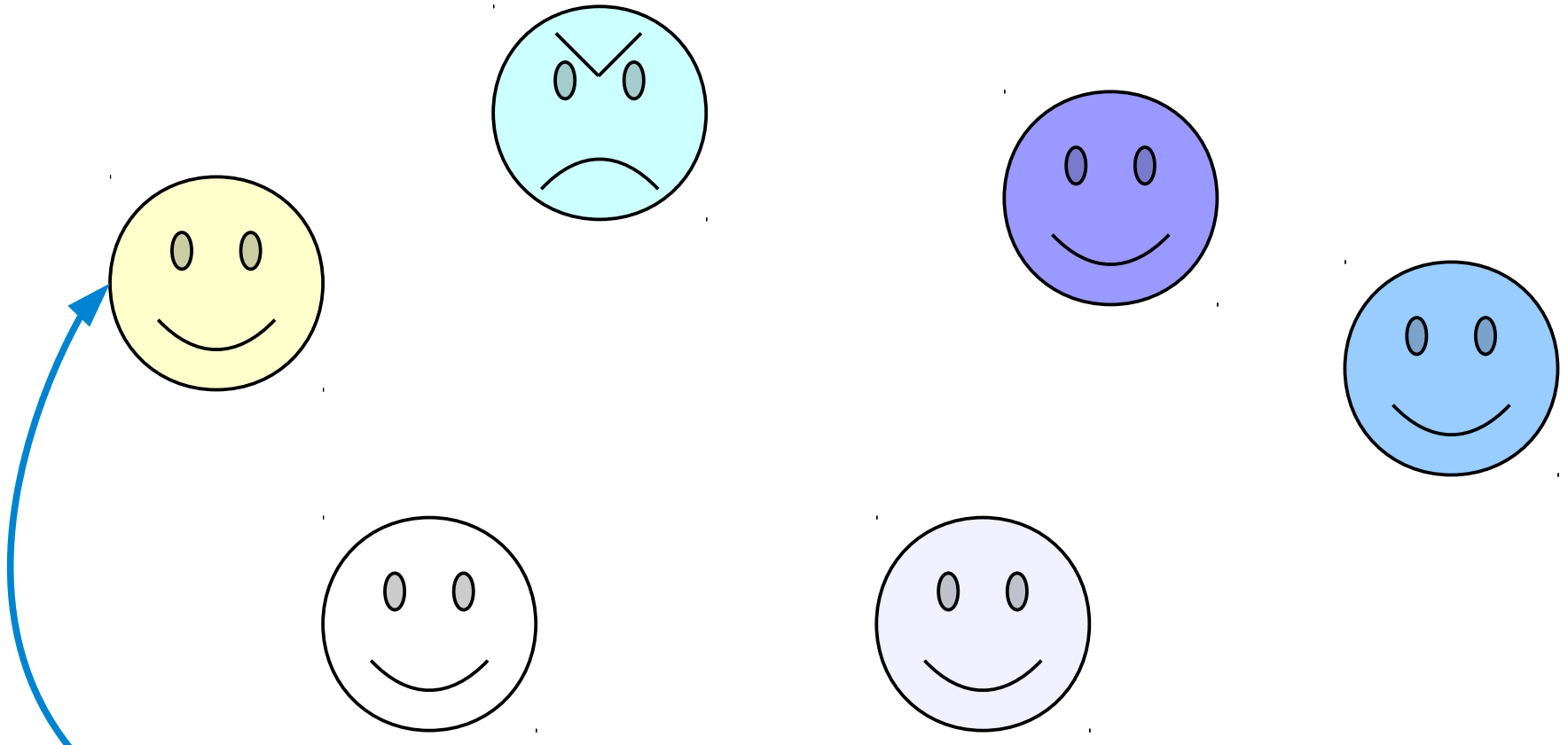
Since $Smiling(x)$ is true for every choice of x, this statement evaluates to true.

