Mathematical Logic

Part One

Question: How do we formalize the definitions and reasoning we use in our proofs?

Where We're Going

Propositional Logic (Today)

- Basic logical connectives.
- Truth tables.
- Logical equivalences.

First-Order Logic (Wednesday/Friday)

- Reasoning about properties of multiple objects.

Propositional Logic

A ***proposition*** is a statement that is, by itself, either true or false.

Some Sample Propositions

- Puppies are cuter than kittens.
- Kittens are cuter than puppies.
- Usain Bolt can outrun everyone in this room.
- CS103 is useful for cocktail parties.
- This is the last entry on this list.

Some Sample Propositions

- I am not throwing away my shot.
- I’m just like my country.
- I’m young, scrappy, and hungry.
- I’m not throwing away my shot.
- I’m ‘a get a scholarship to King’s College.
- I prob’ly shouldn’t brag, but dag, I amaze and astonish.
- The problem is I got a lot of brains but no polish.

Things That Aren't Propositions

Commands cannot be true or false.

Things That Aren't Propositions

Questions cannot be true or false.

Propositional Logic

- ***Propositional logic*** is a mathematical system for reasoning about propositions and how they relate to one another.
- Every statement in propositional logic consists of ***propositional variables*** combined via ***propositional connectives***.
- Each variable represents some proposition, such as “You liked it” or “You should have put a ring on it.”
- Connectives encode how propositions are related, such as “If you liked it, then you should have put a ring on it.”

Propositional Variables

Each proposition will be represented by a ***propositional variable***.

Propositional variables are usually represented as lower-case letters, such as p, q, r, s, etc.

Each variable can take one one of two values: true or false.

Propositional Connectives

There are seven different propositional connectives, many of which will be familiar from programming.

First, there's the logical "NOT" operation:

$$\neg p$$

You'd read this out loud as "not p."

The fancy name for this operation is ***logical negation***.

Propositional Connectives

There are seven different propositional connectives, many of which will be familiar from programming.

Next, there's the logical "AND" operation:

$$p \wedge q$$

You'd read this out loud as "p and q."

The fancy name for this operation is ***logical conjunction***.

Propositional Connectives

There are seven different propositional connectives, many of which will be familiar from programming.

Then, there's the logical "OR" operation:

$$p \vee q$$

You'd read this out loud as "p or q."

The fancy name for this operation is ***logical disjunction***. This is an *inclusive* or.

Truth Tables

A ***truth table*** is a table showing the truth value of a propositional logic formula as a function of its inputs.

Let's go look at the truth tables for the three connectives we've seen so far:

¬ ∧ ∨

Summary of Important Points

The ∨ connective is an *inclusive* "or." It's true if at least one of the operands is true.

Similar to the **||** operator in C, C++, Java, etc. and the **or** operator in Python.

If we need an exclusive "or" operator, we can build it out of what we already have.

Try this yourself! Take a minute to combine these operators together to form an expression that represents the exclusive or of p and q.

Mathematical Implication

Implication

We can represent implications using this connective:

$$\boldsymbol{p \to q}$$

You'd read this out loud as "p implies q."

The fancy name for this is the ***material conditional***.

Question: What should the truth table for $p \to q$ look like?

Pull out a sheet of paper, make a guess, and talk things over with your neighbors!

Why This Truth Table?

The truth values of the → are the way they are because they're *defined* that way.

The intuition:

Every propositional formula should be either true or false – that’s just a guiding design principle behind propositional logic.

We want $p \to q$ to be false only when $p \wedge \neg q$ is true.

In other words, $p \to q$ should be true whenever $\neg(p \wedge \neg q)$ is true.

What's the truth table for $\neg(p \wedge \neg q)$?

Truth Table for Implication

p	q	$p \rightarrow q$
F	F	T
F	T	T
T	F	F
T	T	T

The implication is only false if *p* is true and *q* isn't. It's true otherwise.

Truth Table for Implication

p	q	$p \rightarrow q$
F	F	T
F	T	T
T	F	F
T	T	T

The implication is only false if *p* is true and *q* isn't. It's true otherwise.

You will need to commit this table to memory. We're going to be using it *a lot* over the rest of the week.

Fun Fact: The Contrapositive Revisited

The Biconditional Connective

The Biconditional Connective

On Friday, we saw that "p if and only if q" means both that $p \rightarrow q$ and $q \rightarrow p$.

We can write this in propositional logic using the ***biconditional*** connective:

$$\boldsymbol{p \leftrightarrow q}$$

This connective's truth table has the same meaning as "p implies q and q implies p."

Based on that, what should its truth table look like?

Take a guess, and talk it over with your neighbor!

Biconditionals

The ***biconditional*** connective $p \leftrightarrow q$ is read "p if and only if q."

Here's its truth table:

p	q	$p \leftrightarrow q$
F	F	T
F	T	F
T	F	F
T	T	T

Biconditionals

The ***biconditional*** connective $p \leftrightarrow q$ is read "p if and only if q."

Here's its truth table:

p	q	$p \leftrightarrow q$
F	F	T
F	T	F
T	F	F
T	T	T

One interpretation of $\leftrightarrow$ is to think of it as equality: the two propositions must have equal truth values.

True and False

There are two more “connectives” to speak of: true and false.

The symbol $\top$ is a value that is always true.

The symbol $\bot$ is value that is always false.

These are often called connectives, though they don't connect anything.

(Or rather, they connect zero things.)

Proof by Contradiction

Suppose you want to prove p is true using a proof by contradiction.

The setup looks like this:

- Assume p is false.
- Derive something that we know is false.
- Conclude that p is true.

In propositional logic:

$$\boldsymbol{(\neg p \to \bot) \to p}$$

Operator Precedence

How do we parse this statement?

$$\neg x \to y \vee z \to x \vee y \wedge z$$

Operator precedence for propositional logic:

$\neg$

$\wedge$

$\vee$

$\to$

$\leftrightarrow$

All operators are right-associative.

We can use parentheses to disambiguate.

Operator Precedence

How do we parse this statement?

$$\neg x \to y \vee z \to x \vee y \wedge z$$

Operator precedence for propositional logic:

$\neg$

$\wedge$

$\vee$

$\to$

$\leftrightarrow$

All operators are right-associative.