The Universal Quantifier



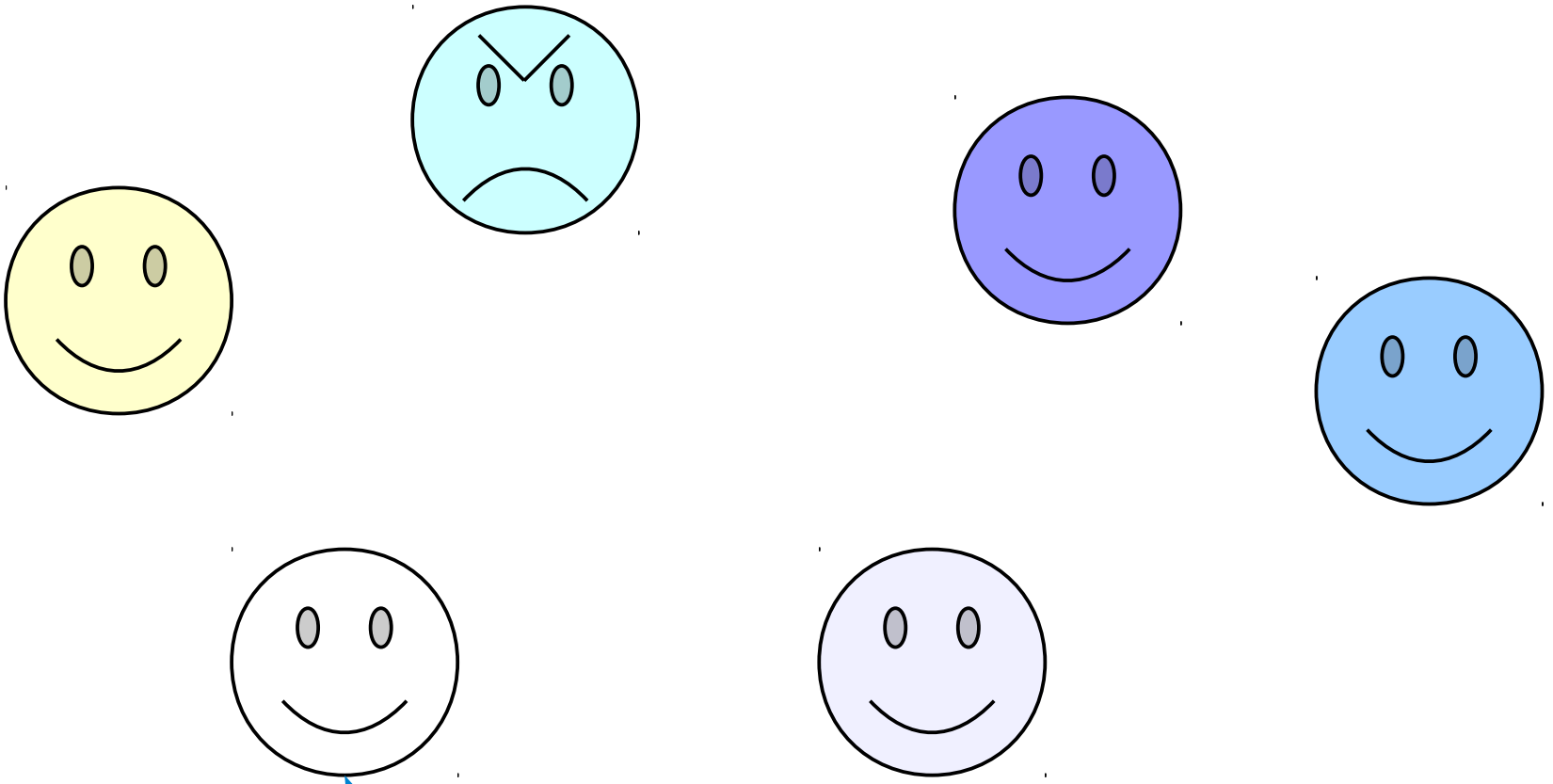
$\forall x.\ \mathit{Smiling}(x)$

The Universal Quantifier



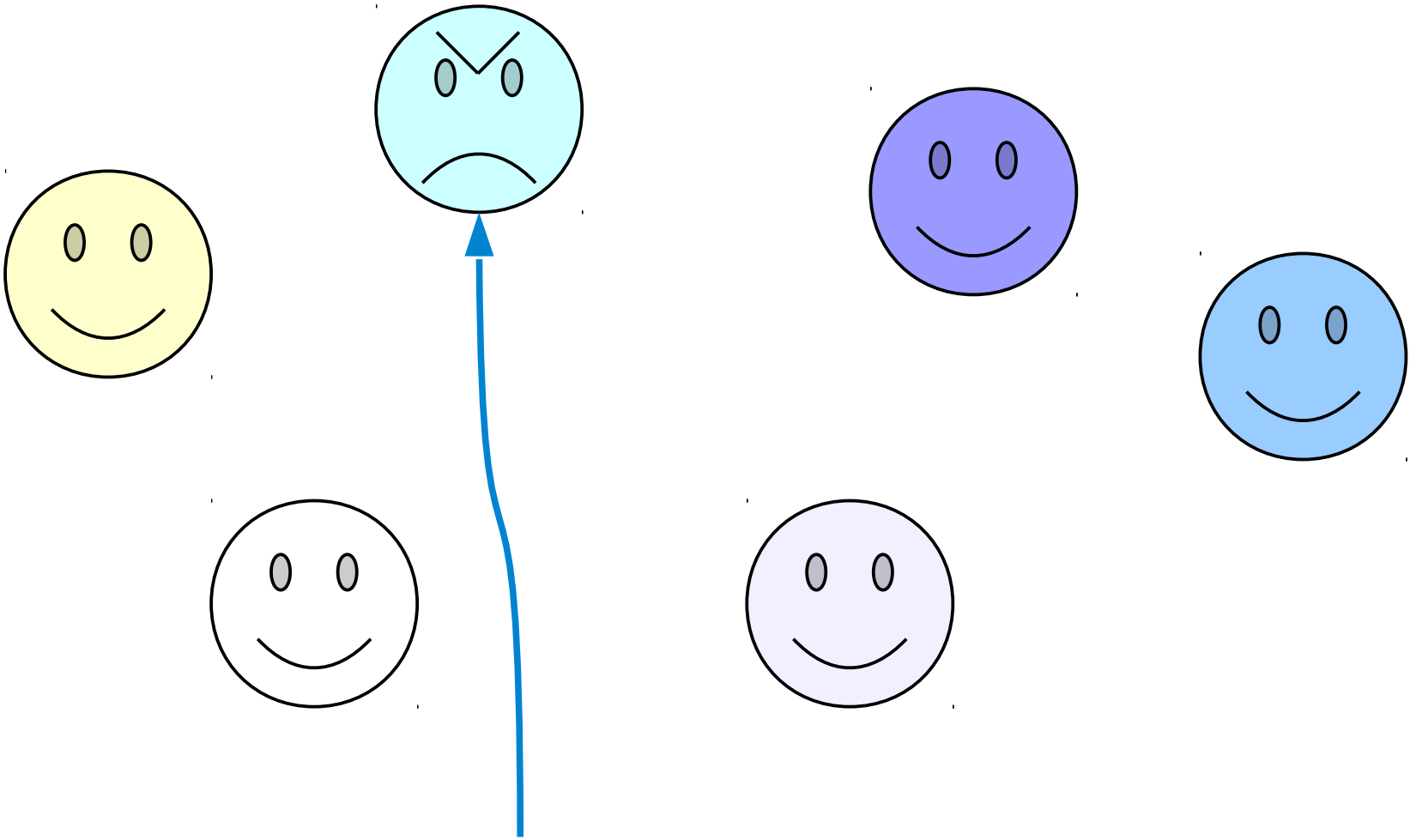
$\forall x.\ \mathit{Smiling}(x)$

The Universal Quantifier



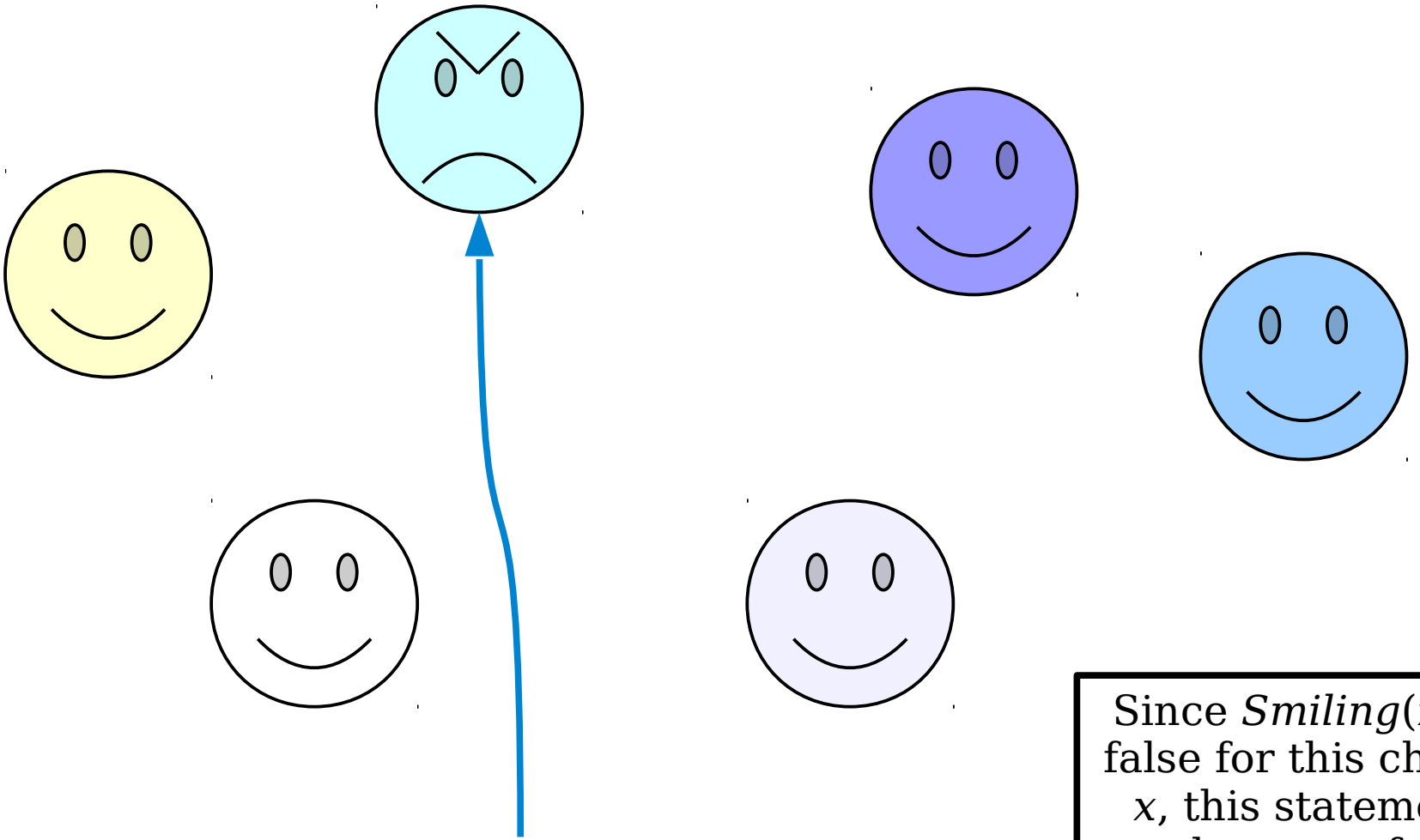
$\forall x.\ \mathit{Smiling}(x)$

The Universal Quantifier



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

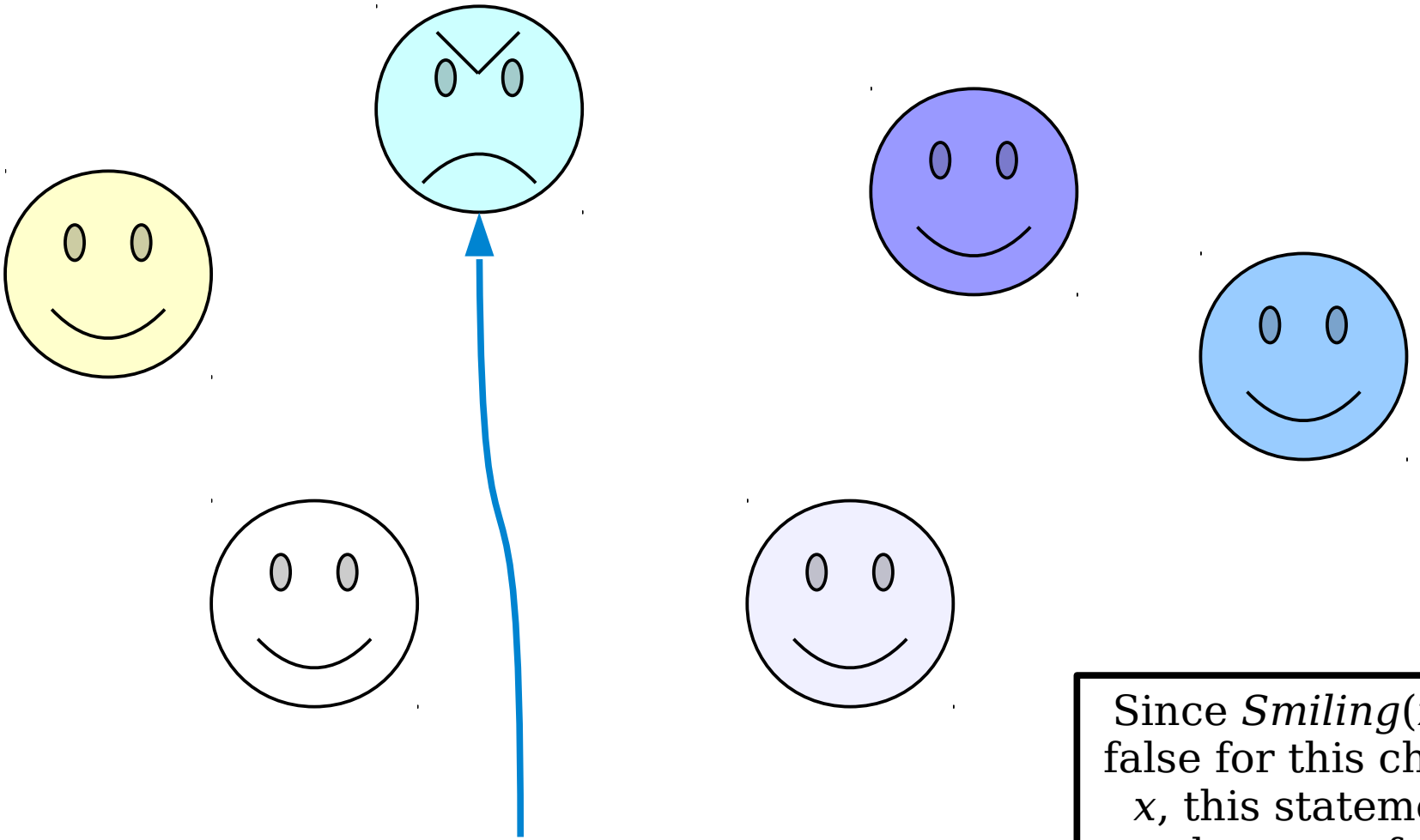
The Universal Quantifier



$\forall x.\ Smiling(x)$

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

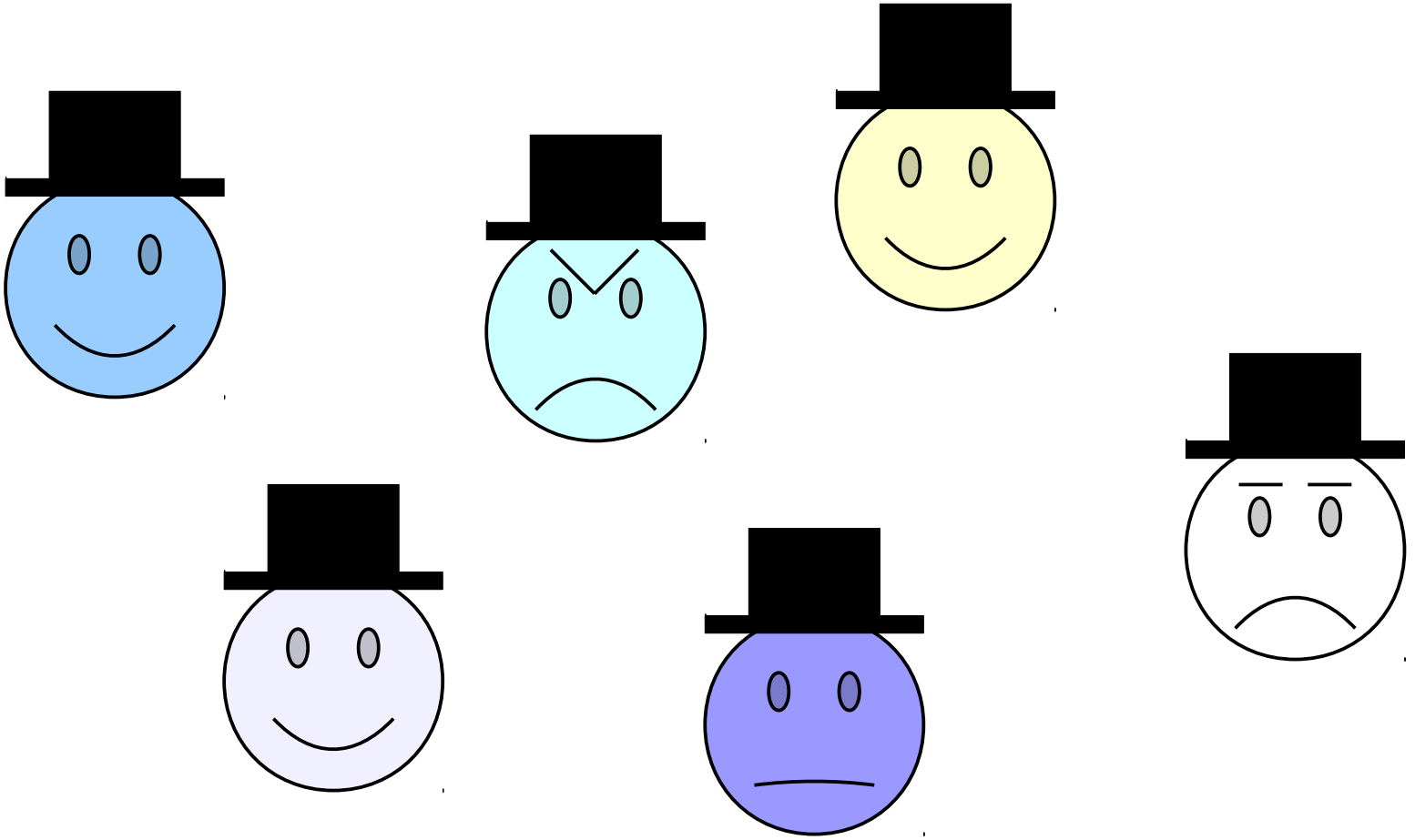
The Universal Quantifier



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

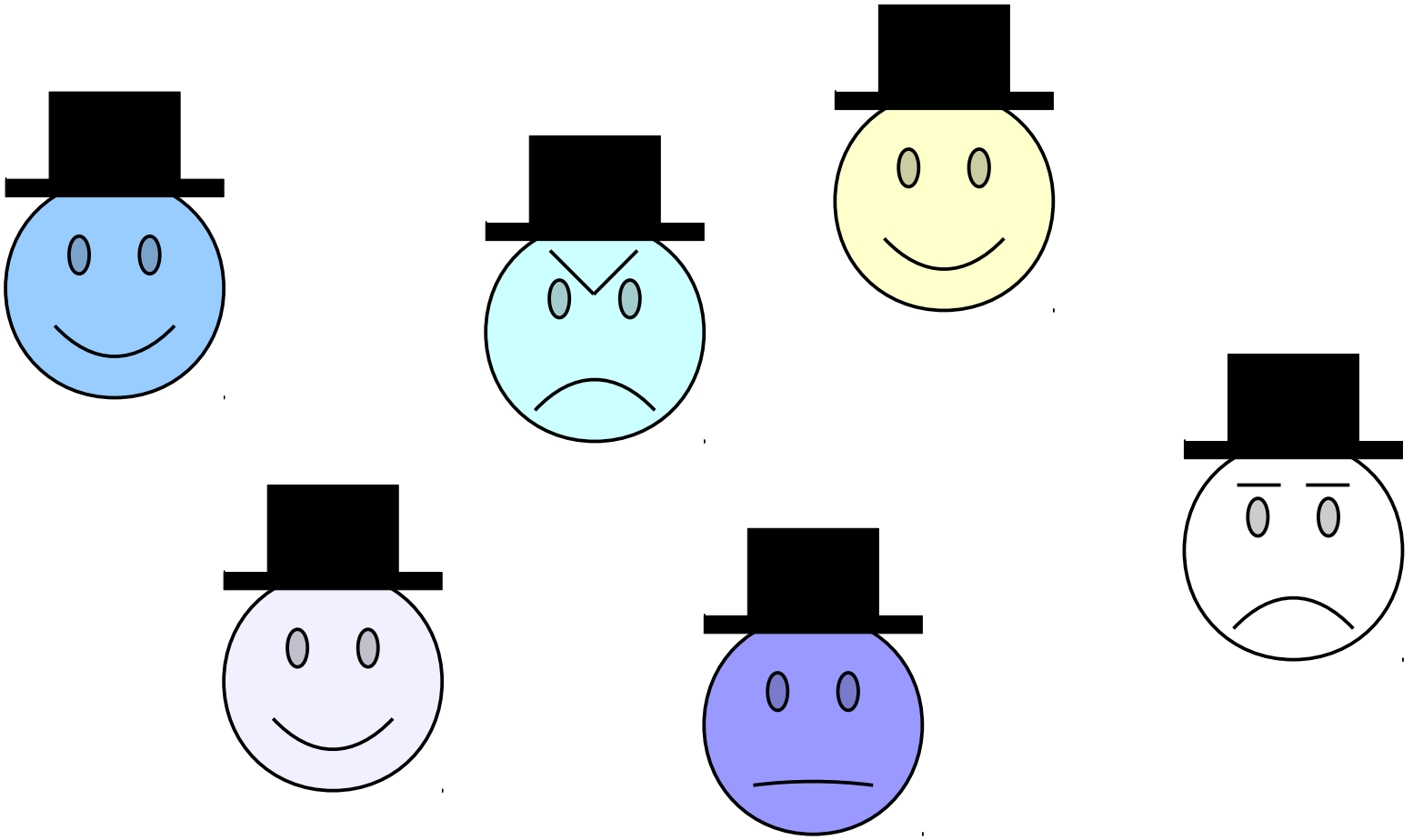
Since $Smiling(x)$ is false for this choice x, this statement evaluates to false.

The Universal Quantifier



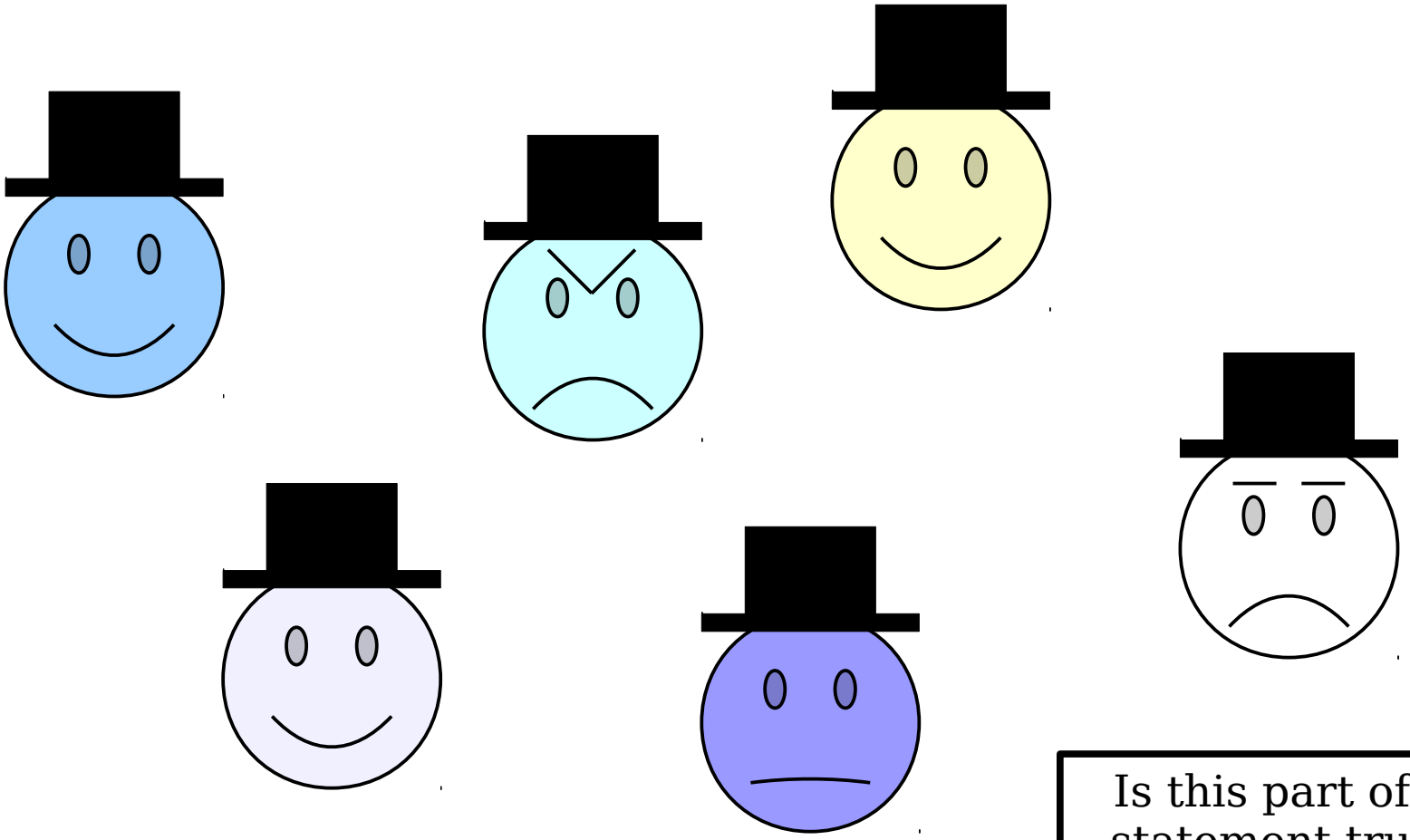
$(\forall x.\ \mathit{Smiling}(x)) \rightarrow (\forall y.\ \mathit{WearingHat}(y))$

The Universal Quantifier



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

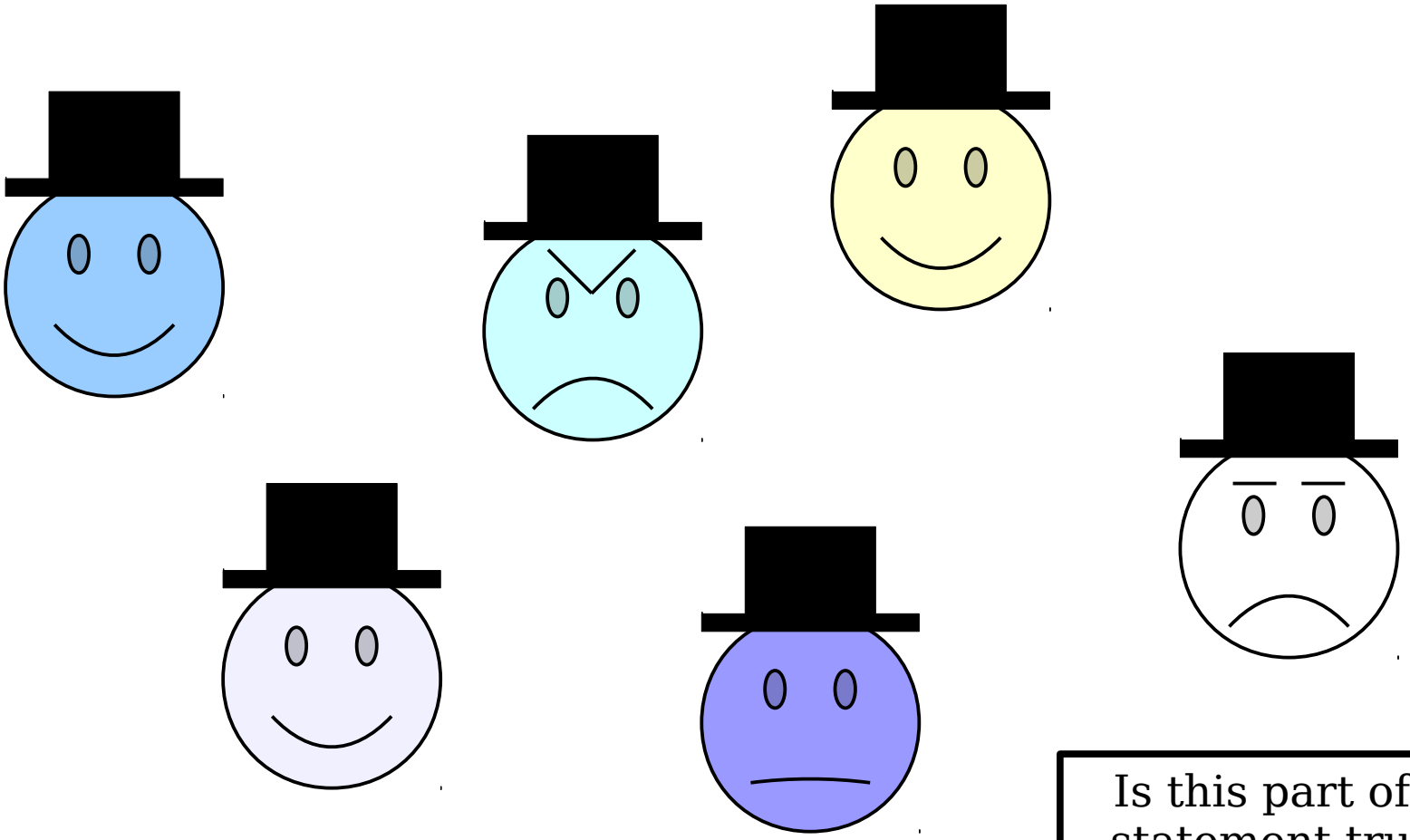
The Universal Quantifier



Is this part of the statement true or false?

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

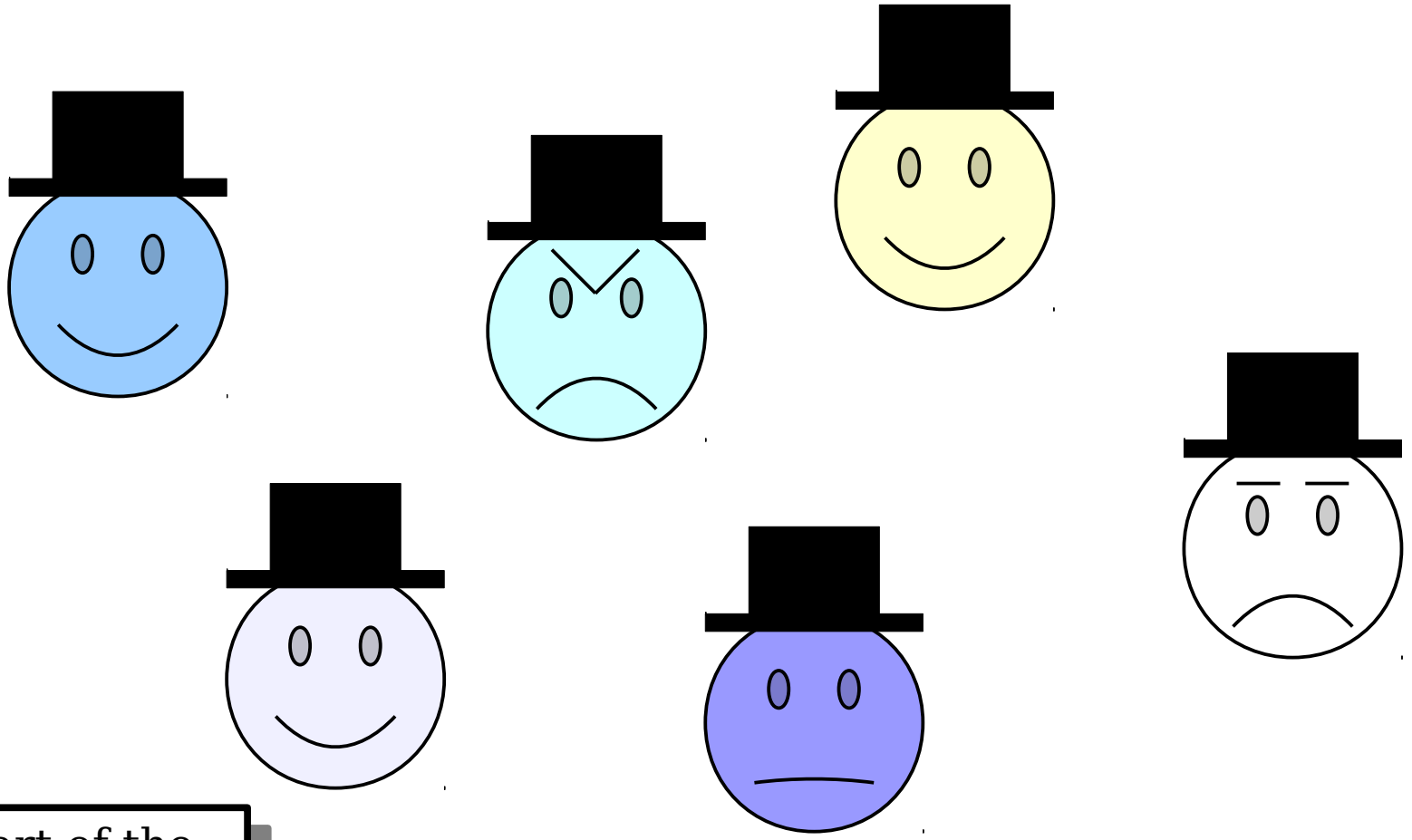
The Universal Quantifier



Is this part of the statement true or false?

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

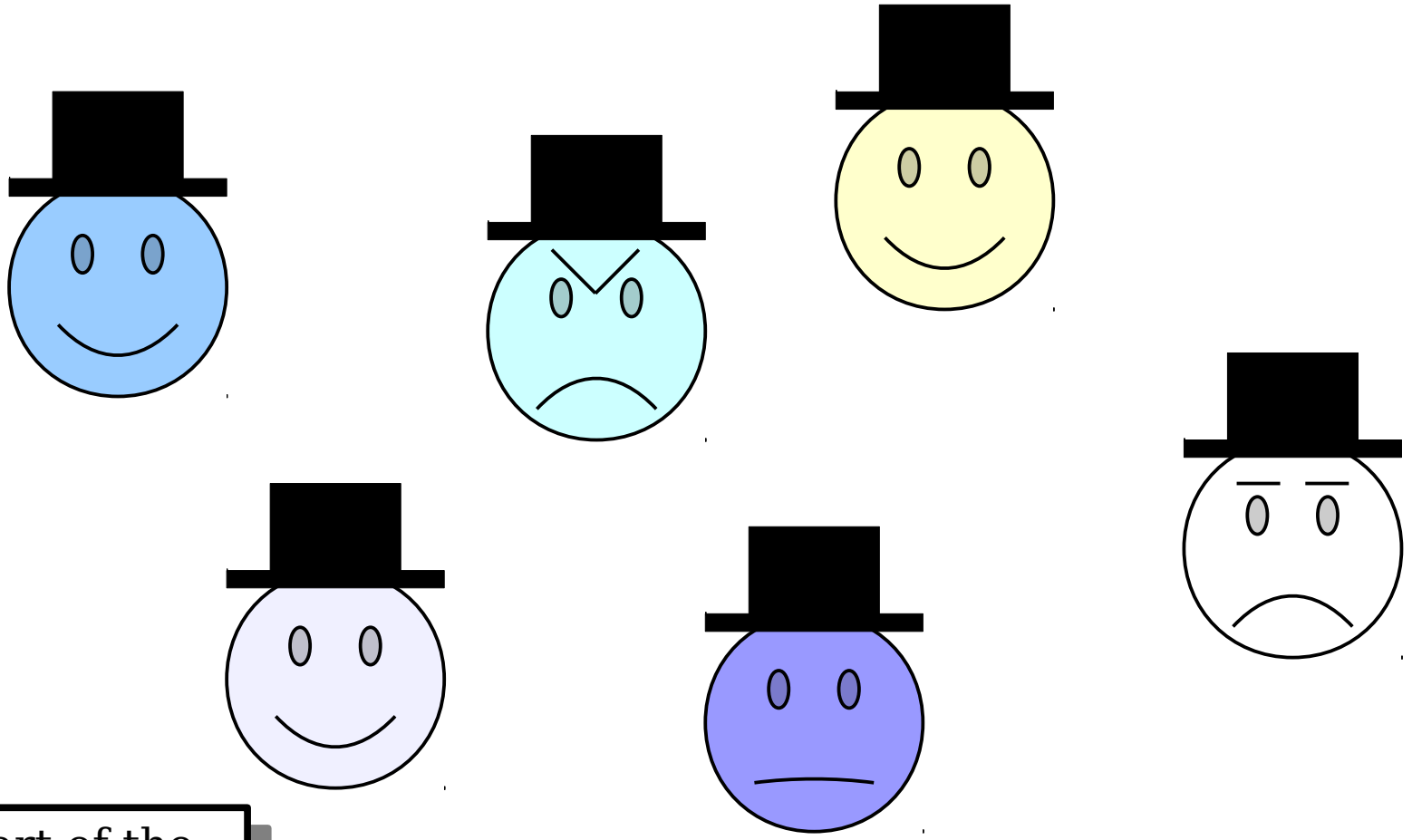
The Universal Quantifier



Is this part of the statement true or false?

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

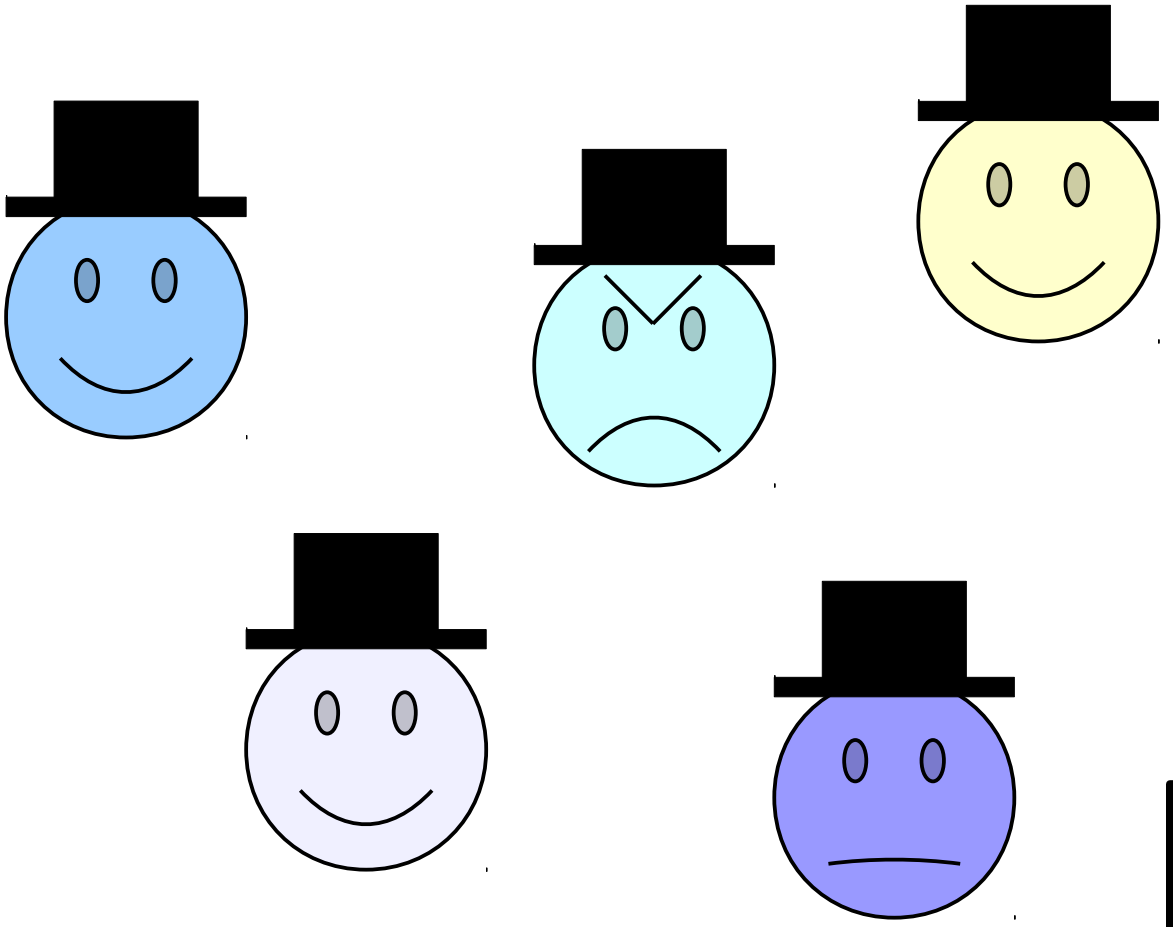
The Universal Quantifier



Is this part of the statement true or false?

$$\text{~~}(\forall x.\ \textit{Smiling}(x))\text{~~} \rightarrow (\forall y.\ \textit{WearingHat}(y))$$

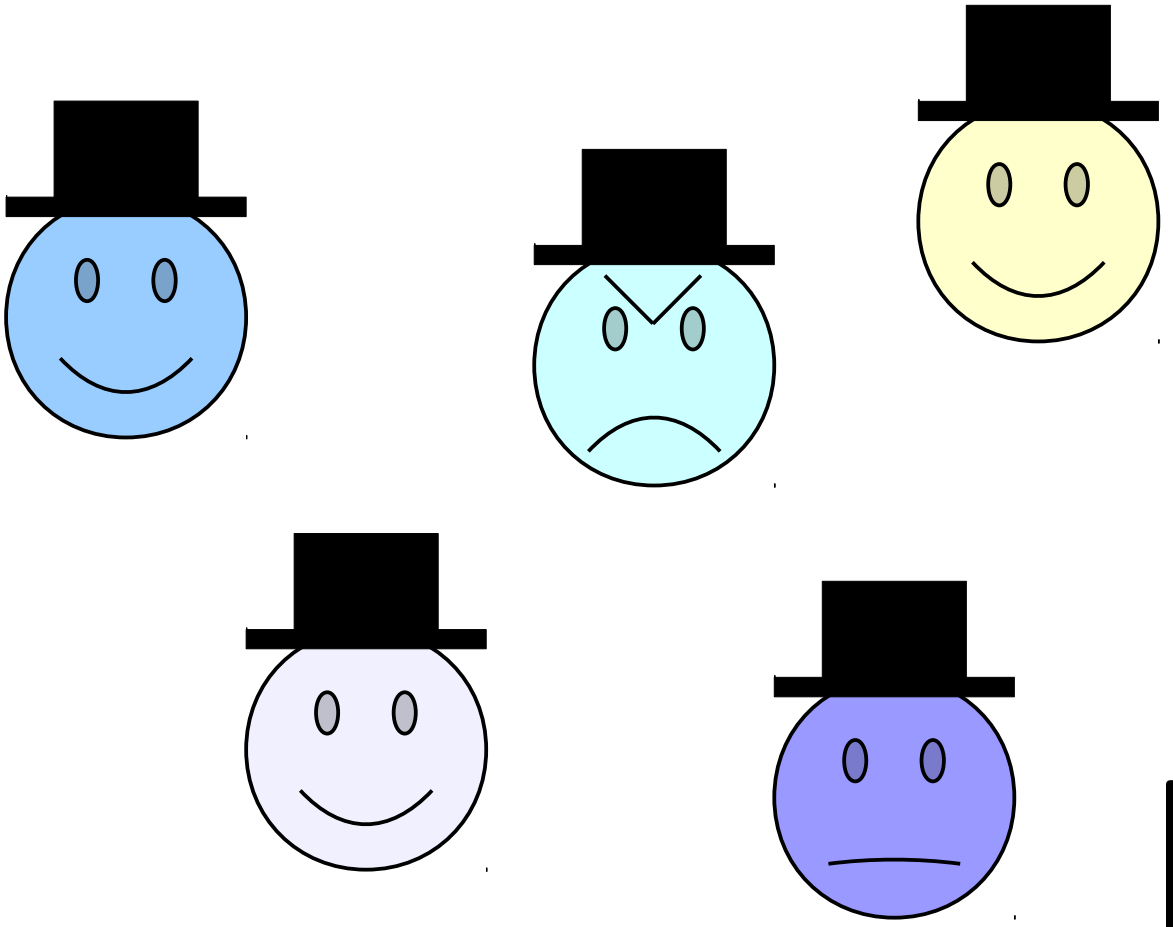
The Universal Quantifier



Is this overall statement true or false in this scenario?

$$\text{~~}(\forall x.\ \textit{Smiling}(x))\text{~~} \rightarrow (\forall y.\ \textit{WearingHat}(y))$$

The Universal Quantifier



Is this overall statement true or false in this scenario?

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

Fun with Edge Cases

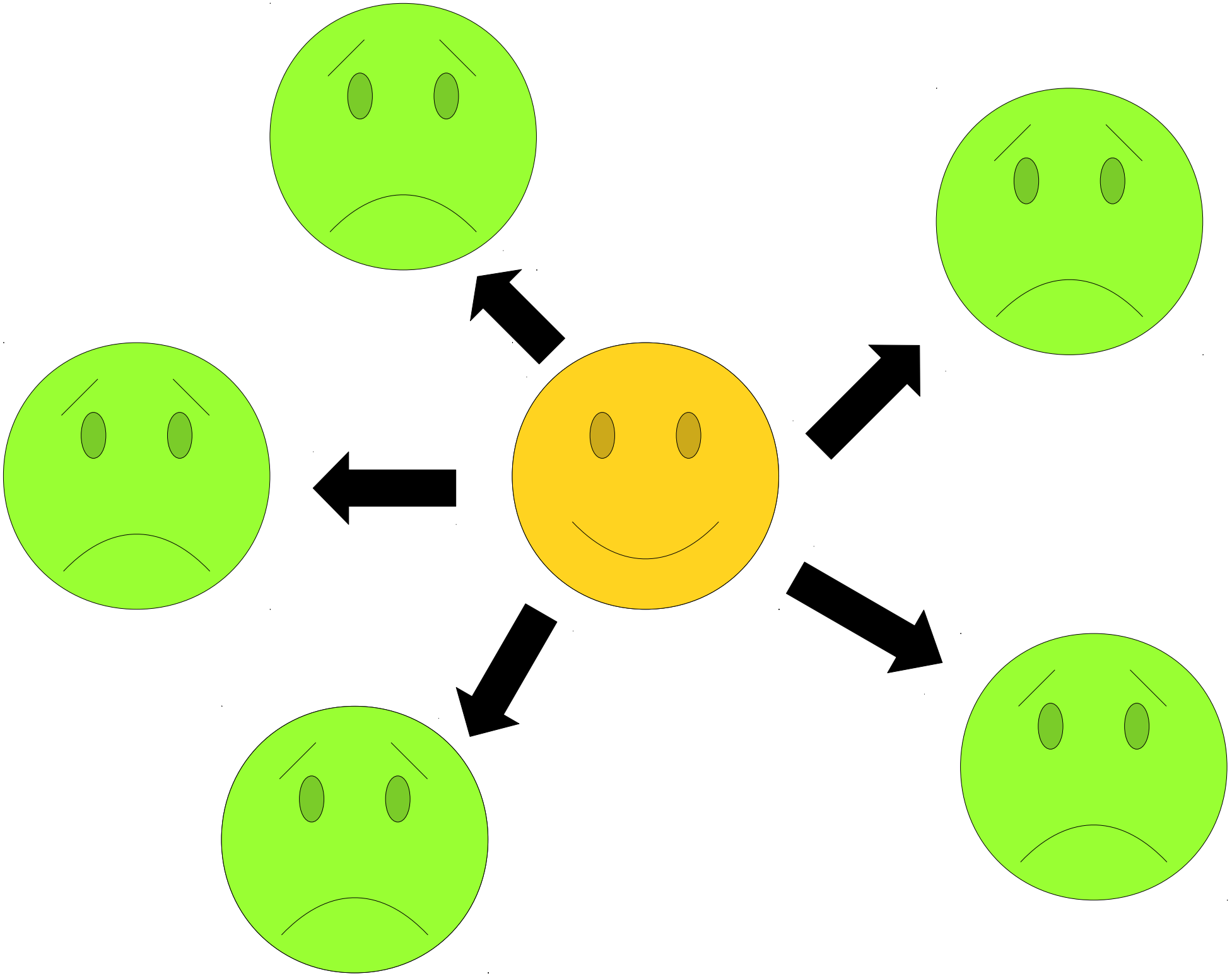
$$\forall x.\ \mathit{Smiling}(x)$$

Fun with Edge Cases

Universally-quantified statements are said to be ***vacuously true*** in empty worlds.

$\forall x.\ Smiling(x)$

Time-Out for Announcements!



Flu Shots!

- The *selfish* reason to get a flu shot: the flu is horrible. Don’t get it.
- The *altruistic* reason to get a flu shot: the flu is horrible. Don’t give it to your friends or family.
- Stanford offers free flu shots. Stop by Vaden between 3:00PM – 6:00PM on Monday, October 7th to get one.
- There are lots of other times; for more information, check **this link**.

Code the Change

- Code the Change is holding its first meeting this evening from 9PM – 10PM in Huang 305.
- Want to learn more? Check out **http://codethechange.stanford.edu/**.

SOLE Frosh Intern Program

- Stanford’s Society of Latinx Engineers (SOLE) is running a frosh intern program. Here’s their description:
- “The Frosh Intern Program is a great opportunity for you to be more involved with SOLE (Society of Latinx Engineers) where you will get to shadow an officer position of your preference in the fall and another in the winter. You will attend officer meetings, help out with society events, and carry out any project ideas you want to bring to SOLE, all while having the officer core to support you from beginning to end. You will get to work with the other interns on fun projects such as planning out one of the weekly meetings with food/activity of your choice. The internship provides great leadership experience, and many interns actually go on to become SOLE officers. We're looking for people who are passionate and willing to commit time and effort into helping SOLE grow into a bigger and better familia.”
- Apply online **here**. Deadline is Thursday, October 10th at 11:00PM.

she++

- Interesting in mentoring high schoolers in CS? Want to do CS outreach? Want to meet a great community of people? Apply to she++!
- Info session on Friday, 6:00PM – 7:30PM on the Storey Lawn. Click **here** to RSVP.
- Apply to she++ using **this link**.

WiCS + SBSE

Mixer

+

THURSDAY, OCTOBER 3
6:30 - 8:00 PM
GATES 219

+

➜ Want to meet more engineering students in the community?

➜ (Just as importantly) want to make new friends over yummy food?

Fill out this link https://forms.gle/MGxrkrpFeE53RvCYA if you would be interested in doing a spotlight

DIGITAL GOVERNMENT IN CALIFORNIA

civic tech // gov tech // public interest tech

APPLY AT ASB.STANFORD.EDU
BY FRI OCT. 11 @ 11:59 P.M.

THANKSGIVING BACK 2019
NOV. 24–27, 2019

- Meet with state and local government leaders, prominent community organizations and non-profits, startups and technology policy leaders
- Learn about how governments use technology to provide services and how citizens engage with their governments through technology
- Explore how technology builds and erodes trust for communities and movements to safeguard against misuse

ThanksGiving Back
STANFORD UNIVERSITY

ASSU
ASSOCIATED STUDENTS OF STANFORD UNIVERSITY

CS103 Announcements

Checkpoints Graded

- The Problem Set One checkpoint will be graded and released by 5PM today on GradeScope.
- ***You need to look over our feedback as soon as possible.***
  - The purpose of the checkpoint is to help you see where to focus and how to improve.
  - If you don’t review the feedback you received, you risk making the same mistakes in the future.

Your Questions

“How will the knowledge I learn in CS103 be applied/used? Any concrete examples?”

In a more abstract sense, learning to think about problems through a mathematical lens will help you design better systems. I can give a few examples of that if you'd like!

Other topics from this course – graph theory, equivalence relations, finite automata, regular expressions, etc. – show up *everywhere!* It's amazing how much mileage you'll get out of those concepts.

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.
- Need to take a negation? Translate your statement into FOL, negate it, then translate it back.
- Want to prove something by contrapositive? Translate your implication into FOL, take the contrapositive, then translate it back.

Translating Into Logic

- When translating from English into first-order logic, we recommend that you

  think of first-order logic as a mathematical programming language.

- Your goal is to learn how to combine basic concepts (quantifiers, connectives, etc.) together in ways that say what you mean.

Using the predicates

- *Happy*(*x*), which states that *x* is happy, and
- *WearingHat*(*x*), which states that *x* is wearing a hat,

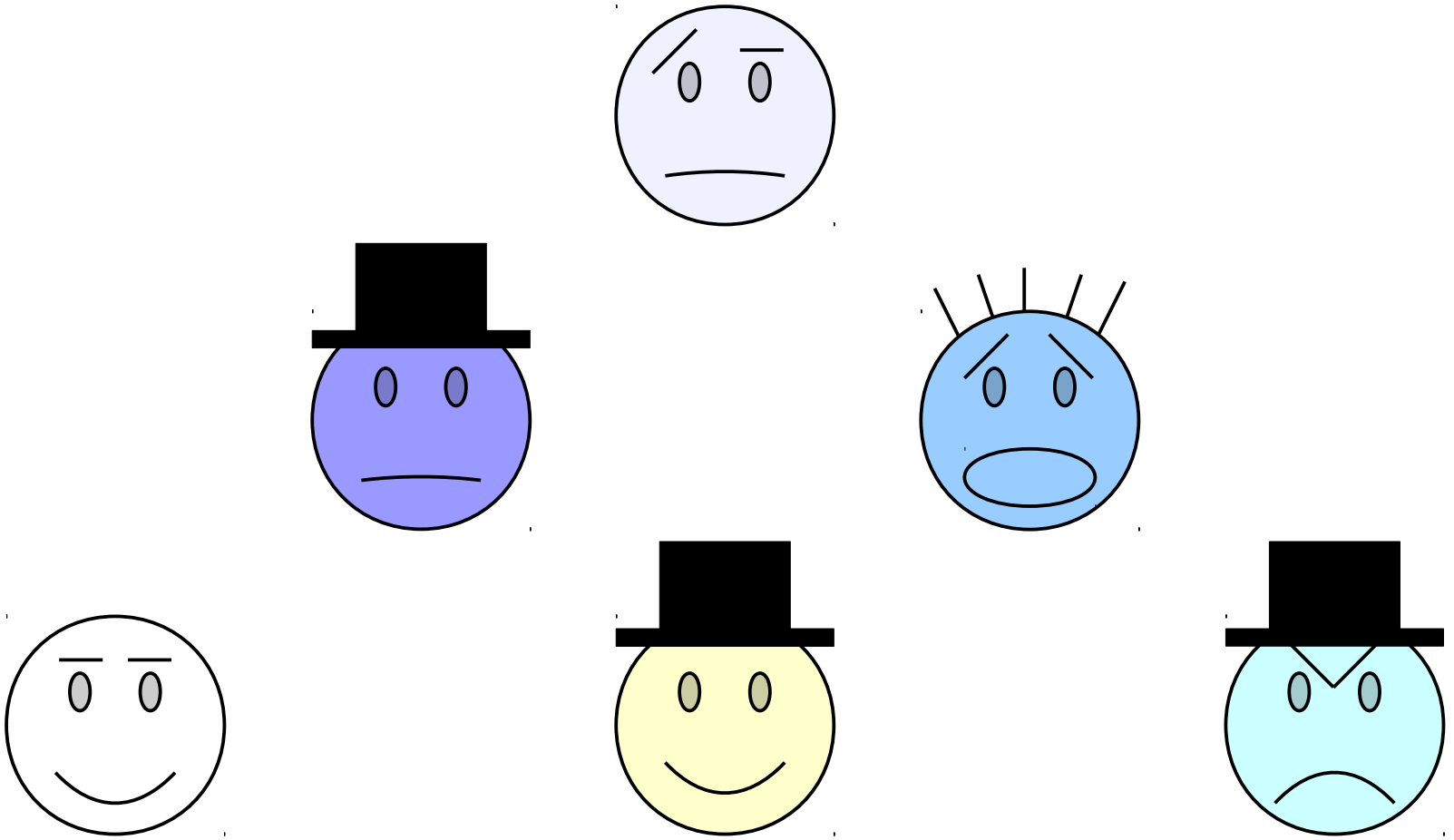
write a sentence in first-order logic that says

some happy person wears a hat.