We can use parentheses to disambiguate.

Operator Precedence

How do we parse this statement?

$$(\neg x) \to y \vee z \to x \vee y \wedge z$$

Operator precedence for propositional logic:

$\neg$

$\wedge$

$\vee$

$\to$

$\leftrightarrow$

All operators are right-associative.

We can use parentheses to disambiguate.

Operator Precedence

How do we parse this statement?

$$(\neg x) \to y \vee z \to x \vee y \wedge z$$

Operator precedence for propositional logic:

$\neg$

$\wedge$

$\vee$

$\to$

$\leftrightarrow$

All operators are right-associative.

We can use parentheses to disambiguate.

Operator Precedence

How do we parse this statement?

$$(\neg x) \to y \vee z \to x \vee (y \wedge z)$$

Operator precedence for propositional logic:

$\neg$

$\wedge$

$\vee$

$\to$

$\leftrightarrow$

All operators are right-associative.

We can use parentheses to disambiguate.

Operator Precedence

How do we parse this statement?

$$(\neg x) \to y \vee z \to x \vee (y \wedge z)$$

Operator precedence for propositional logic:

$\neg$

$\wedge$

$\vee$

$\to$

$\leftrightarrow$

All operators are right-associative.

We can use parentheses to disambiguate.

Operator Precedence

How do we parse this statement?

$$(\neg x) \rightarrow (y \vee z) \rightarrow (x \vee (y \wedge z))$$

Operator precedence for propositional logic:

$\neg$

$\wedge$

$\vee$

$\rightarrow$

$\leftrightarrow$

All operators are right-associative.

We can use parentheses to disambiguate.

Operator Precedence

How do we parse this statement?

$$(\neg x) \rightarrow (y \vee z) \rightarrow (x \vee (y \wedge z))$$

Operator precedence for propositional logic:

$\neg$

$\wedge$

$\vee$

$\rightarrow$

$\leftrightarrow$

All operators are right-associative.

We can use parentheses to disambiguate.

Operator Precedence

How do we parse this statement?

$$(\neg x) \rightarrow ((y \vee z) \rightarrow (x \vee (y \wedge z)))$$

Operator precedence for propositional logic:

$\neg$

$\wedge$

$\vee$

$\rightarrow$

$\leftrightarrow$

All operators are right-associative.

We can use parentheses to disambiguate.

Operator Precedence

How do we parse this statement?

$$(\neg x) \rightarrow ((y \vee z) \rightarrow (x \vee (y \wedge z)))$$

Operator precedence for propositional logic:

$\neg$

$\wedge$

$\vee$

$\rightarrow$

$\leftrightarrow$

All operators are right-associative.

We can use parentheses to disambiguate.

Operator Precedence

The main points to remember:

- ¬ binds to whatever immediately follows it.
- ∧ and ∨ bind more tightly than →.

We will commonly write expressions like $p \wedge q \rightarrow r$ without adding parentheses.

For more complex expressions, we'll try to add parentheses.

Confused? ***Please ask!***

The Big Table

Connective	Read As	C++ Version	Fancy Name
¬	“not”	!	Negation
∧	“and”	&&	Conjunction
∨	“or”	||	Disjunction
→	“implies”		Implication
↔	“if and only if”		Biconditional
⊤	“true”	`true`	Truth
⊥	“false”	`false`	Falsity

Time-Out for Announcements!

Problem Set One

The checkpoint problem for PS1 was due at 11:59PM last night.

We'll try to have it graded and returned by Wednesday morning.

The remaining problems from PS1 are due on Thursday at 11:59PM.

Have questions? Stop by office hours or ask on Campuswire!

Back to CS103!

Recap So Far

A ***propositional variable*** is a variable that is either true or false.

The ***propositional connectives*** are

- Negation: $\neg p$
- Conjunction: $p \wedge q$
- Disjunction: $p \vee q$
- Implication: $p \to q$
- Biconditional: $p \leftrightarrow q$
- True: $\top$
- False: $\bot$

Translating into Propositional Logic

Some Sample Propositions

a: Aang will be in the path of totality during the solar eclipse.

b: Aang will defeat the firelord.

Some Sample Propositions

a: Aang will be in the path of totality during the solar eclipse.

b: Aang will defeat the firelord.

"Aang won't defeat the firelord if Aang is not in the path of totality during the solar eclipse."

Some Sample Propositions

a: Aang will be in the path of totality during the solar eclipse.

b: Aang will defeat the firelord.

"Aang won't defeat the firelord if Aang is not in the path of totality during the solar eclipse."

$$\neg a \rightarrow \neg b$$

“*p* if *q*”

translates to

$$q \rightarrow p$$

It does *not* translate to

⚠ $p \rightarrow q$ ⚠

Some Sample Propositions

a: Aang will be in the path of totality during the solar eclipse.

b: Aang will defeat the firelord.

c: The plan goes smoothly.

Some Sample Propositions

a: Aang will be in the path of totality during the solar eclipse.

b: Aang will defeat the firelord.

c: The plan goes smoothly.

"If Aang will be in the path of totality, but the plan does not go smoothly, Aang won't defeat the firelord."

Some Sample Propositions

a: Aang will be in the path of totality during the solar eclipse.

b: Aang will defeat the firelord.

c: The plan went smoothly.

"If Aang will be in the path of totality, but the plan does not go smoothly, Aang won't defeat the firelord."

$$a \wedge \neg c \rightarrow \neg b$$

“***p*, but *q***”

translates to

$$p \wedge q$$

The Takeaway Point

When translating into or out of propositional logic, be very careful not to get tripped up by nuances of the English language.

In fact, this is one of the reasons we have a symbolic notation in the first place!

Many prepositional phrases lead to counterintuitive translations; make sure to double-check yourself!

Propositional Equivalences

Quick Question:

What would I have to show you to convince you that the statement $p \wedge q$ is false?