“Some happy person wears a hat.”

$\exists x.\ (Happy(x) \wedge WearingHat(x))$

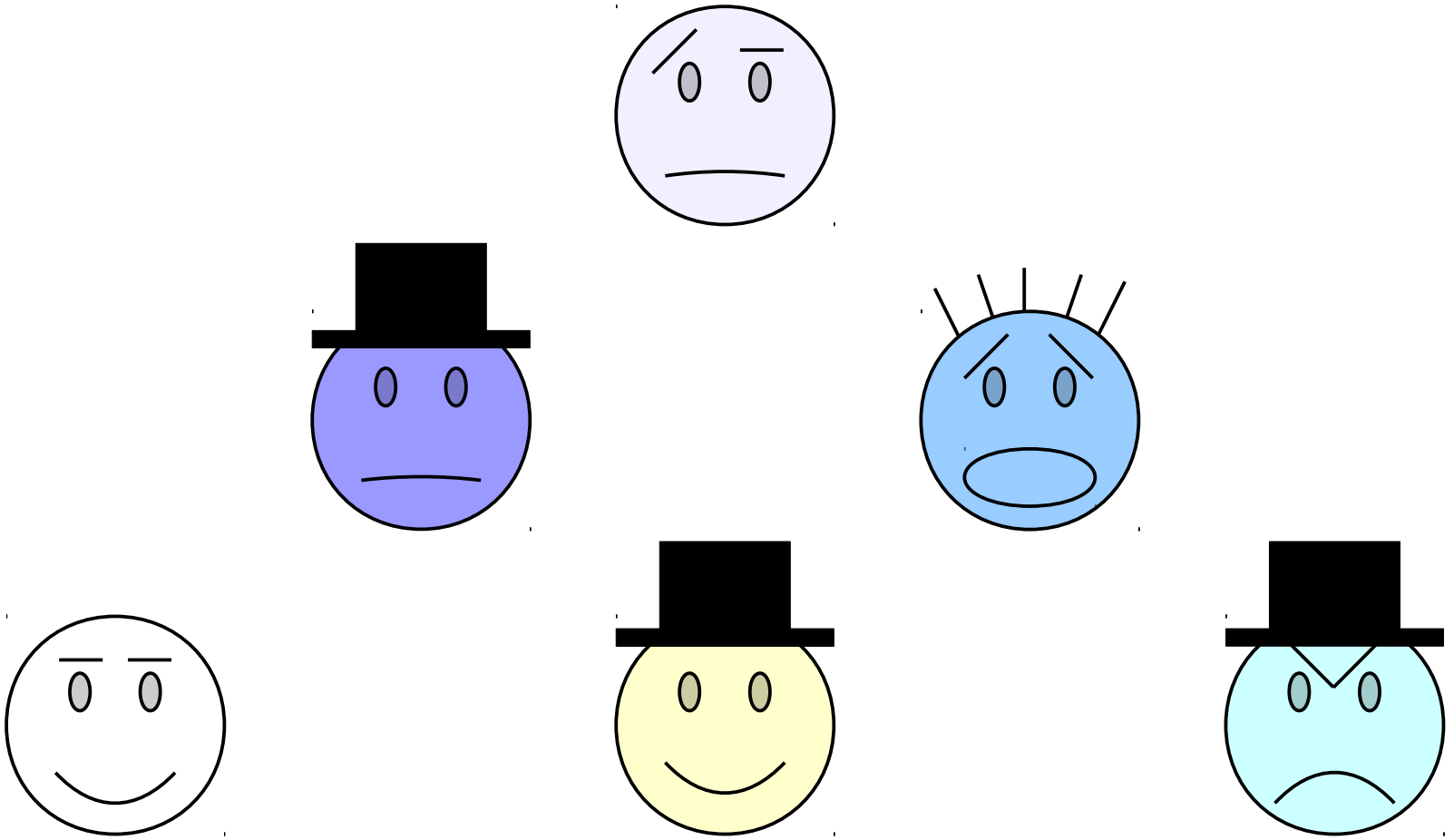
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$



"Some happy person wears a hat."

$\exists x.\ (Happy(x) \wedge WearingHat(x))$

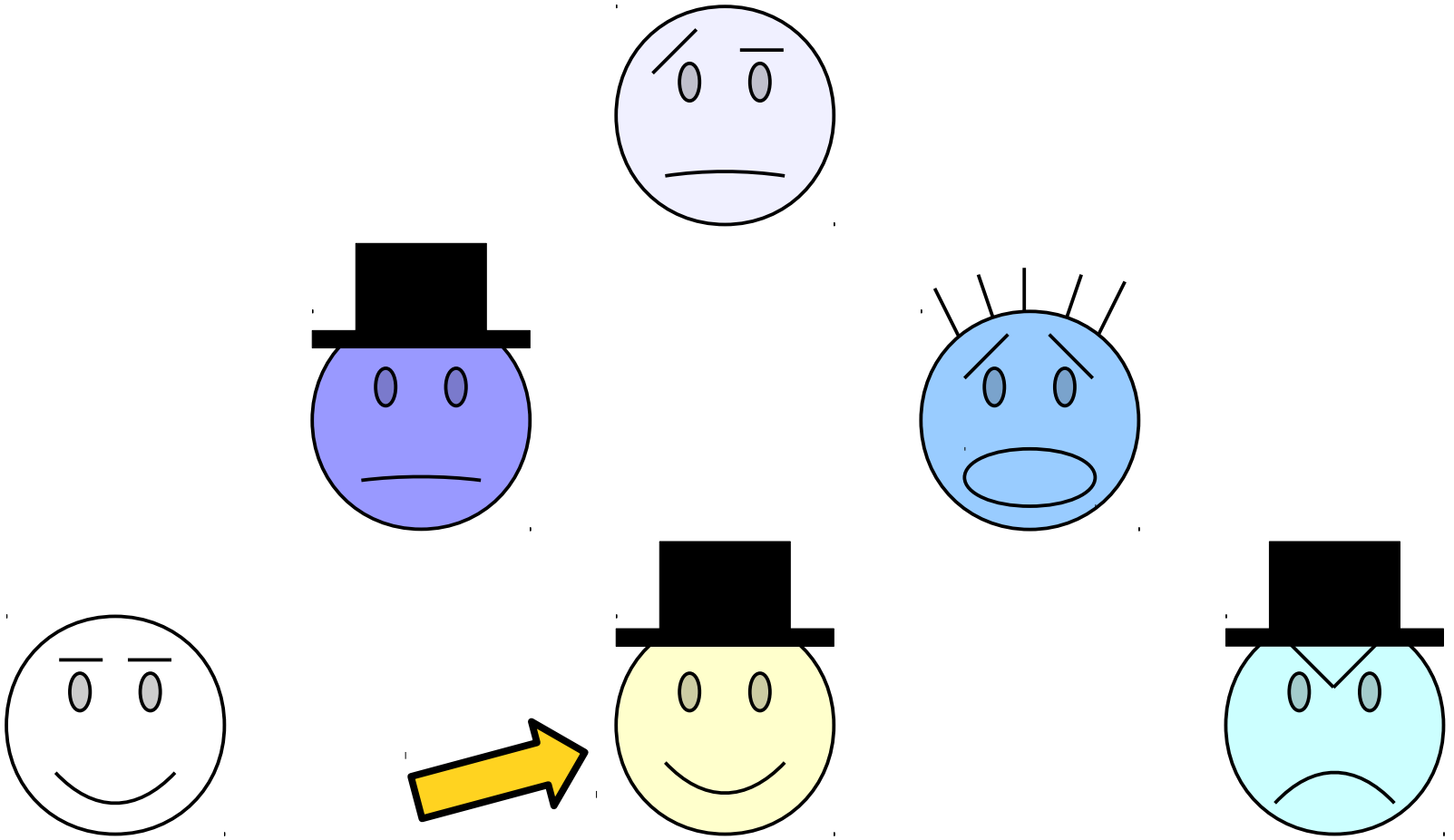
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$



"Some happy person wears a hat."

$\exists x.\ (Happy(x) \wedge WearingHat(x))$

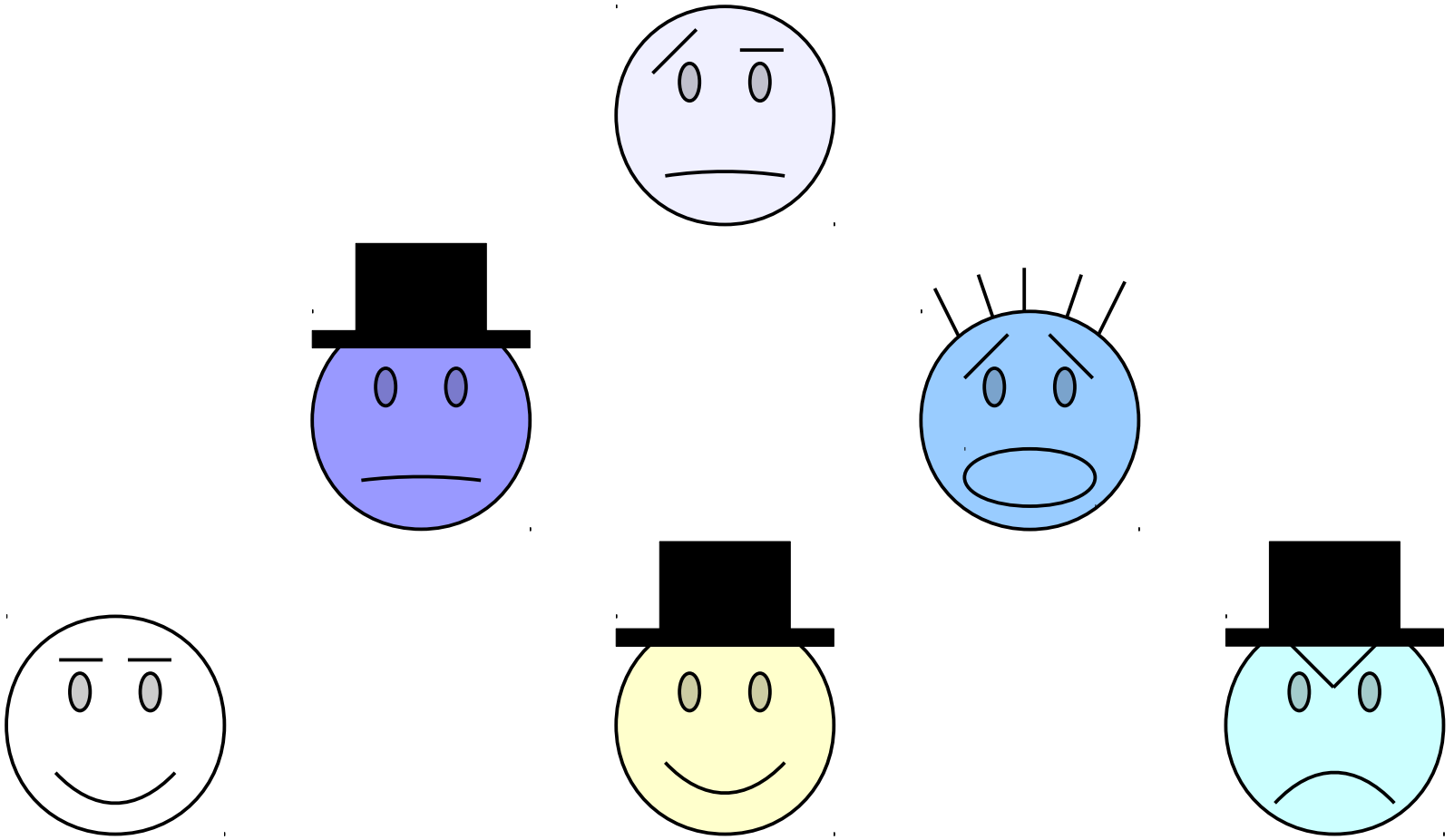
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$



"Some happy person wears a hat." **True**

$\exists x.\ (Happy(x) \wedge WearingHat(x))$

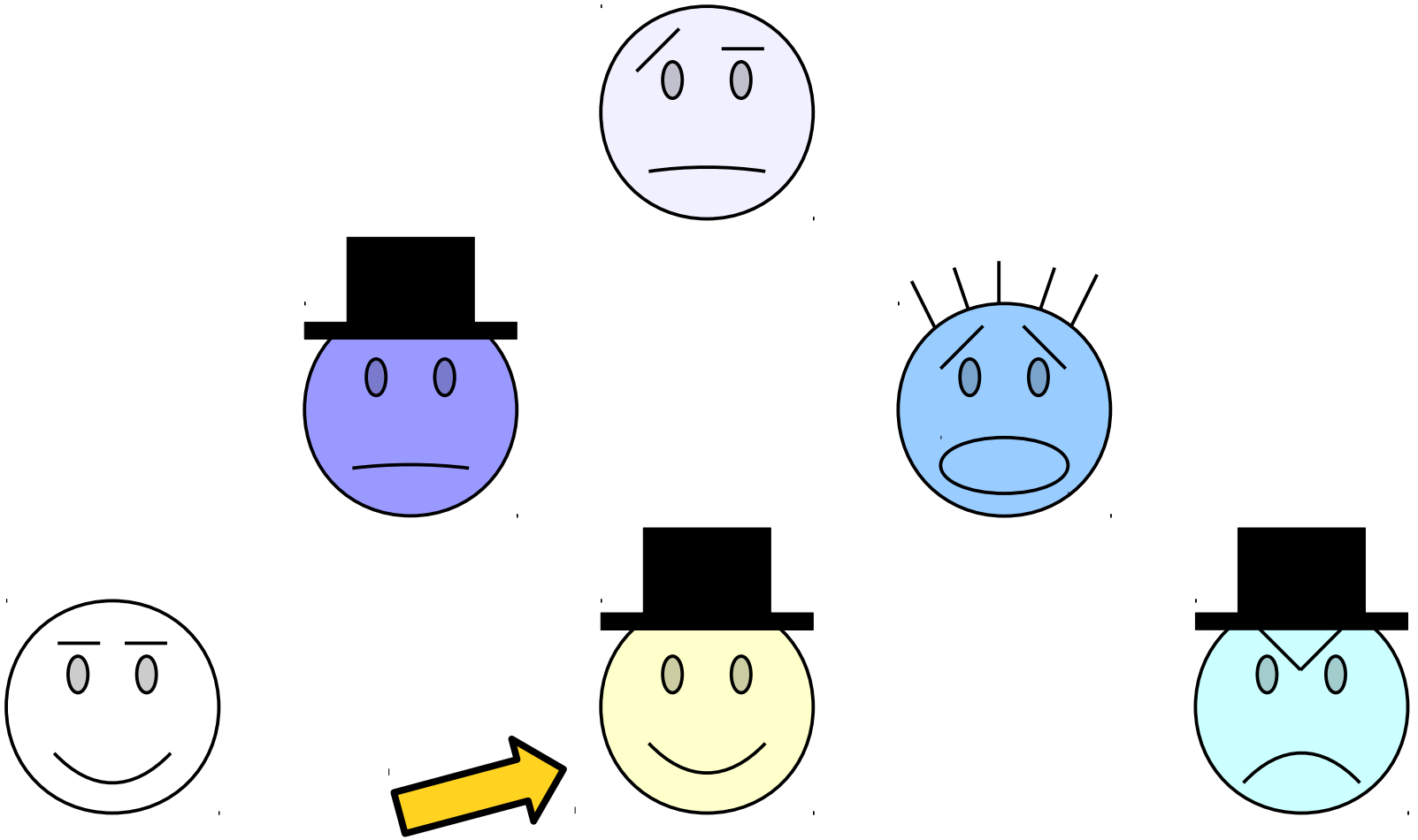
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$



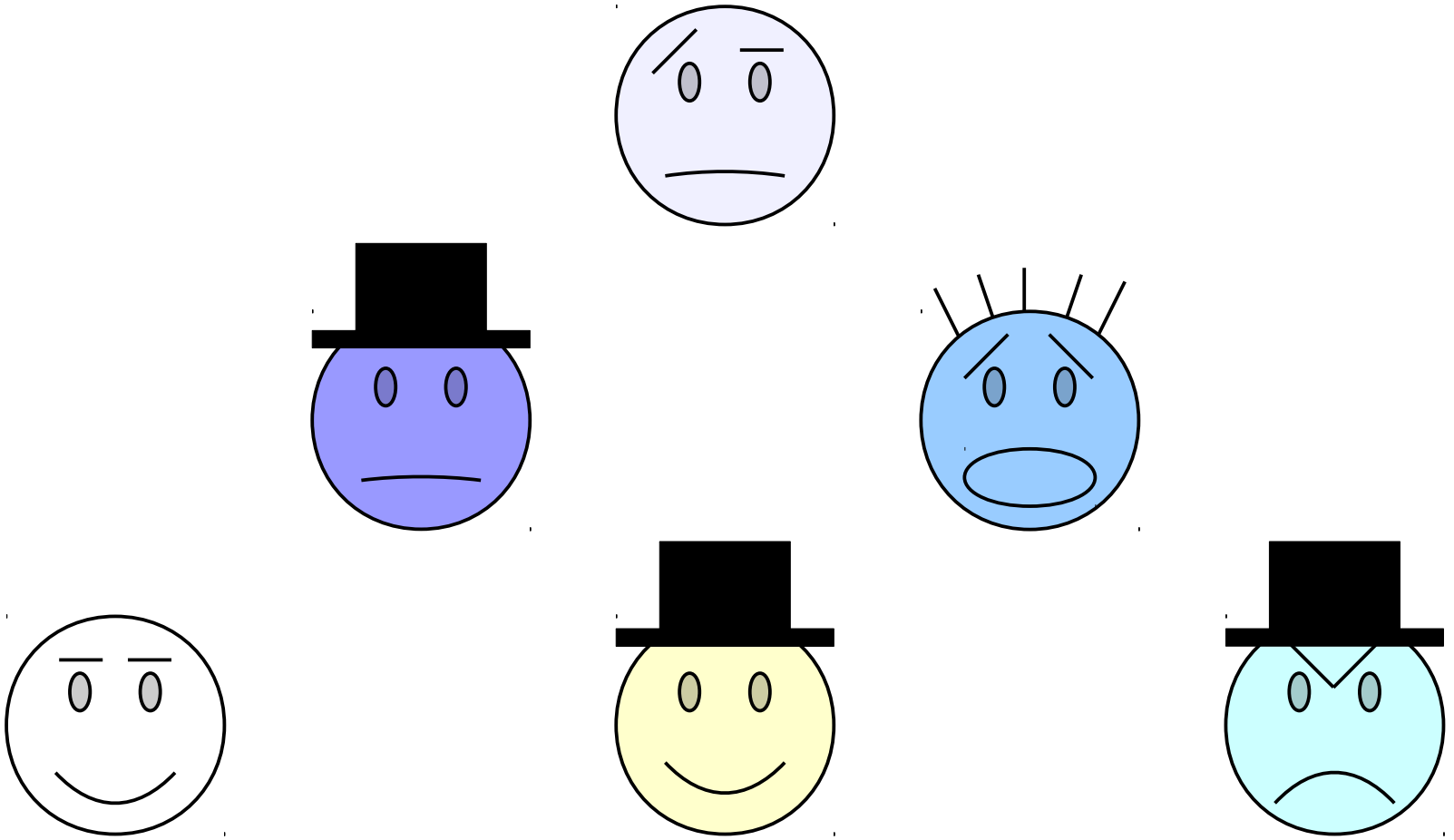
"Some happy person wears a hat." ***True***

$\exists x.\ (Happy(x) \wedge WearingHat(x))$

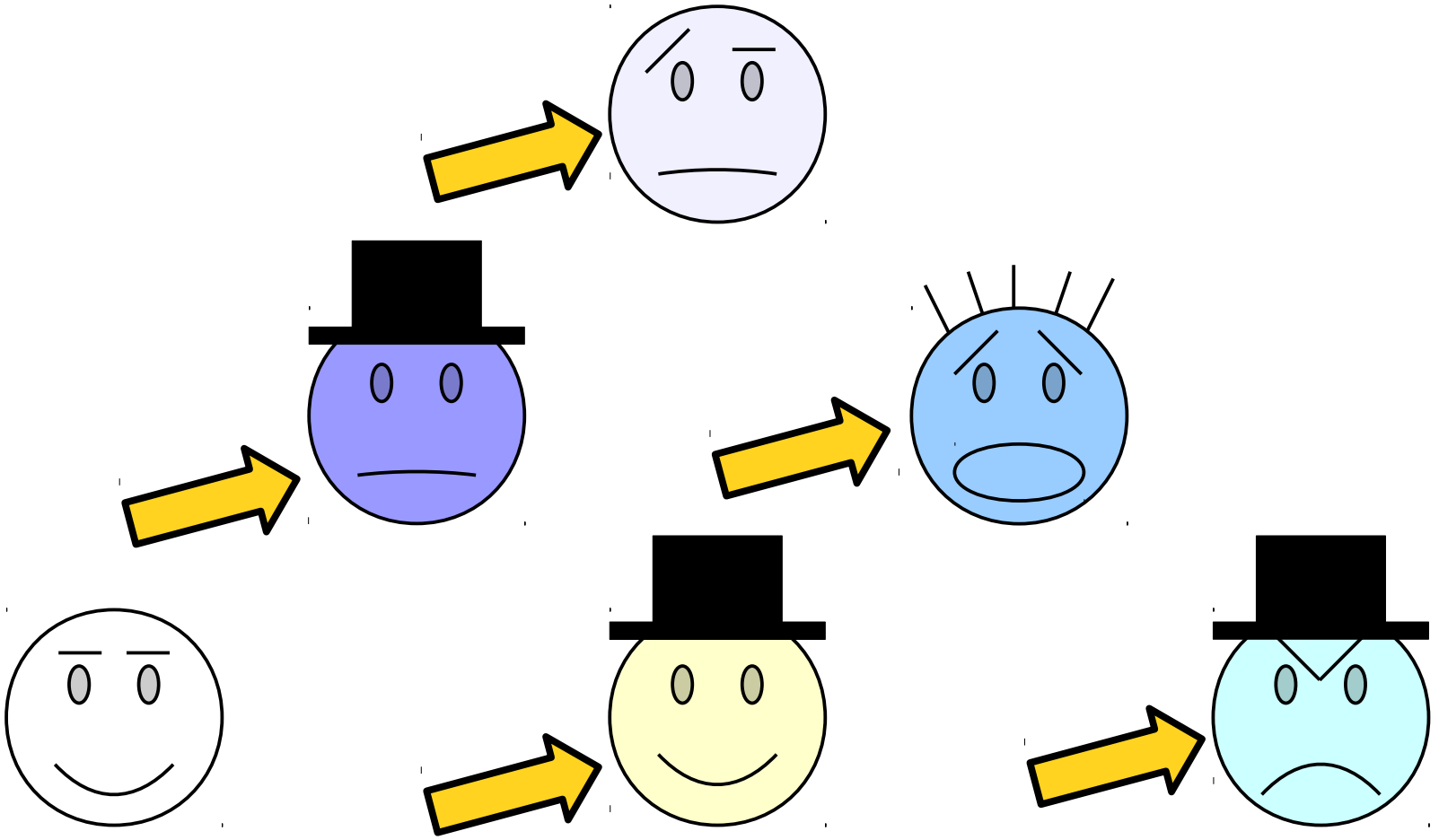
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$



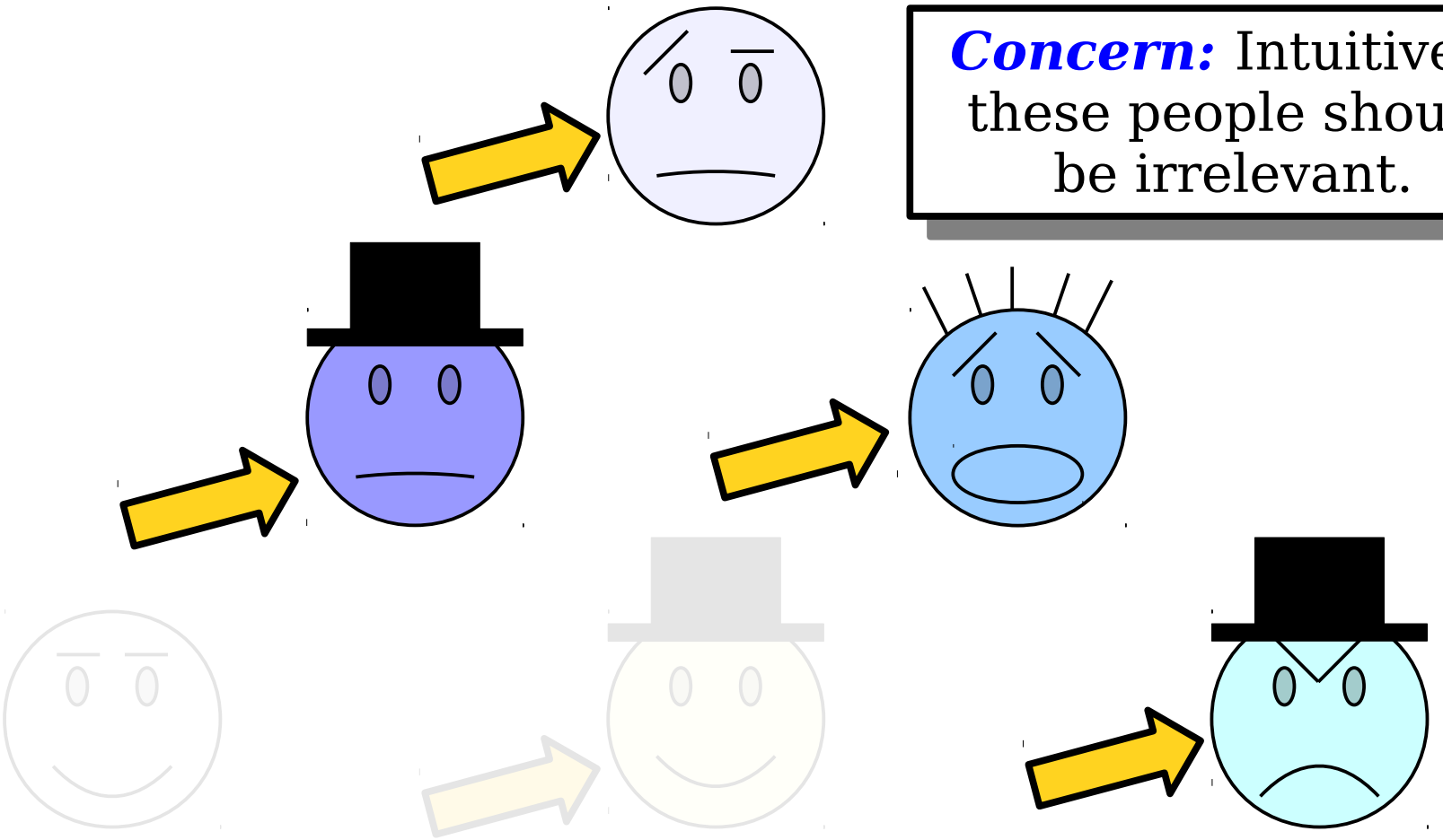
“Some happy person wears a hat.”	***True***
$\exists x.\ (Happy(x) \wedge WearingHat(x))$	***True***
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$	



“Some happy person wears a hat.”	***True***
$\exists x.\ (Happy(x) \wedge WearingHat(x))$	***True***
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$	



“Some happy person wears a hat.”	***True***
$\exists x.\ (Happy(x) \wedge WearingHat(x))$	***True***
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$	***True***



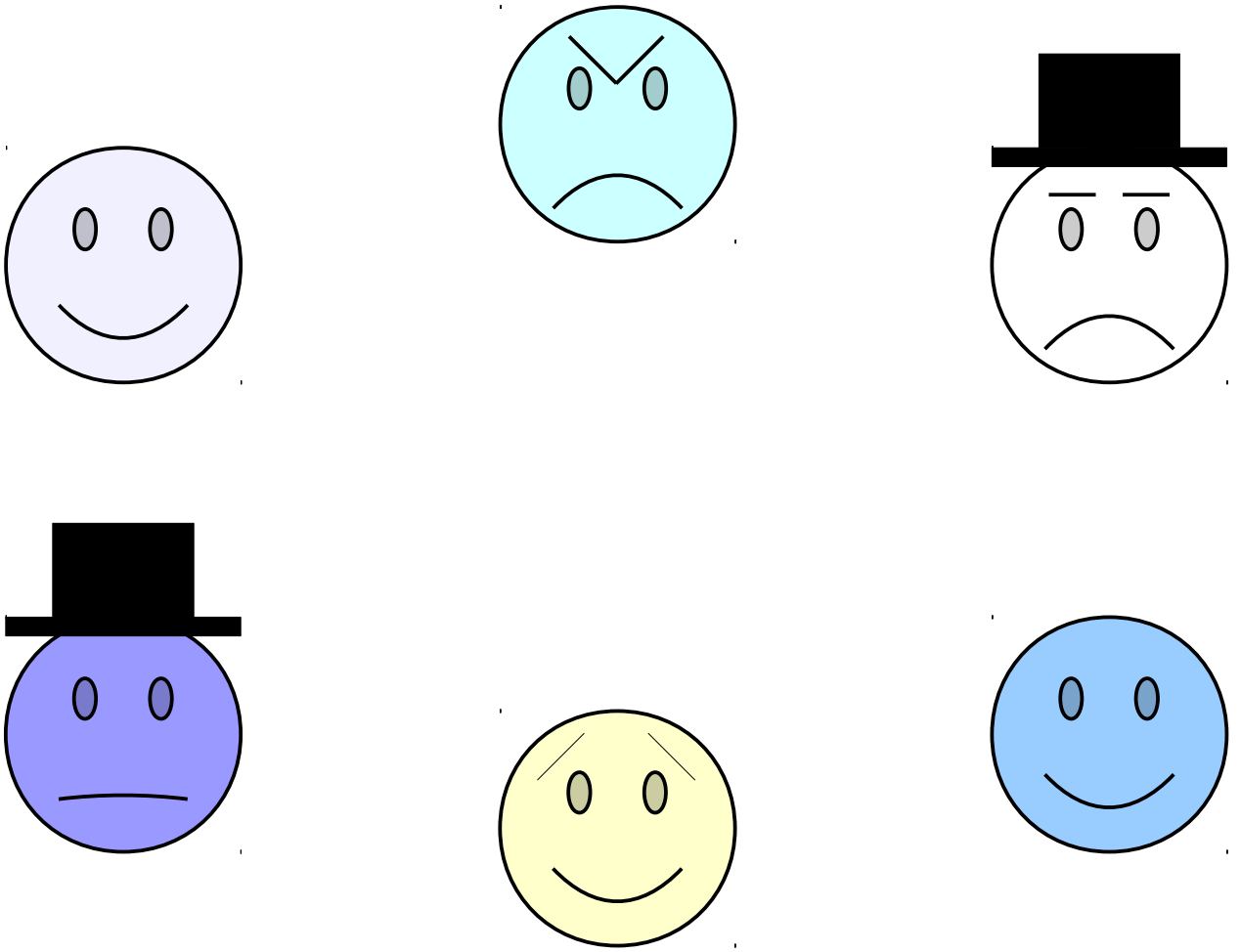
Concern: Intuitively, these people should be irrelevant.

"Some happy person wears a hat."	***True***
$\exists x.\ (Happy(x) \wedge WearingHat(x))$	***True***
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$	***True***

“Some happy person wears a hat.”

$\exists x.\ (Happy(x) \wedge WearingHat(x))$

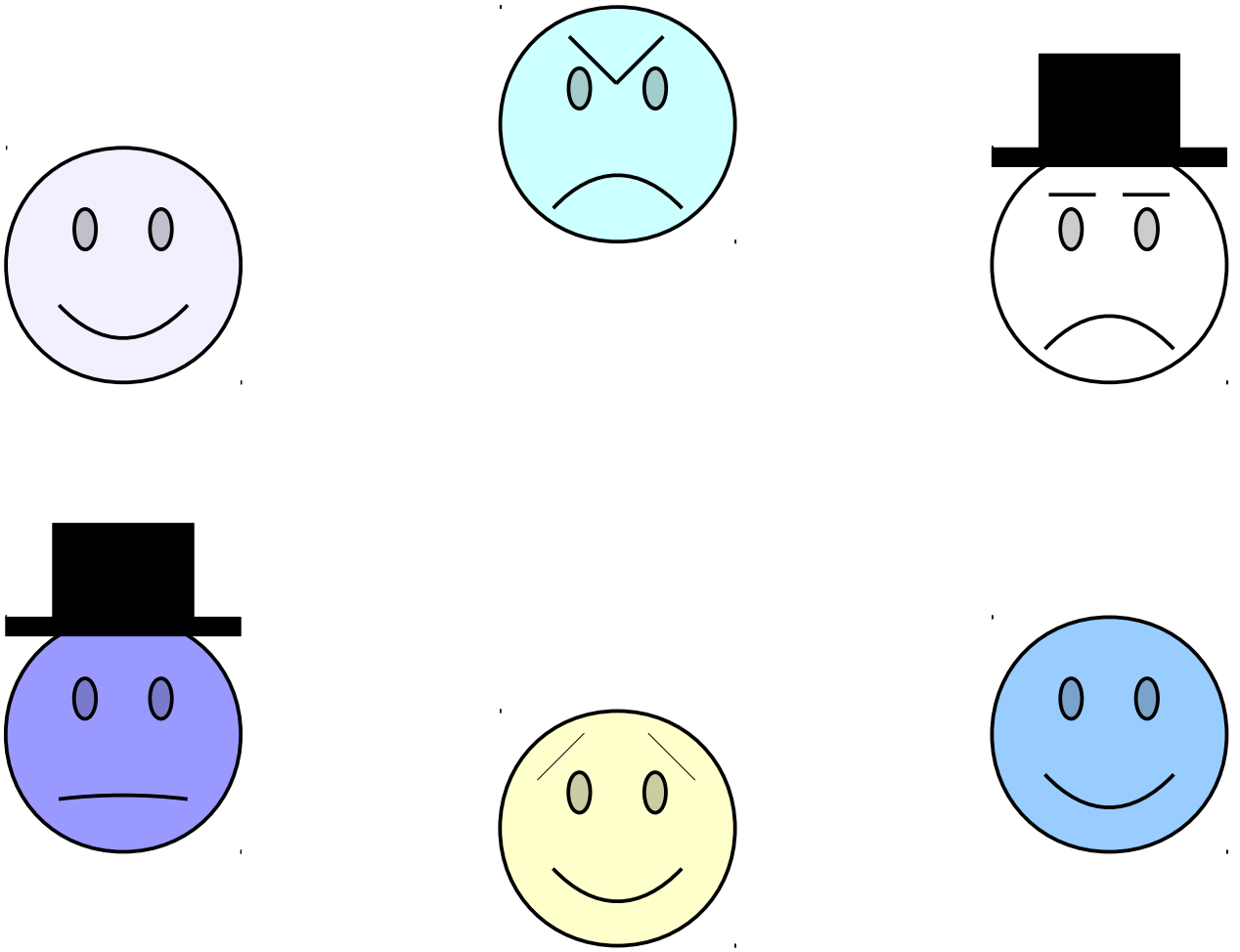
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$



"Some happy person wears a hat."

$\exists x.\ (Happy(x) \wedge WearingHat(x))$

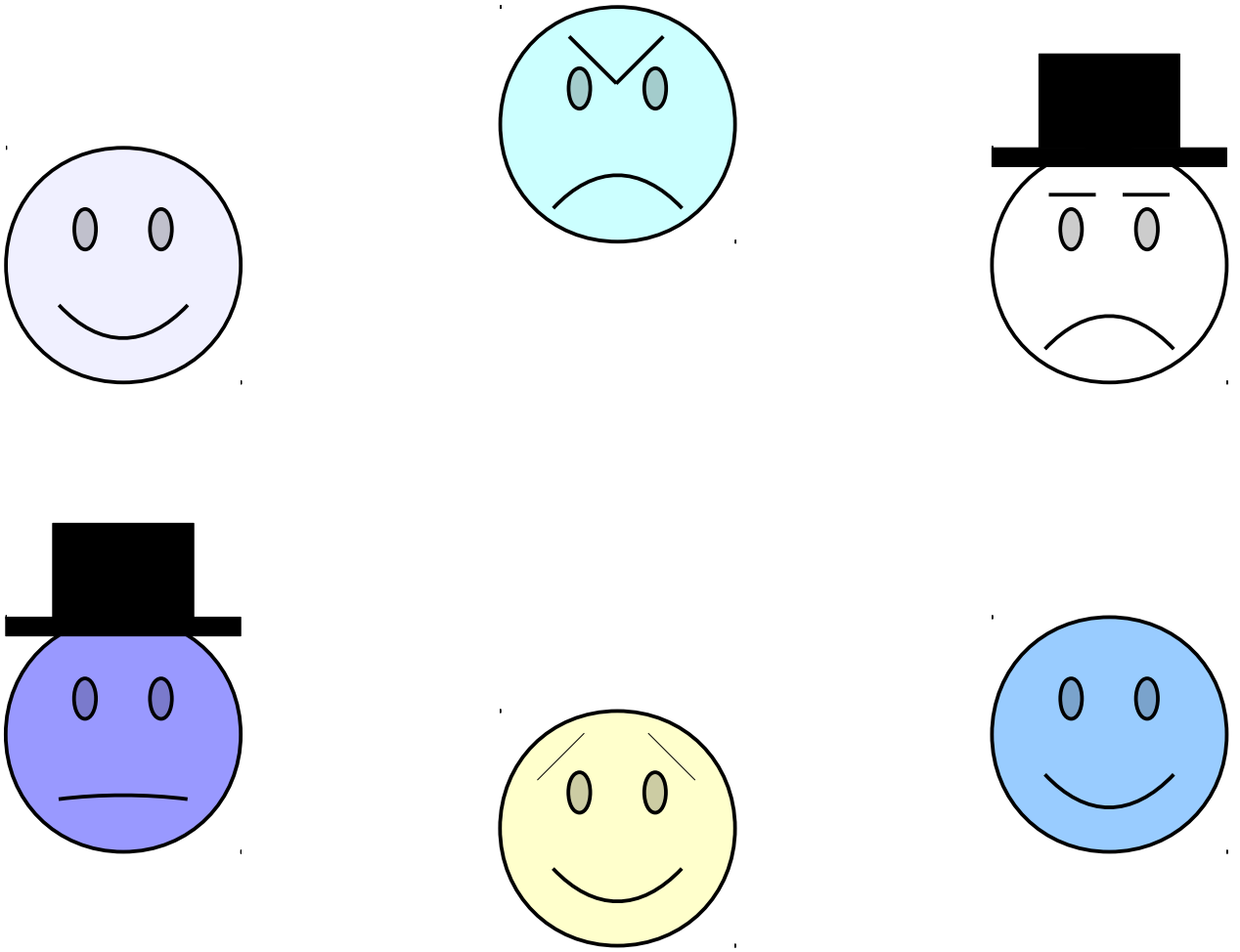
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$



"Some happy person wears a hat."