Quick Question:

What would I have to show you to convince you that the statement $p \vee q$ is false?

de Morgan's Laws

Using truth tables, we concluded that

$$\neg(p \wedge q)$$

is equivalent to

$$\neg p \vee \neg q$$

We also saw that

$$\neg(p \vee q)$$

is equivalent to

$$\neg p \wedge \neg q$$

These two equivalences are called ***De Morgan's Laws***.

de Morgan's Laws in Code

Pro tip: Don't write this:

```
if (!(p() && q())) {
    /* ... */
}
```

Write this instead:

```
if (!p() || !q()) {
    /* ... */
}
```

(This even short-circuits correctly!)

Logical Equivalence

Because $\neg(p \wedge q)$ and $\neg p \vee \neg q$ have the same truth tables, we say that they're ***equivalent*** to one another.

We denote this by writing

$$\neg(p \wedge q) \equiv \neg p \vee \neg q$$

The $\equiv$ symbol is not a connective.

The statement $\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$ is a propositional formula. If you plug in different values of p and q, it will evaluate to a truth value. It just happens to evaluate to true every time.

The statement $\neg(p \wedge q) \equiv \neg p \vee \neg q$ means "these two formulas have exactly the same truth table."

In other words, the notation $\varphi \equiv \psi$ means "φ and ψ always have the same truth values, regardless of how the variables are assigned."

An Important Equivalence

Earlier, we talked about the truth table for $p \to q$. We chose it so that

$$\boldsymbol{p \to q \quad \equiv \quad \neg(p \wedge \neg q)}$$

Later on, this equivalence will be incredibly useful:

$$\boldsymbol{\neg(p \to q) \quad \equiv \quad p \wedge \neg q}$$

Another Important Equivalence

Here's a useful equivalence. Start with

$$p \to q \equiv \neg(p \wedge \neg q)$$

By De Morgan's laws:

$$\begin{aligned} p \to q &\equiv \neg(p \wedge \neg q) \\ &\equiv \neg p \vee \neg\neg q \\ &\equiv \neg p \vee q \end{aligned}$$

Thus $p \to q \equiv \neg p \vee q$

Another Important Equivalence

Here's a useful equivalence. Start with

$$p \to q \equiv \neg(p \wedge \neg q)$$

By De Morgan's laws:

$$p \to q \equiv \neg(p \wedge \neg q)$$

$$\equiv \neg p \vee \neg\neg q$$

$$\equiv$$

If p is false, then $\neg p \vee q$ is true. If p is true, then q has to be true for the whole expression to be true.

Thus $p \to q \equiv \neg p \vee q$

Why All This Matters

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$x + y = 16 \rightarrow x \geq 8 \vee y \geq 8$$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$x + y = 16 \rightarrow x \geq 8 \vee y \geq 8$$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$\neg(x \geq 8 \vee y \geq 8) \rightarrow \neg(x + y = 16)$$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$\neg(x \geq 8 \vee y \geq 8) \rightarrow \neg(x + y = 16)$$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$\neg(x \geq 8 \vee y \geq 8) \rightarrow \neg(x + y = 16)$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$\neg(x \geq 8 \vee y \geq 8) \rightarrow x + y \neq 16$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$\neg(x \geq 8 \vee y \geq 8) \rightarrow x + y \neq 16$$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$\neg(x \geq 8 \vee y \geq 8) \rightarrow x + y \neq 16$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$\neg(x \geq 8) \wedge \neg(y \geq 8) \rightarrow x + y \neq 16$$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$\neg(x \geq 8) \wedge \neg(y \geq 8) \rightarrow x + y \neq 16$$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$\neg(x \geq 8) \wedge \neg(y \geq 8) \rightarrow x + y \neq 16$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$x < 8 \wedge \neg(y \geq 8) \rightarrow x + y \neq 16$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$x < 8 \wedge \neg(y \geq 8) \rightarrow x + y \neq 16$$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$x < 8 \wedge \neg(y \geq 8) \rightarrow x + y \neq 16$$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$x < 8 \wedge y < 8 \rightarrow x + y \neq 16$$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$x < 8 \wedge y < 8 \rightarrow x + y \neq 16$$

Why All This Matters

Suppose we want to prove the following statement:

"If $x + y = 16$, then $x \geq 8$ or $y \geq 8$"

$$x < 8 \wedge y < 8 \rightarrow x + y \neq 16$$

"If $x < 8$ and $y < 8$, then $x + y \neq 16$"

Theorem: If $x + y = 16$, then $x \geq 8$ or $y \geq 8$.

Proof: By contrapositive. We will prove that if $x < 8$ and $y < 8$, then $x + y \neq 16$. Let x and y be arbitrary numbers such that $x < 8$ and $y < 8$.

Note that

$$\begin{aligned} x + y &< 8 + y \\ &< 8 + 8 \\ &= 16. \end{aligned}$$

This means that $x + y < 16$, so $x + y \neq 16$, which is what we needed to show. ■

Why This Matters

Propositional logic is a tool for reasoning about how various statements affect one another.

To better understand how to prove a result, it often helps to translate what you're trying to prove into propositional logic first.

That said, propositional logic isn't expressive enough to capture all statements. For that, we need something more powerful.

Let’s take a five minute break!

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*.

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

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

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

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

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

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

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:

$$\textit{Tom Marvolo Riddle} = \textit{Lord Voldemort}$$

$$\textit{Morning Star} = \textit{Evening Star}$$

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

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

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

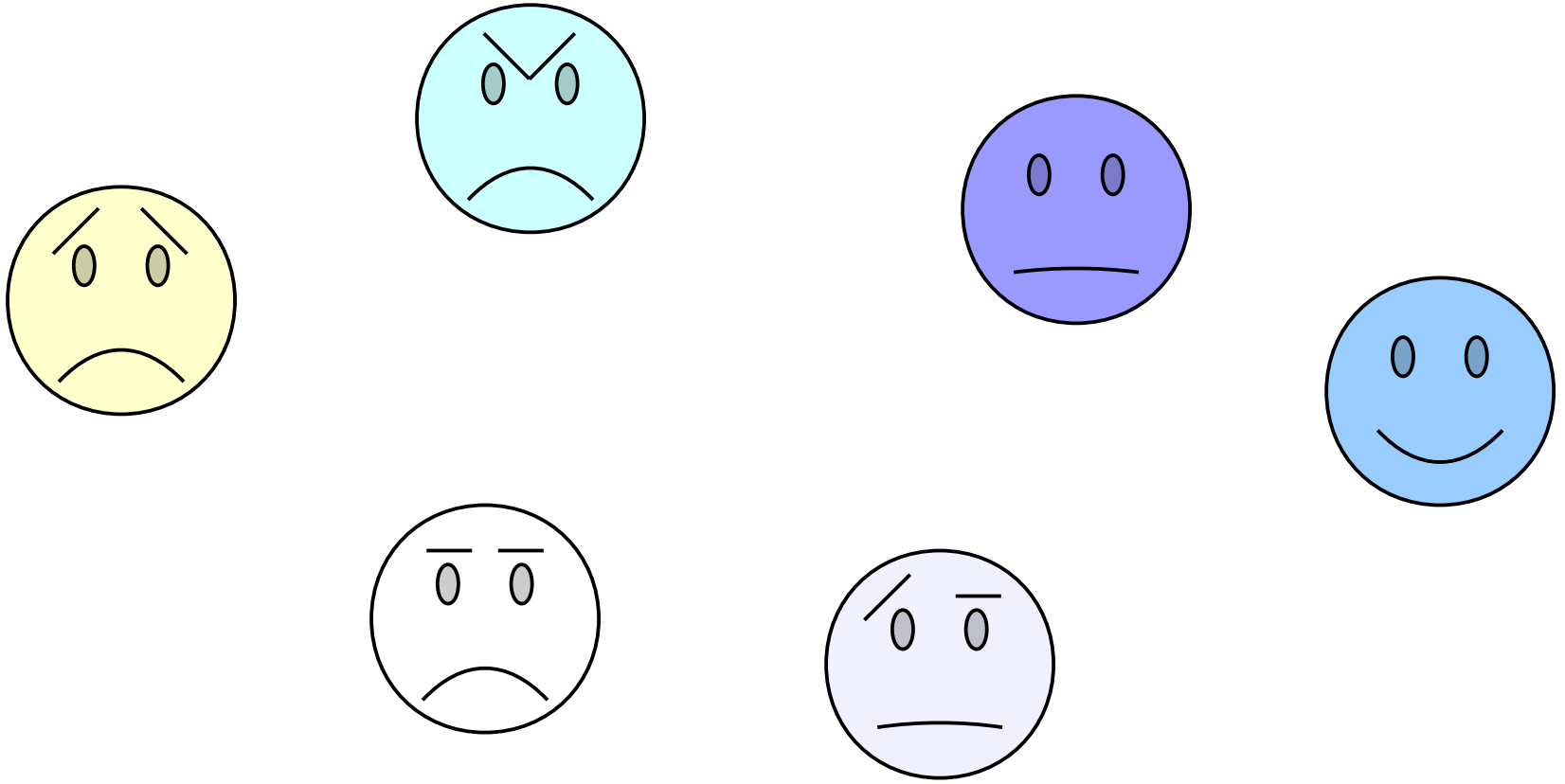
Examples:

$\exists x.\ (\mathit{Even}(x) \wedge \mathit{Prime}(x))$

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

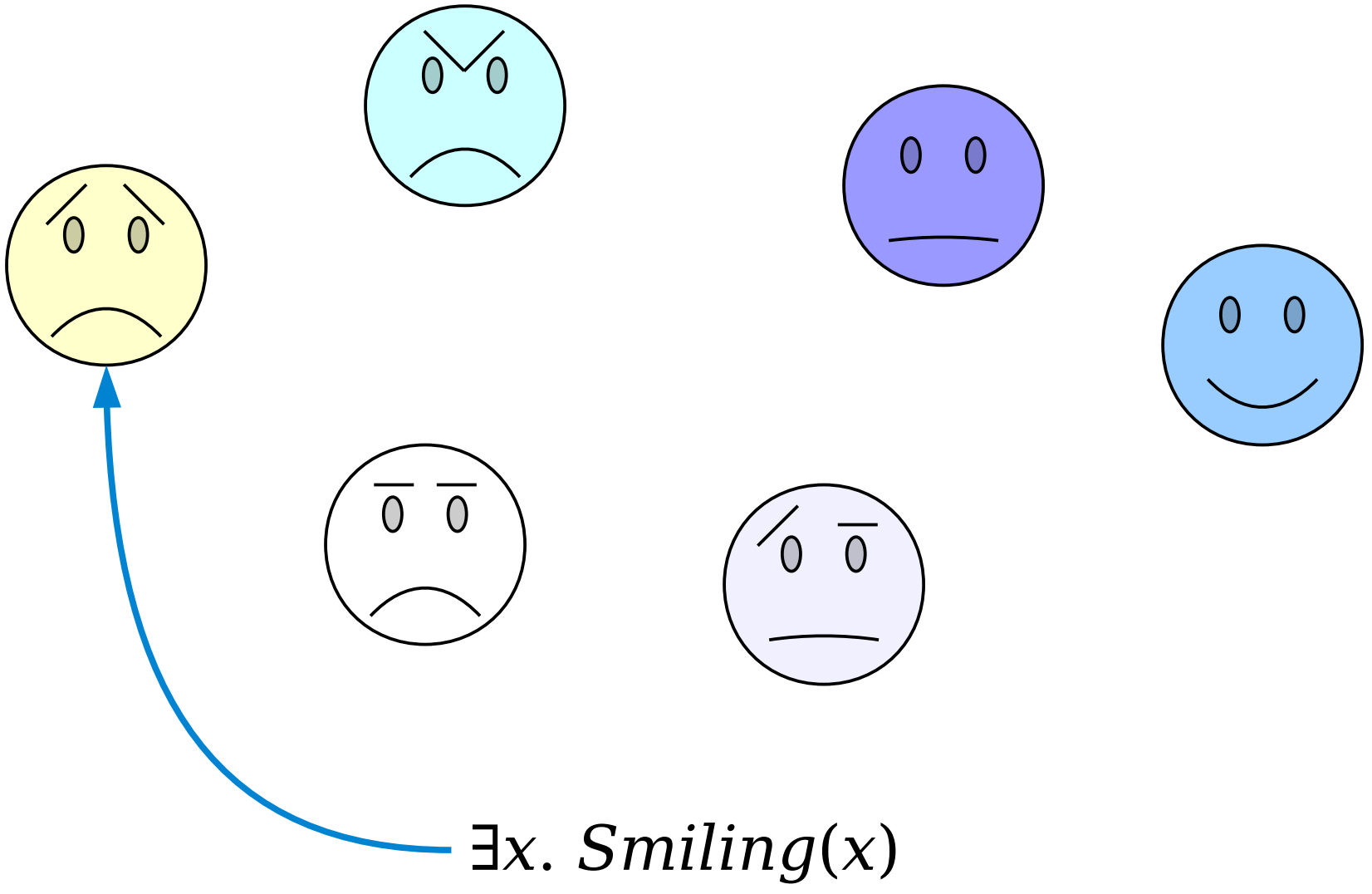
$(\exists w.\ \mathit{Will}(w)) \rightarrow (\exists x.\ \mathit{Way}(x))$