$\exists x.\ (Happy(x) \wedge WearingHat(x))$

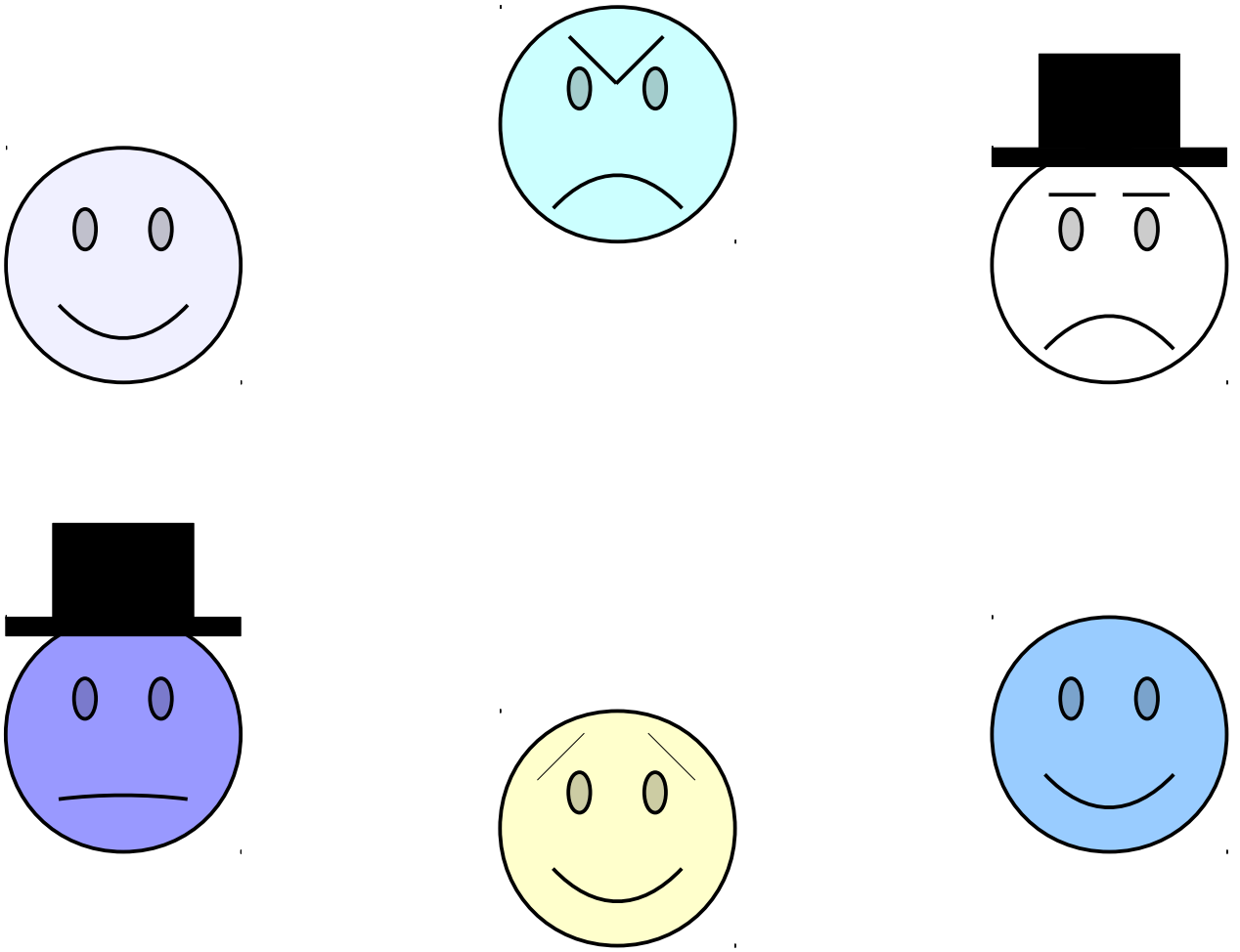
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$



"Some happy person wears a hat." ***False***

$\exists x.\ (Happy(x) \wedge WearingHat(x))$

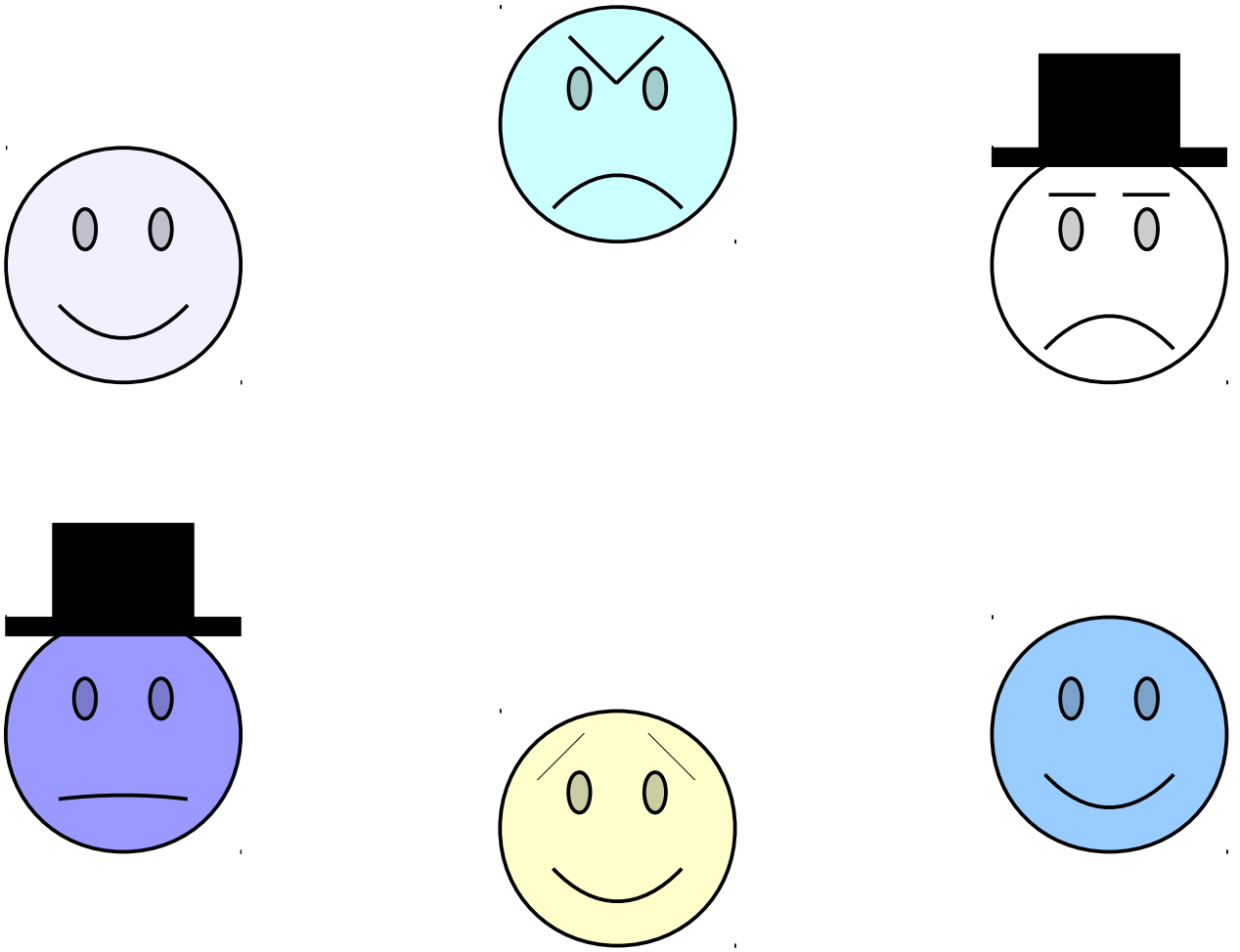
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$



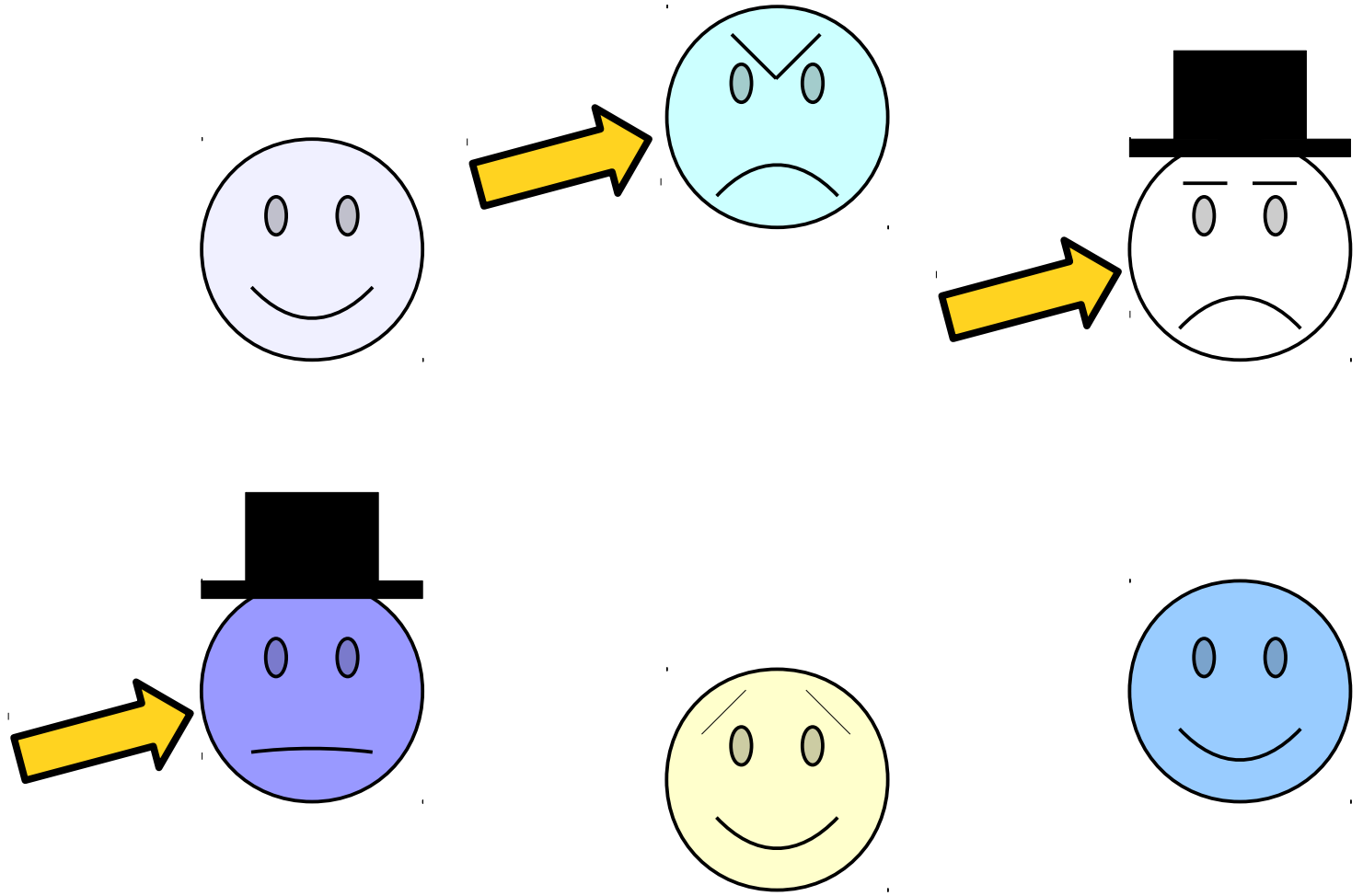
"Some happy person wears a hat." ***False***

$\exists x.\ (Happy(x) \wedge WearingHat(x))$

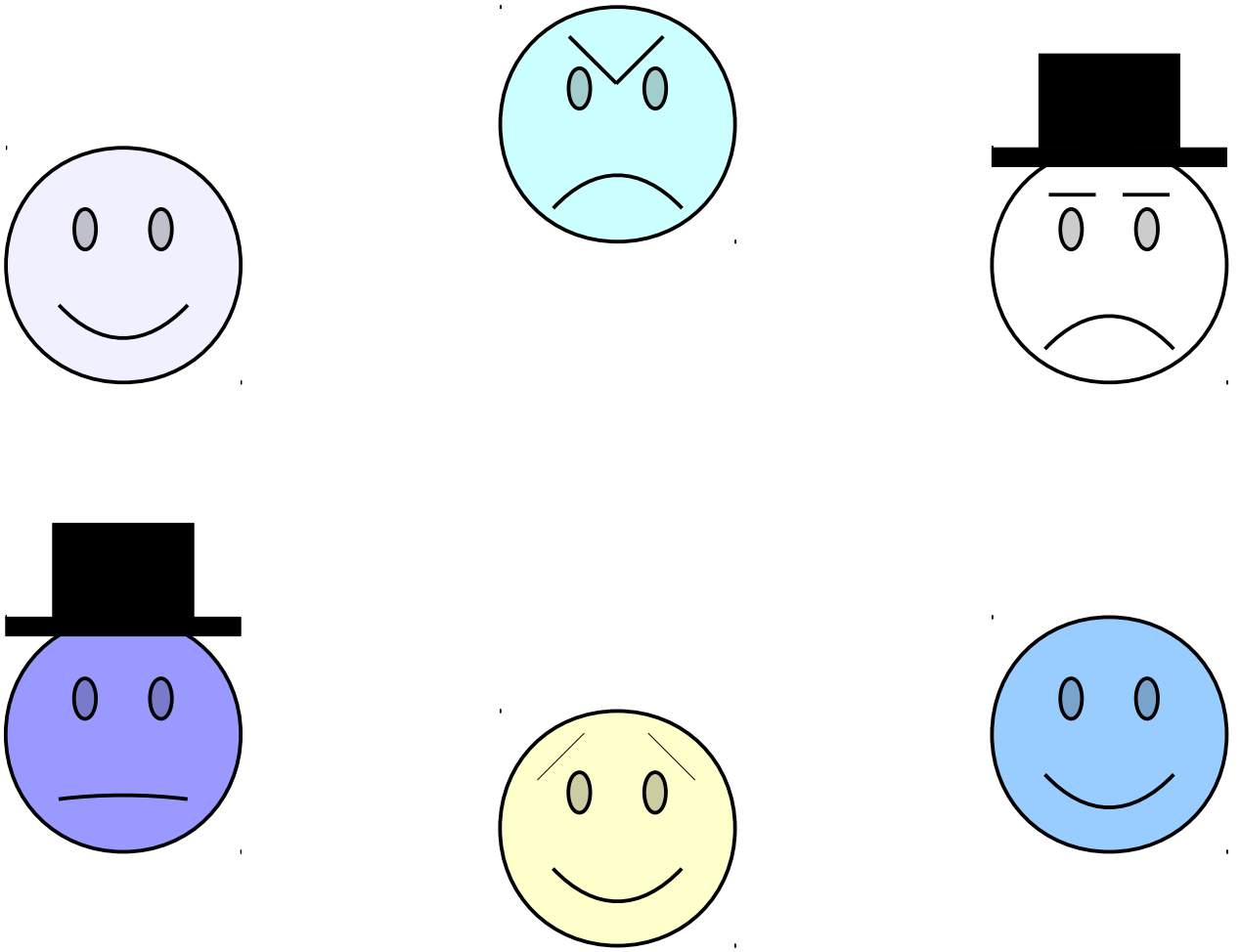
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$



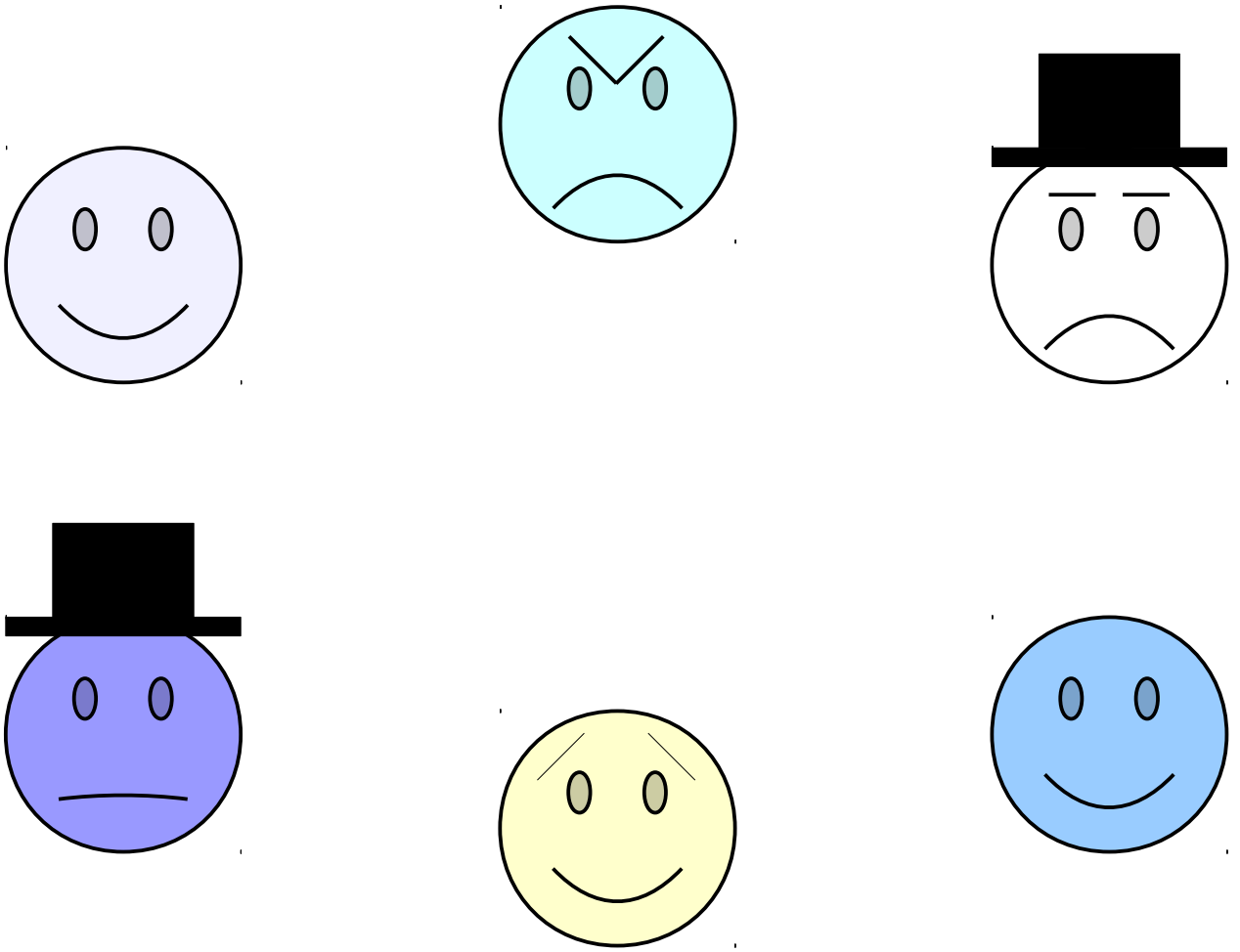
“Some happy person wears a hat.”	***False***
$\exists x.\ (Happy(x) \wedge WearingHat(x))$	***False***
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$	



“Some happy person wears a hat.”	***False***
$\exists x.\ (Happy(x) \wedge WearingHat(x))$	***False***
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$	***True***



“Some happy person wears a hat.”	***False***
$\exists x.\ (Happy(x) \wedge WearingHat(x))$	***False***
$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$	***True***



“Some happy person wears a hat.”	***False***
$\exists x.\ (Happy(x) \wedge WearingHat(x))$	***False***
~~$\exists x.\ (Happy(x) \rightarrow WearingHat(x))$~~	***True***

“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.