The Existential Quantifier

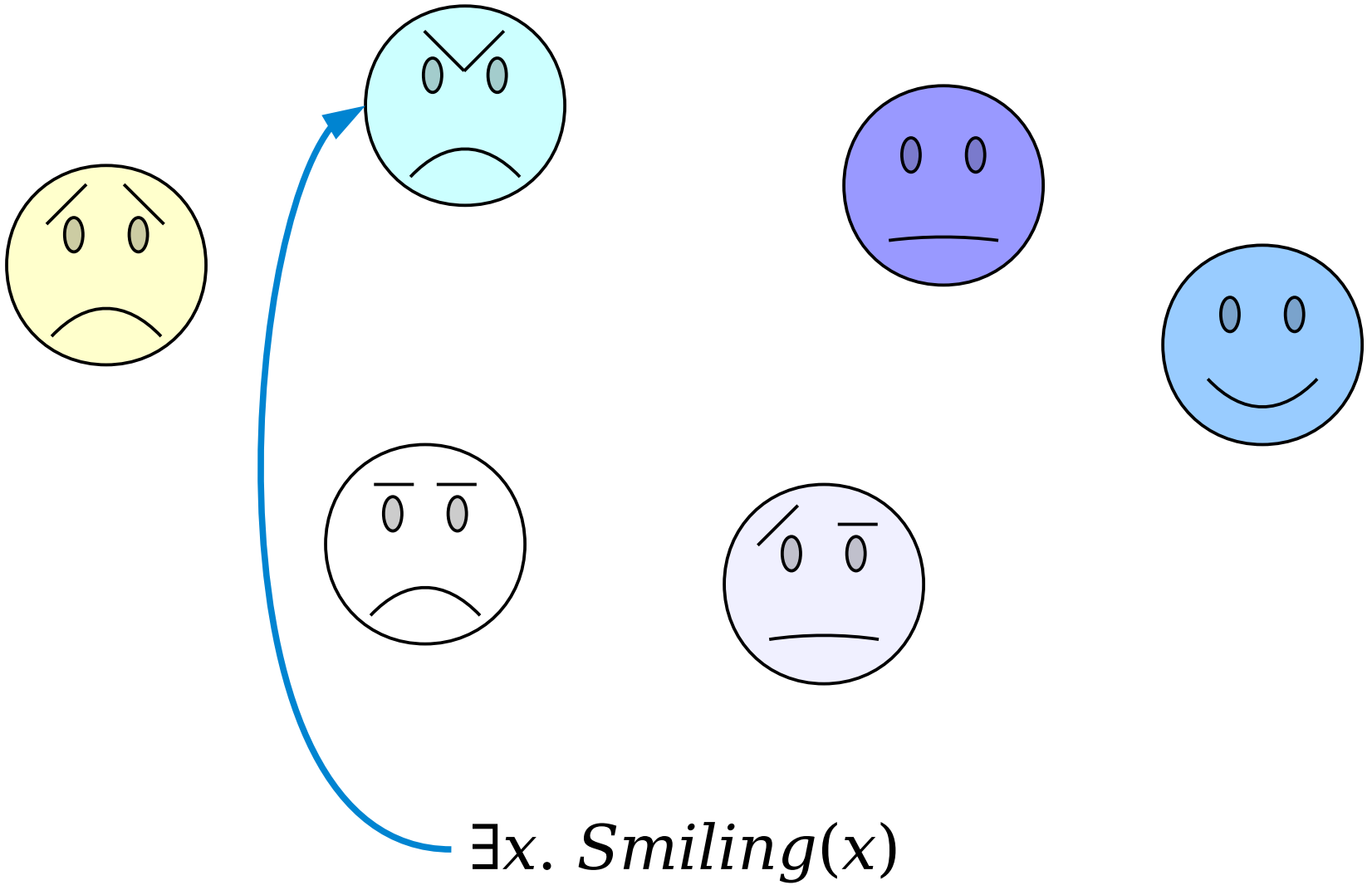


$\exists x.\ \mathit{Smiling}(x)$

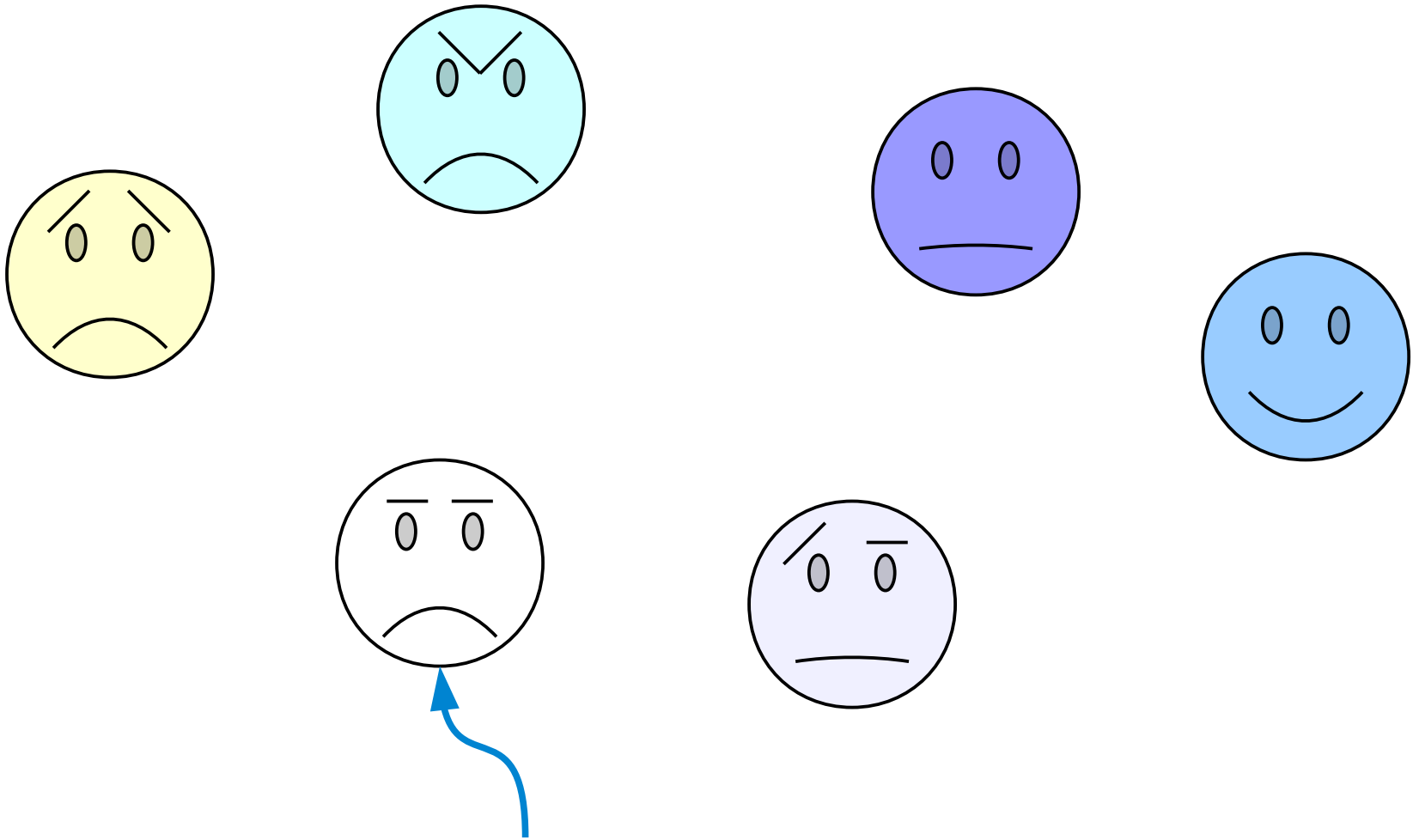
The Existential Quantifier