Using the predicates

- *Happy*(*x*), which states that *x* is happy, and
- *WearingHat*(*x*), which states that *x* is wearing a hat,

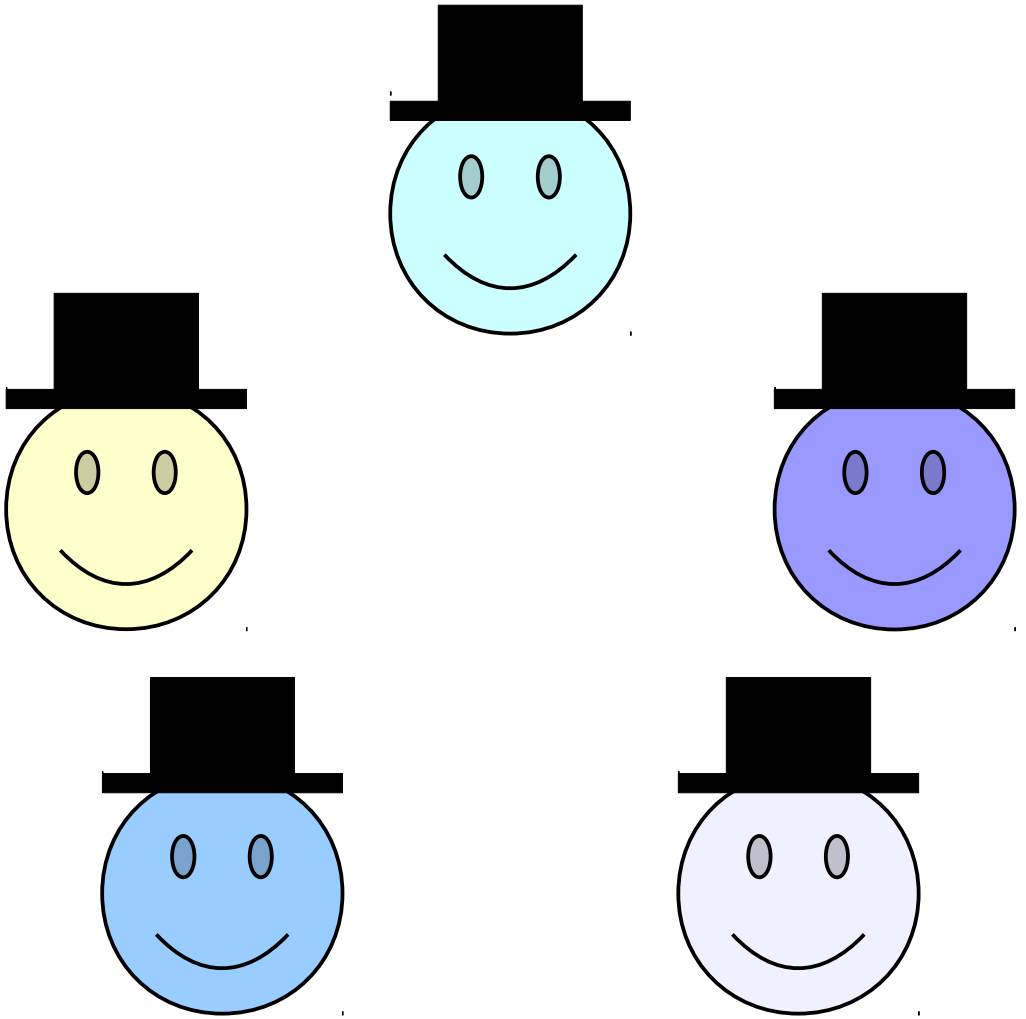
write a sentence in first-order logic that says

every happy person wears a hat.

“Every happy person wears a hat.”

$$\forall x.\ (Happy(x) \wedge WearingHat(x))$$

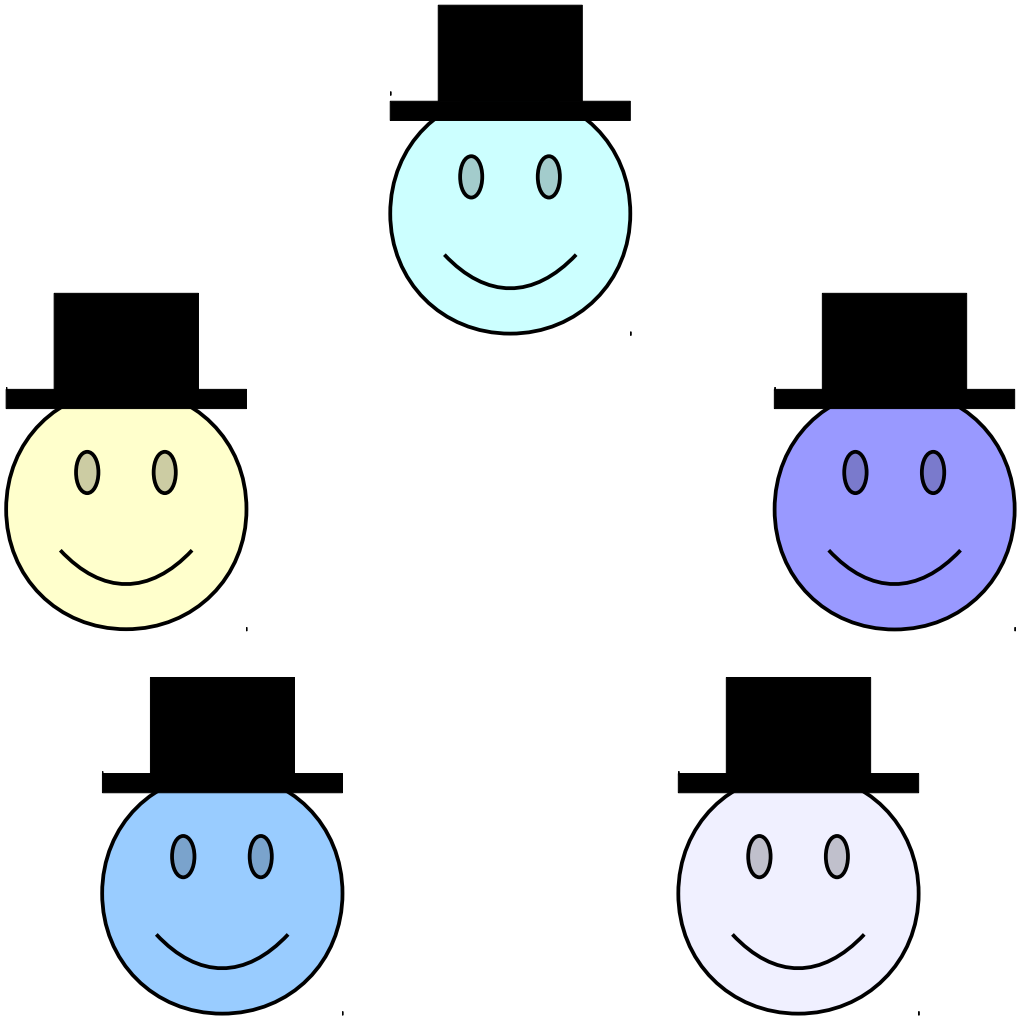
$$\forall x.\ (Happy(x) \rightarrow WearingHat(x))$$



"Every happy person wears a hat."

$\forall x.\ (Happy(x) \wedge WearingHat(x))$

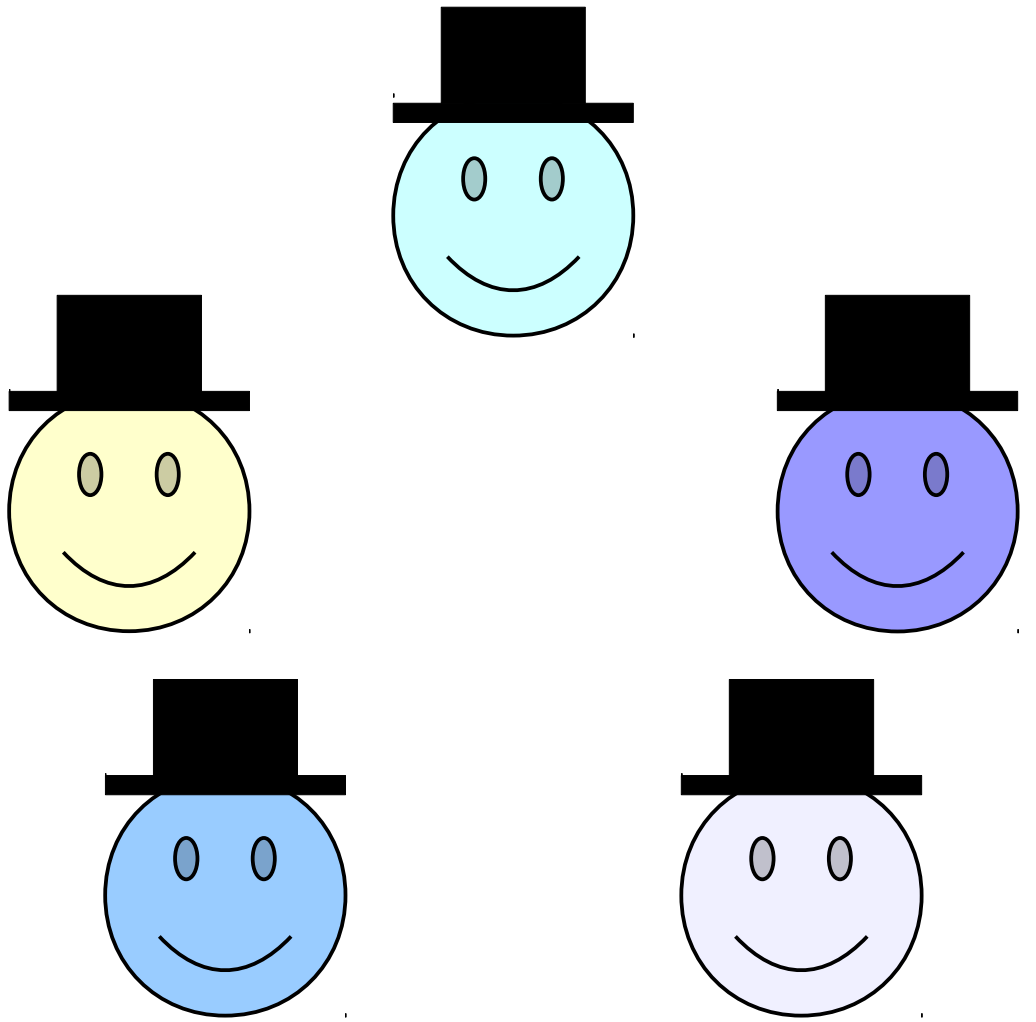
$\forall x.\ (Happy(x) \rightarrow WearingHat(x))$



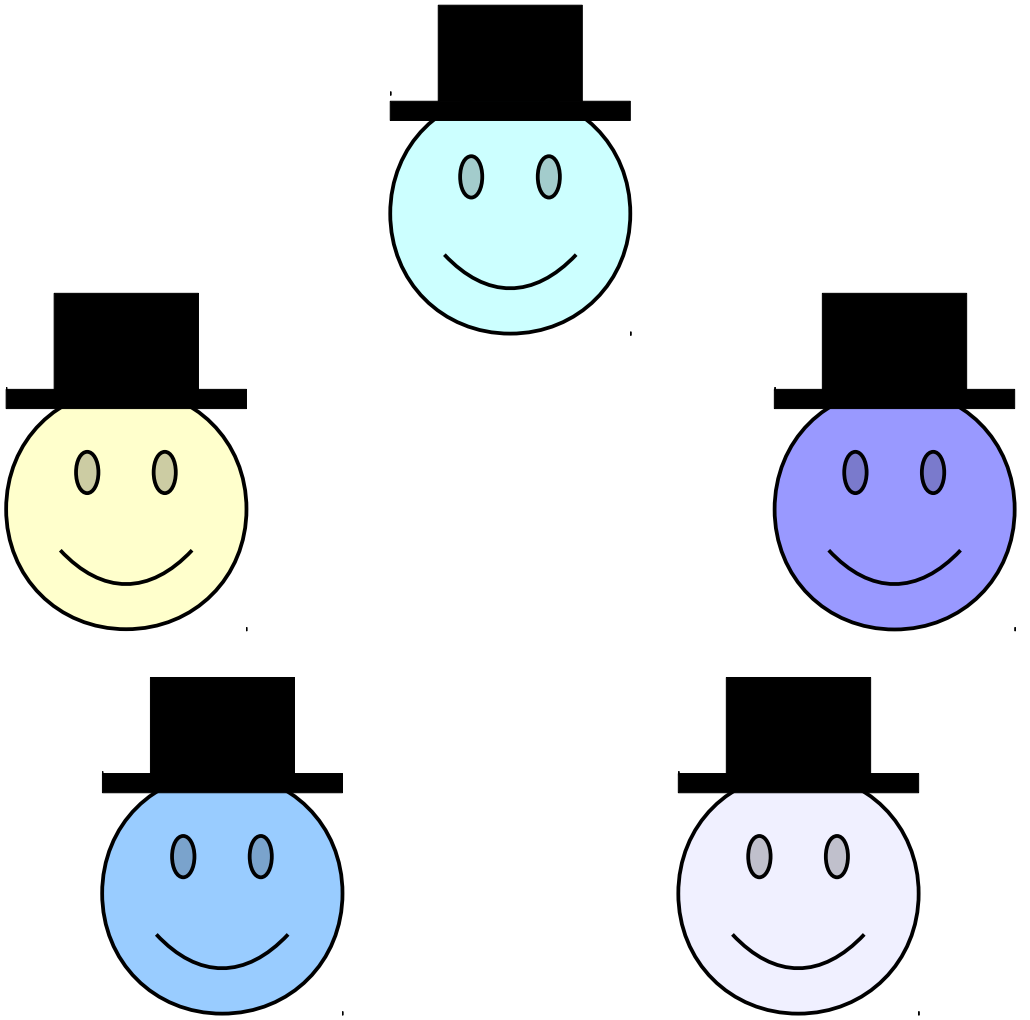
"Every happy person wears a hat." ***True***

$$\forall x.\ (Happy(x) \wedge WearingHat(x))$$

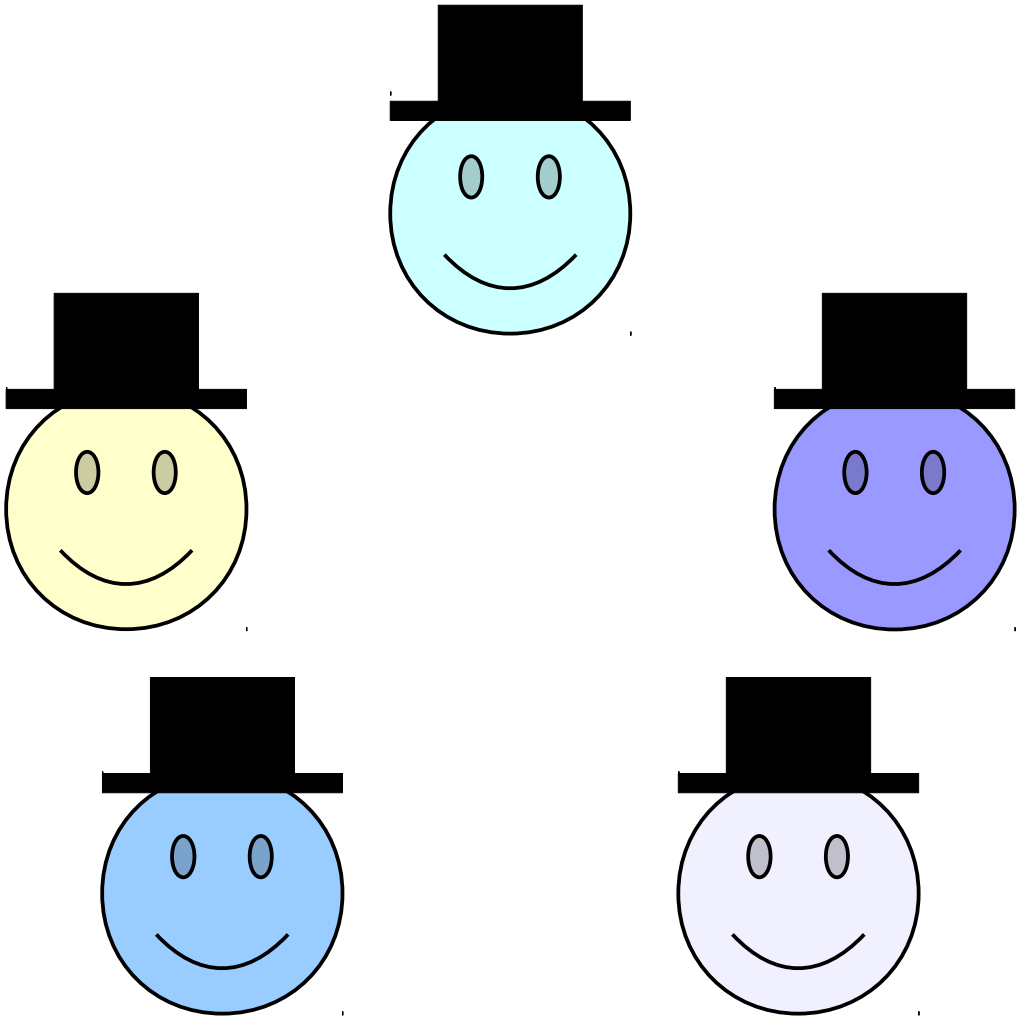
$$\forall x.\ (Happy(x) \rightarrow WearingHat(x))$$



“Every happy person wears a hat.”	***True***
$\forall x.\ (Happy(x) \wedge WearingHat(x))$	***True***
$\forall x.\ (Happy(x) \rightarrow WearingHat(x))$	



“Every happy person wears a hat.”	***True***
$\forall x.\ (Happy(x) \wedge WearingHat(x))$	***True***
$\forall x.\ (Happy(x) \rightarrow WearingHat(x))$	***True***

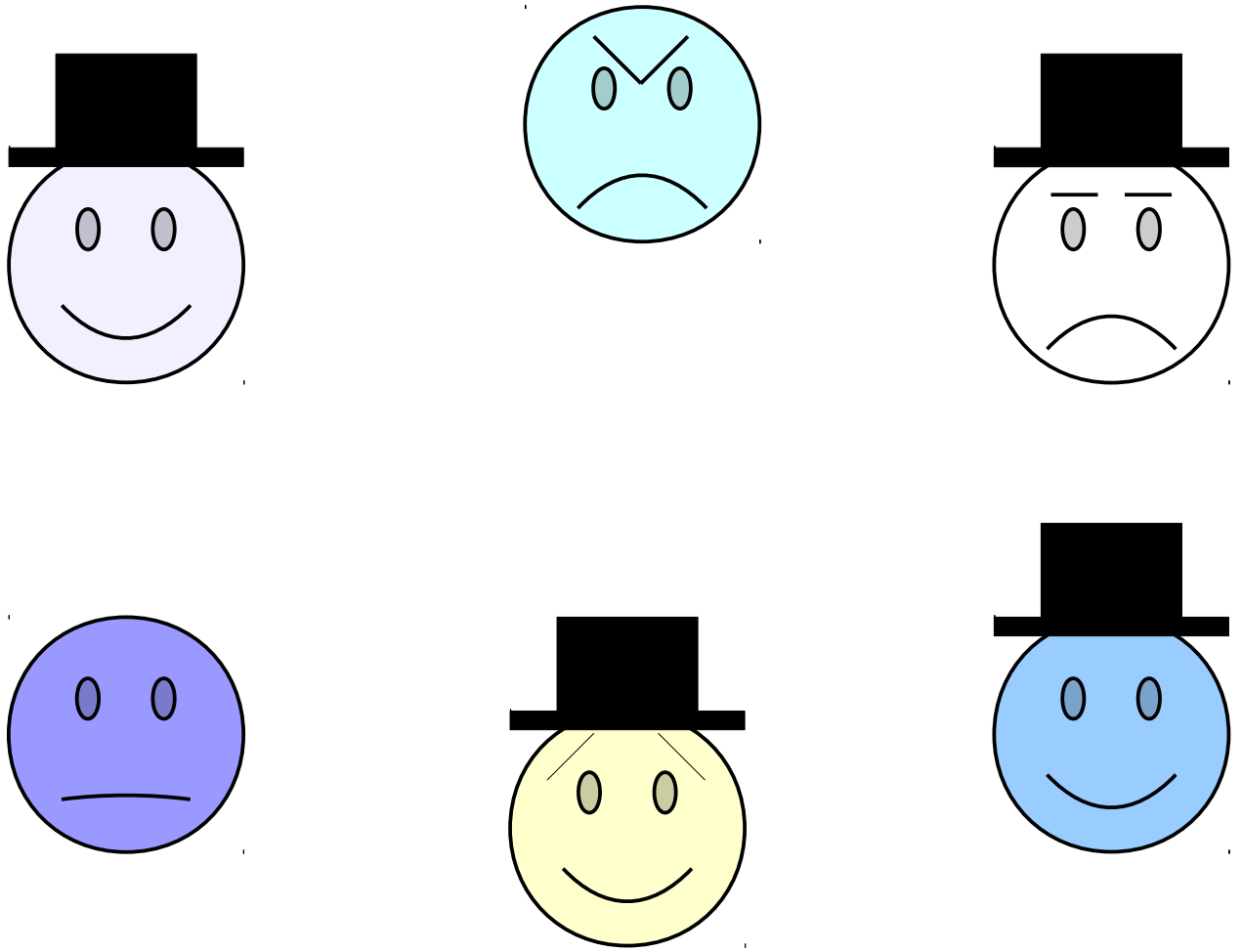


“Every happy person wears a hat.”	***True***
$\forall x.\ (Happy(x) \wedge WearingHat(x))$	***True***
$\forall x.\ (Happy(x) \rightarrow WearingHat(x))$	***True***