The Existential Quantifier

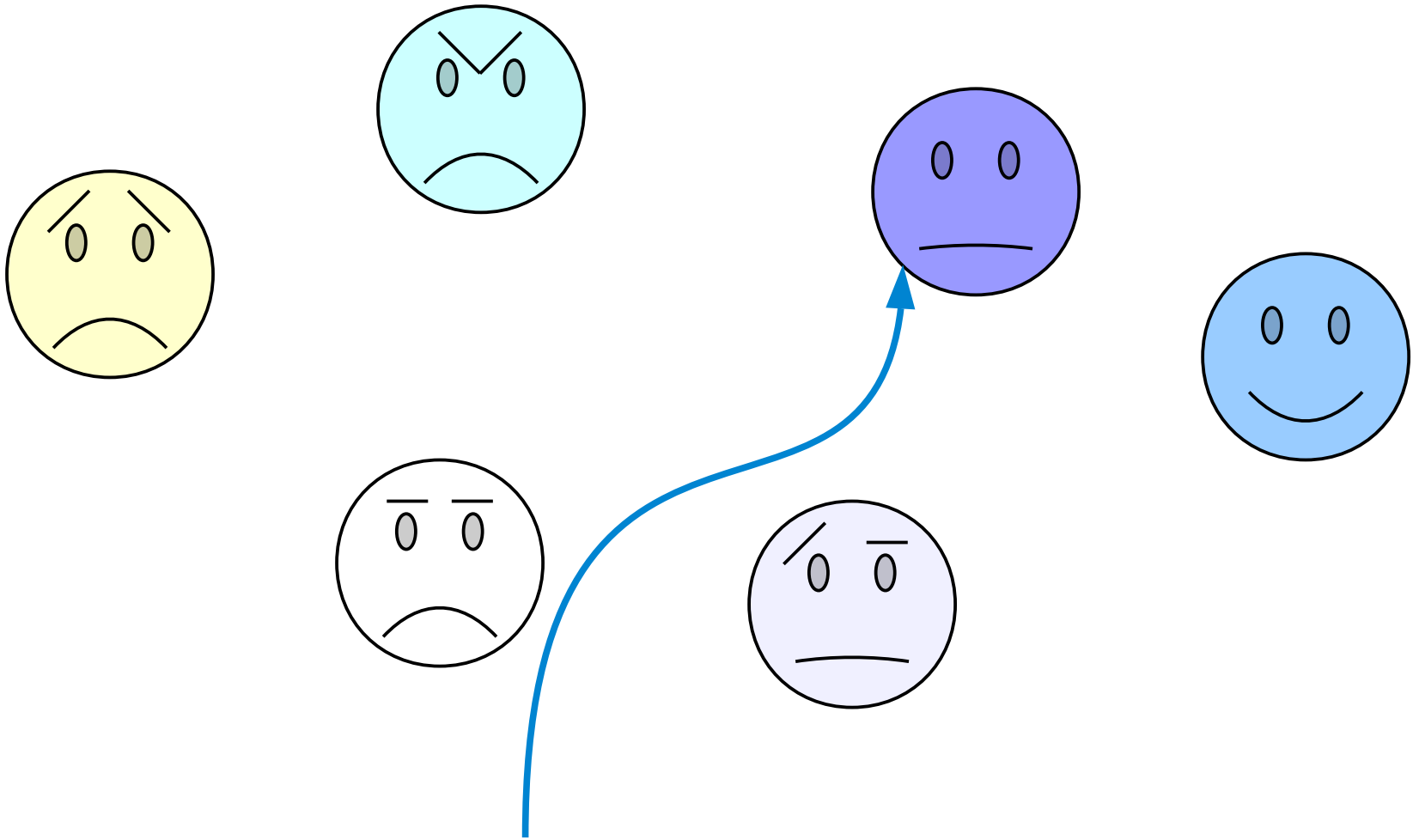


The Existential Quantifier



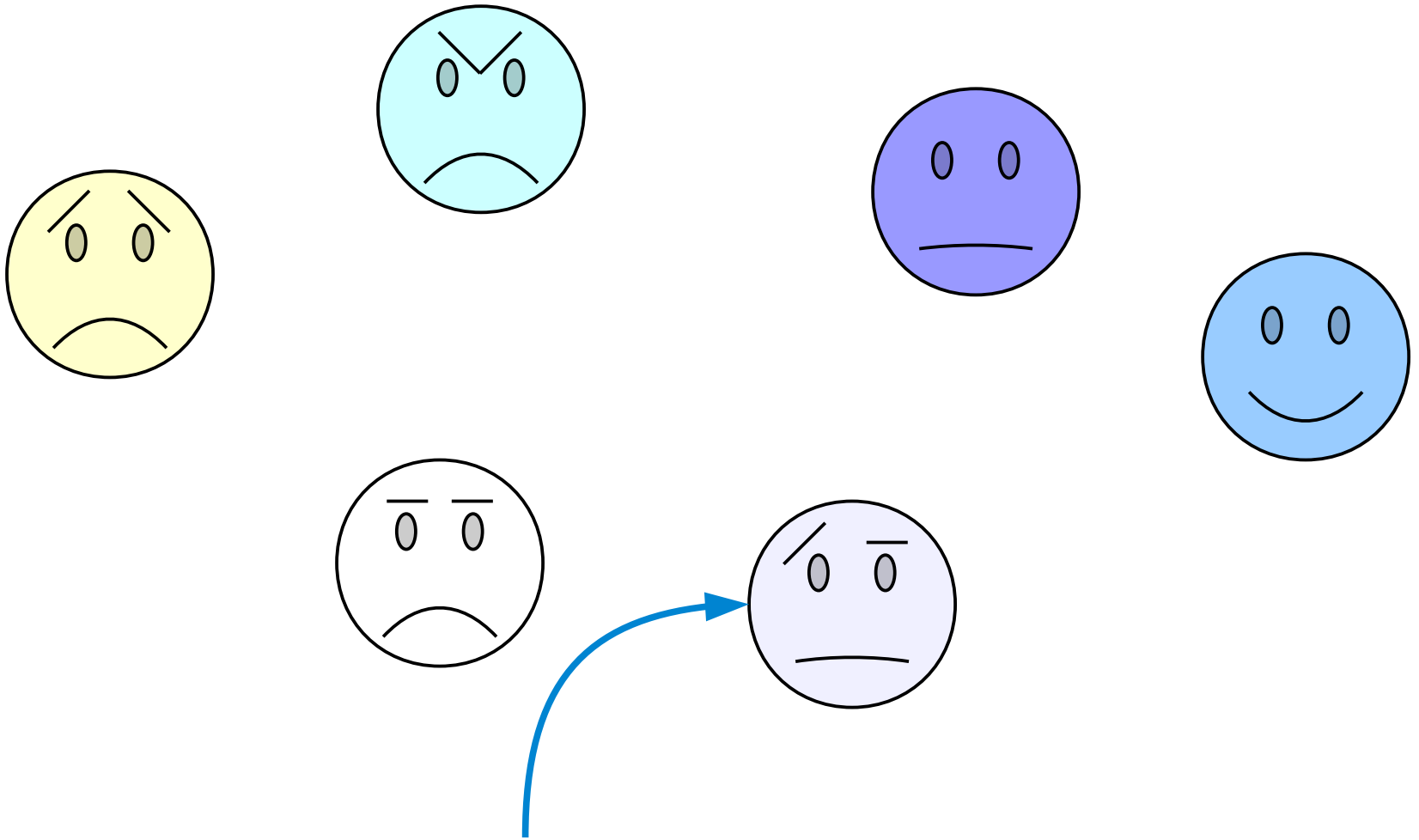