“Every happy person wears a hat.”

$$\forall x.\ (Happy(x) \wedge WearingHat(x))$$

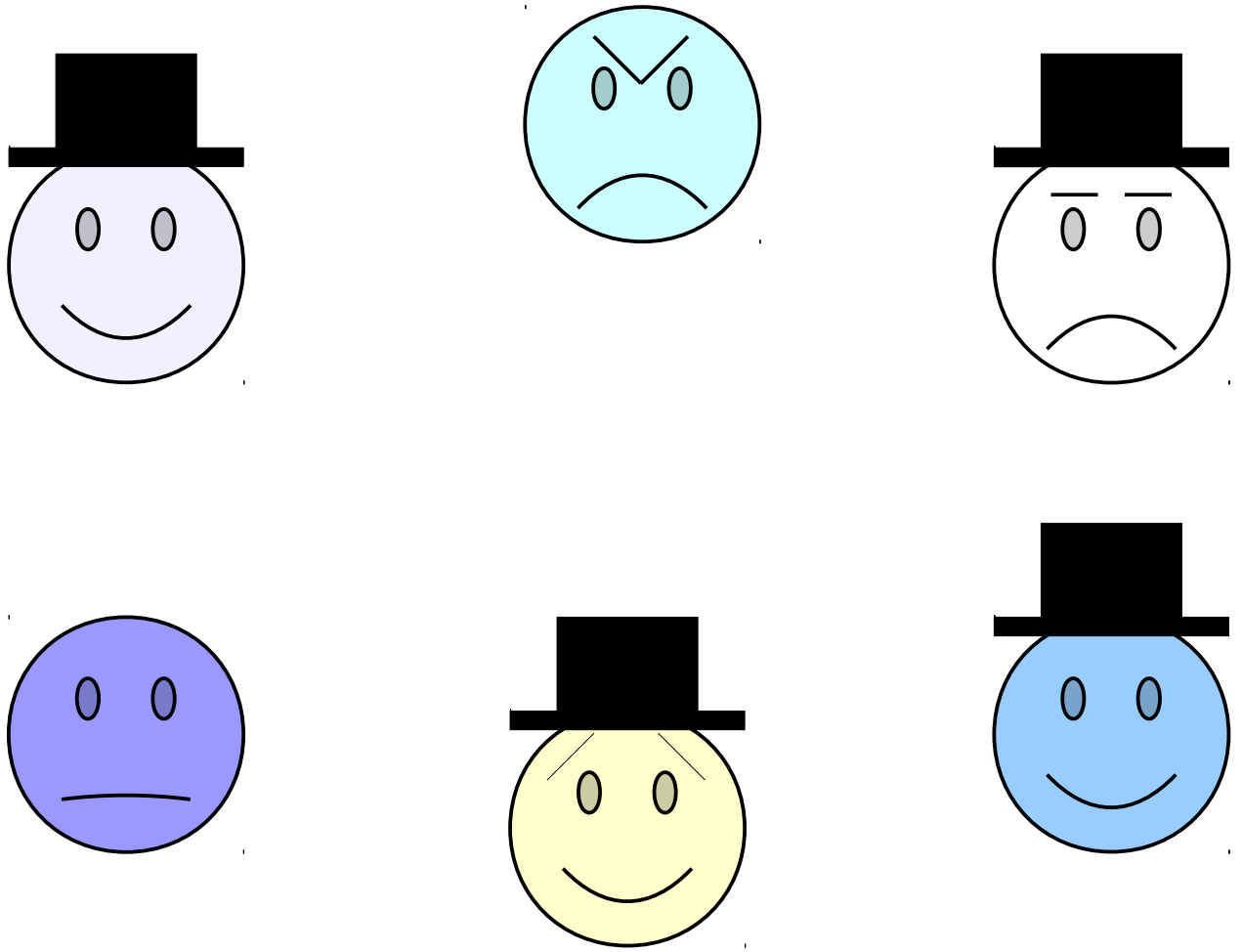
$$\forall x.\ (Happy(x) \rightarrow WearingHat(x))$$



"Every happy person wears a hat."

$\forall x.\ (Happy(x) \wedge WearingHat(x))$

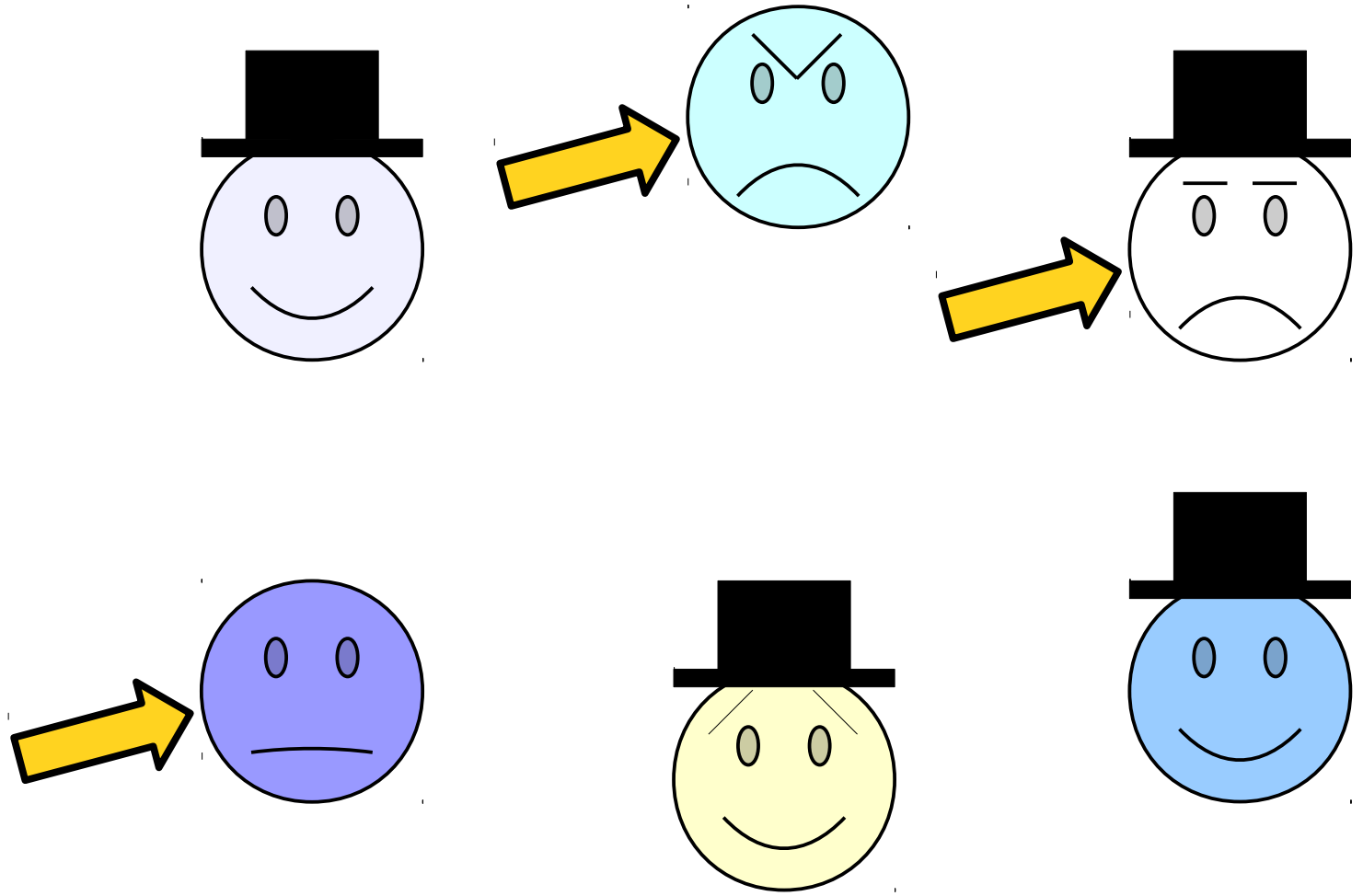
$\forall x.\ (Happy(x) \rightarrow WearingHat(x))$



"Every happy person wears a hat." **True**

$$\forall x.\ (Happy(x) \wedge WearingHat(x))$$

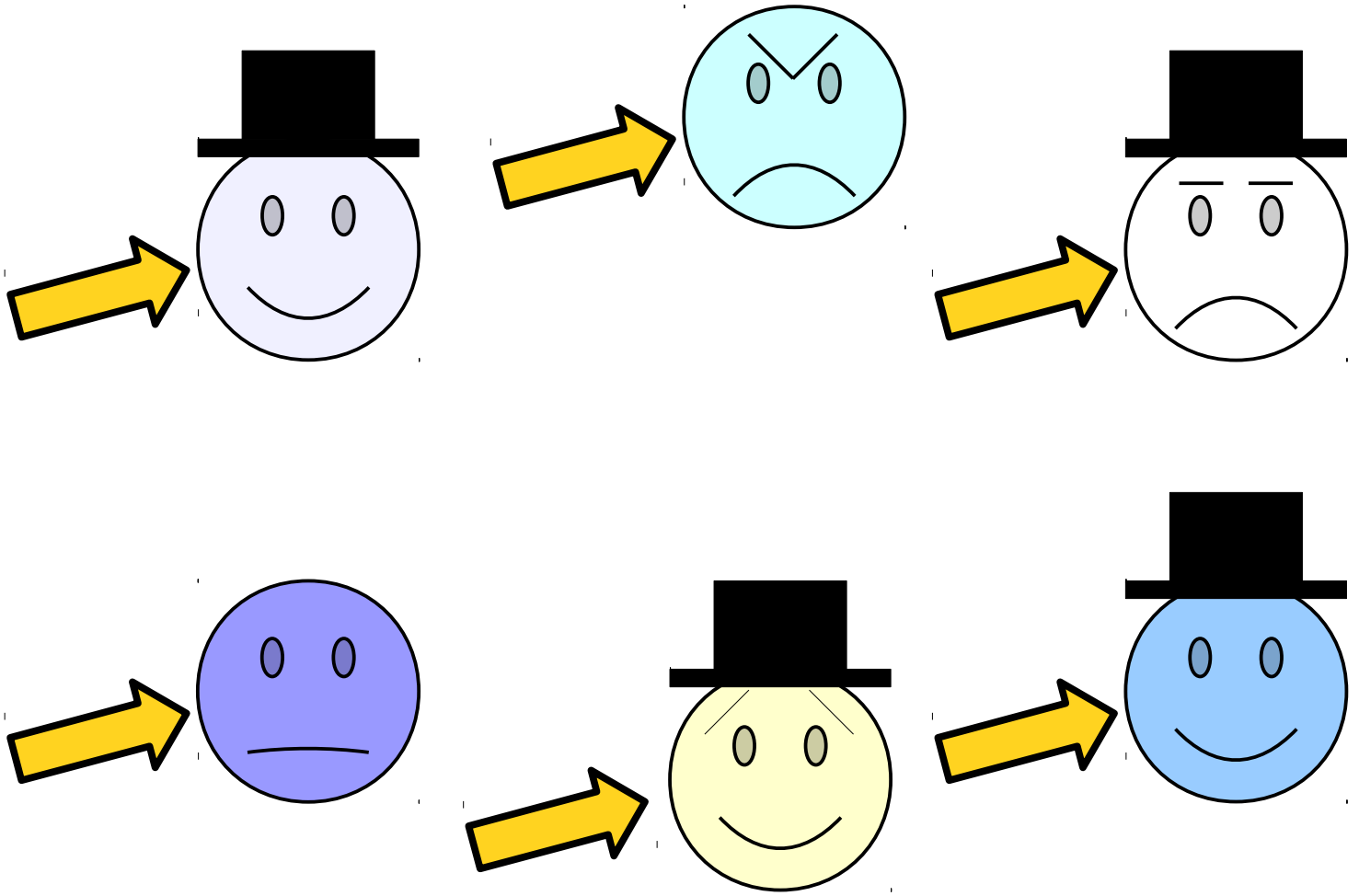
$$\forall x.\ (Happy(x) \rightarrow WearingHat(x))$$



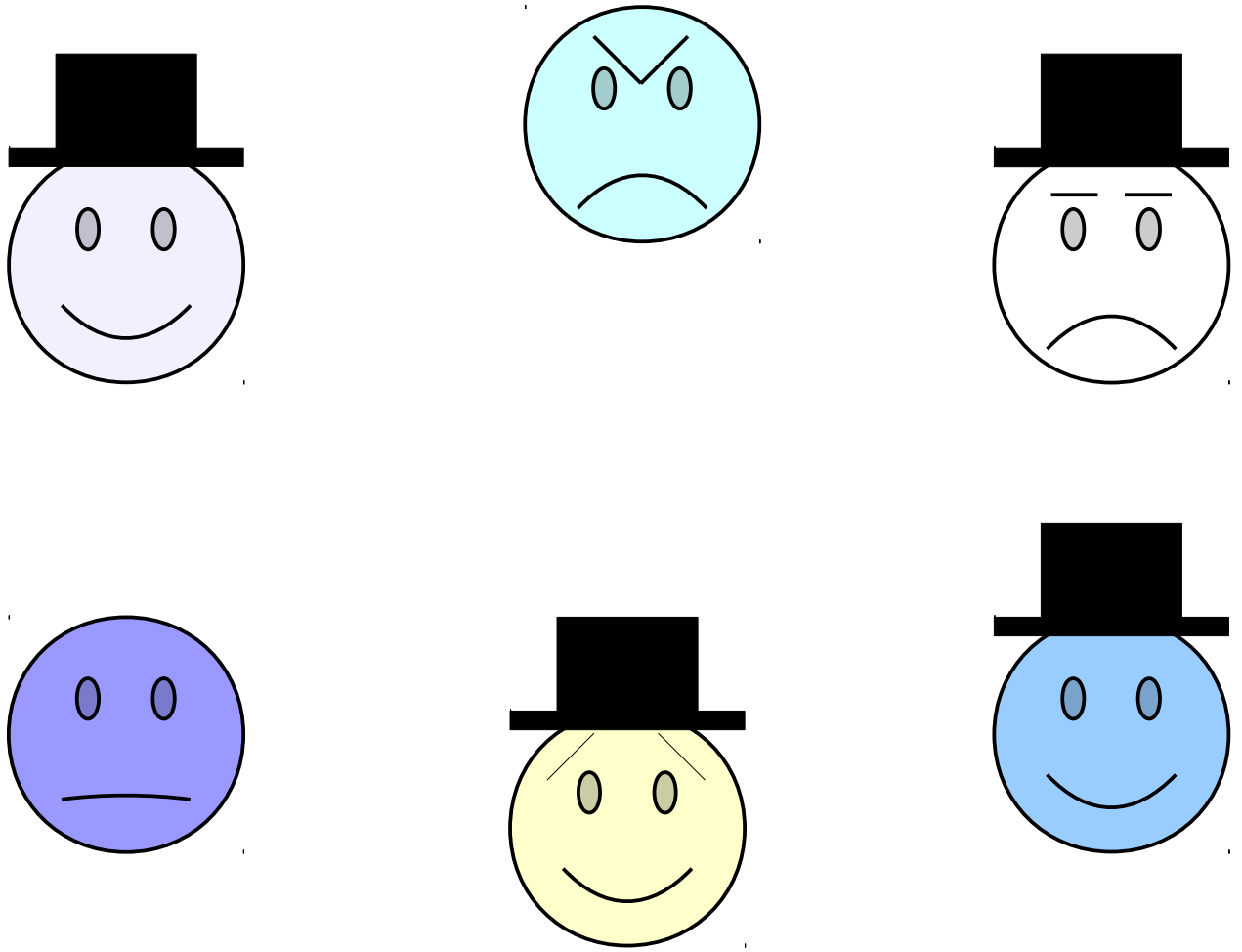
"Every happy person wears a hat." ***True***

$\forall x.\ (Happy(x) \wedge WearingHat(x))$ ***False***

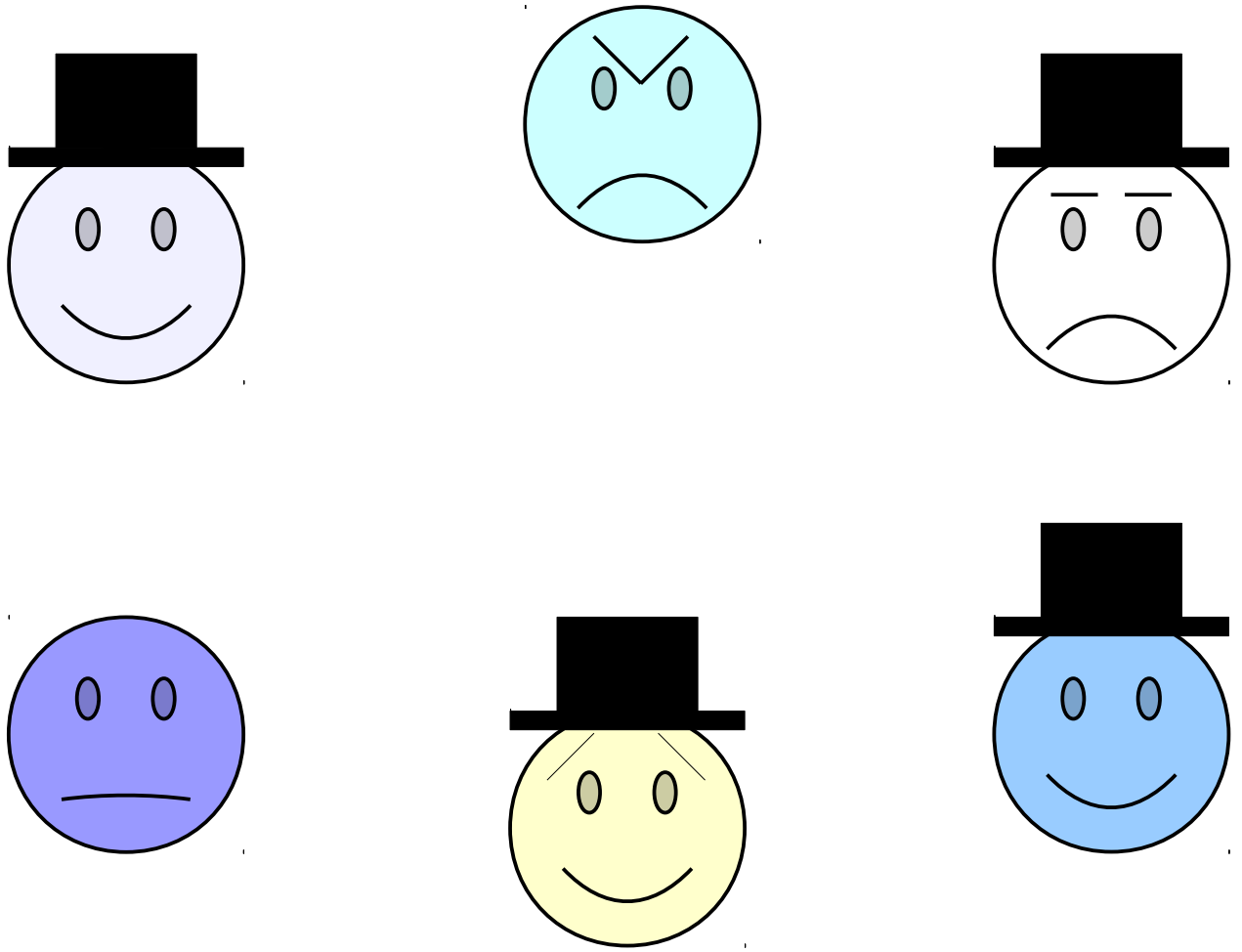
$\forall x.\ (Happy(x) \rightarrow WearingHat(x))$



“Every happy person wears a hat.”	***True***
$\forall x.\ (Happy(x) \wedge WearingHat(x))$	***False***
$\forall x.\ (Happy(x) \rightarrow WearingHat(x))$	***True***



“Every happy person wears a hat.”	***True***
$\forall x.\ (Happy(x) \wedge WearingHat(x))$	***False***
$\forall x.\ (Happy(x) \rightarrow WearingHat(x))$	***True***



“Every happy person wears a hat.”	***True***
~~$\forall x.\ (Happy(x) \wedge WearingHat(x))$~~	***False***
$\forall x.\ (Happy(x) \rightarrow WearingHat(x))$	***True***

“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.

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?