$\exists x.\ \mathit{Smiling}(x)$

The Existential Quantifier



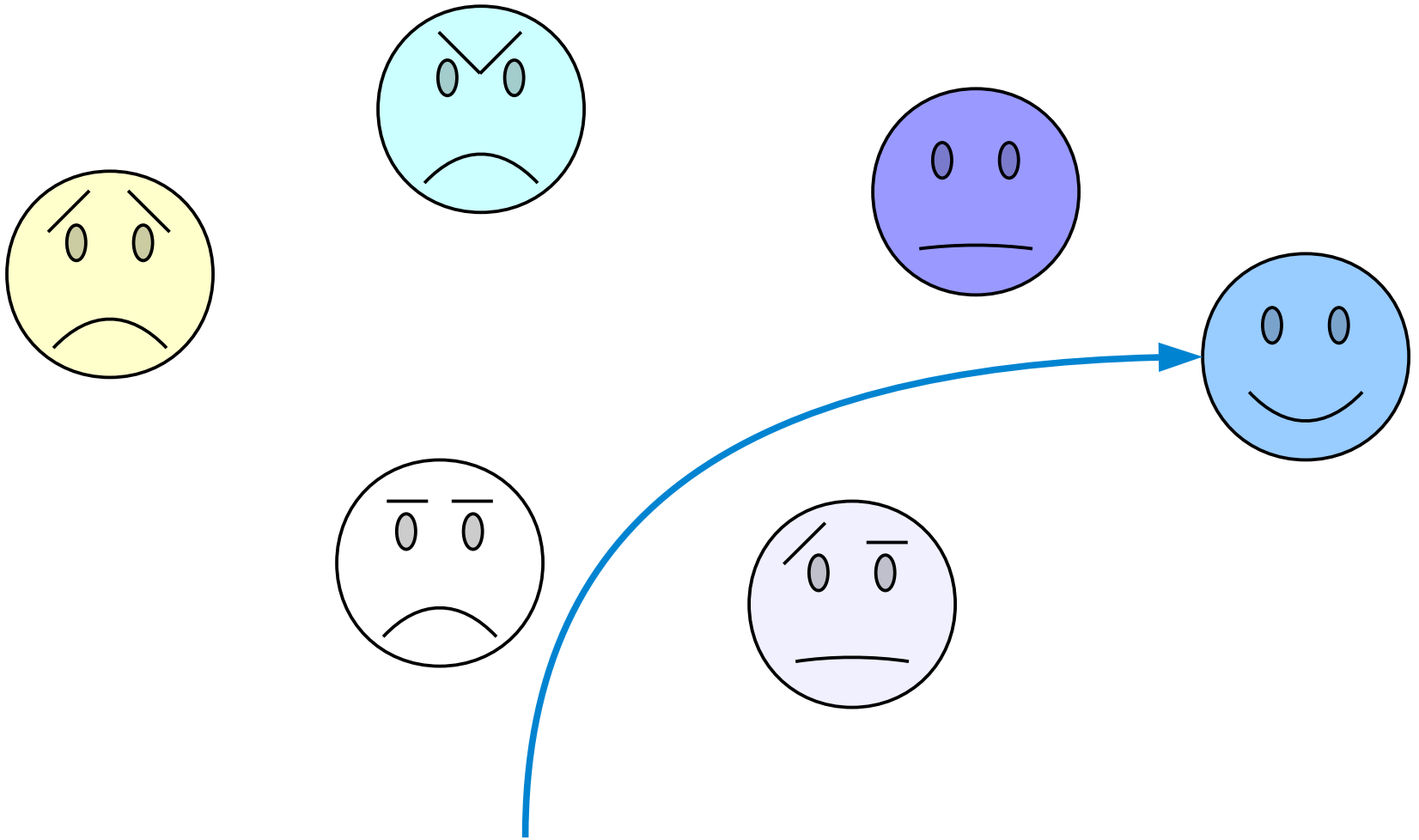
$\exists x.\ \mathit{Smiling}(x)$

The Existential Quantifier



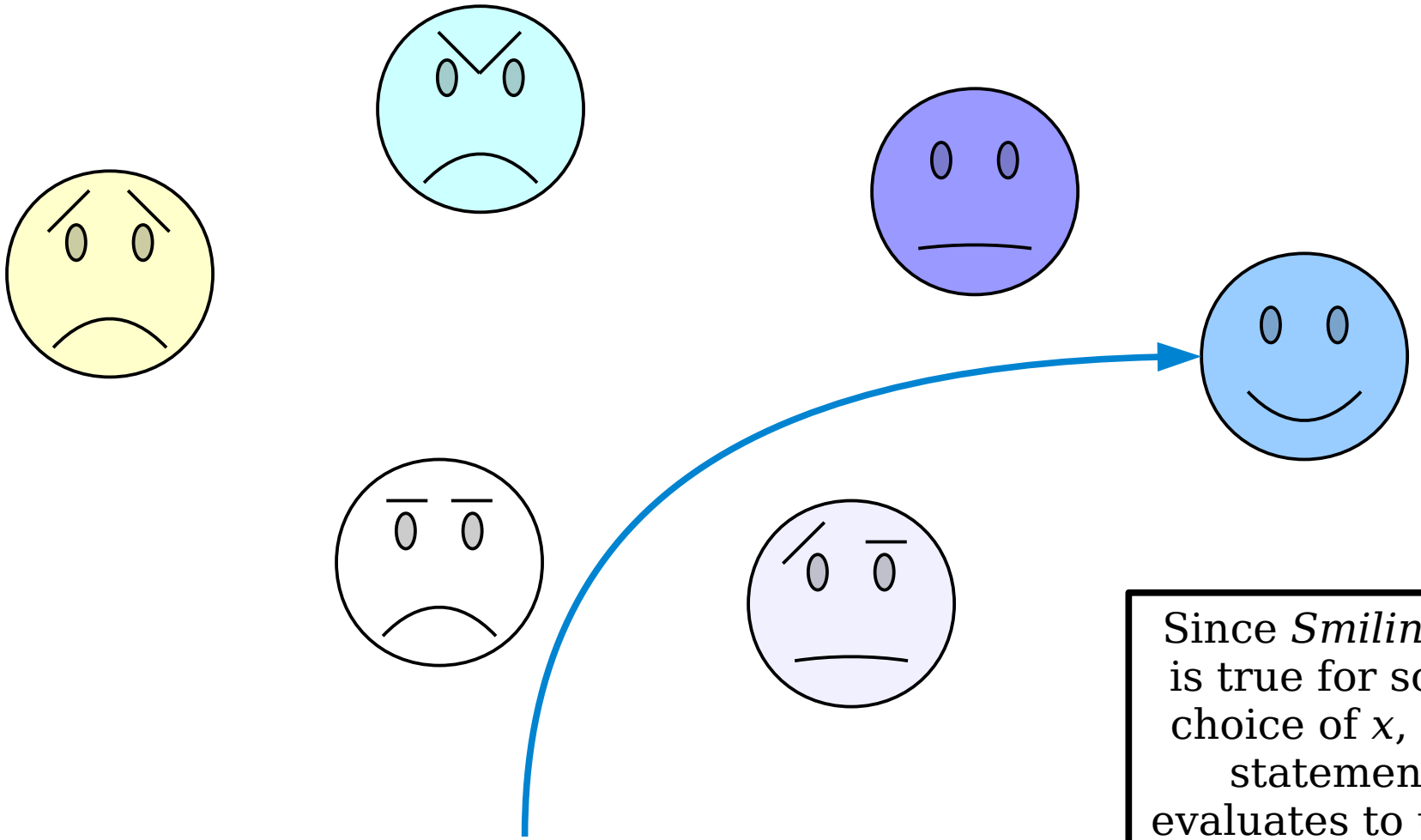
$\exists x.\ \mathit{Smiling}(x)$

The Existential Quantifier



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

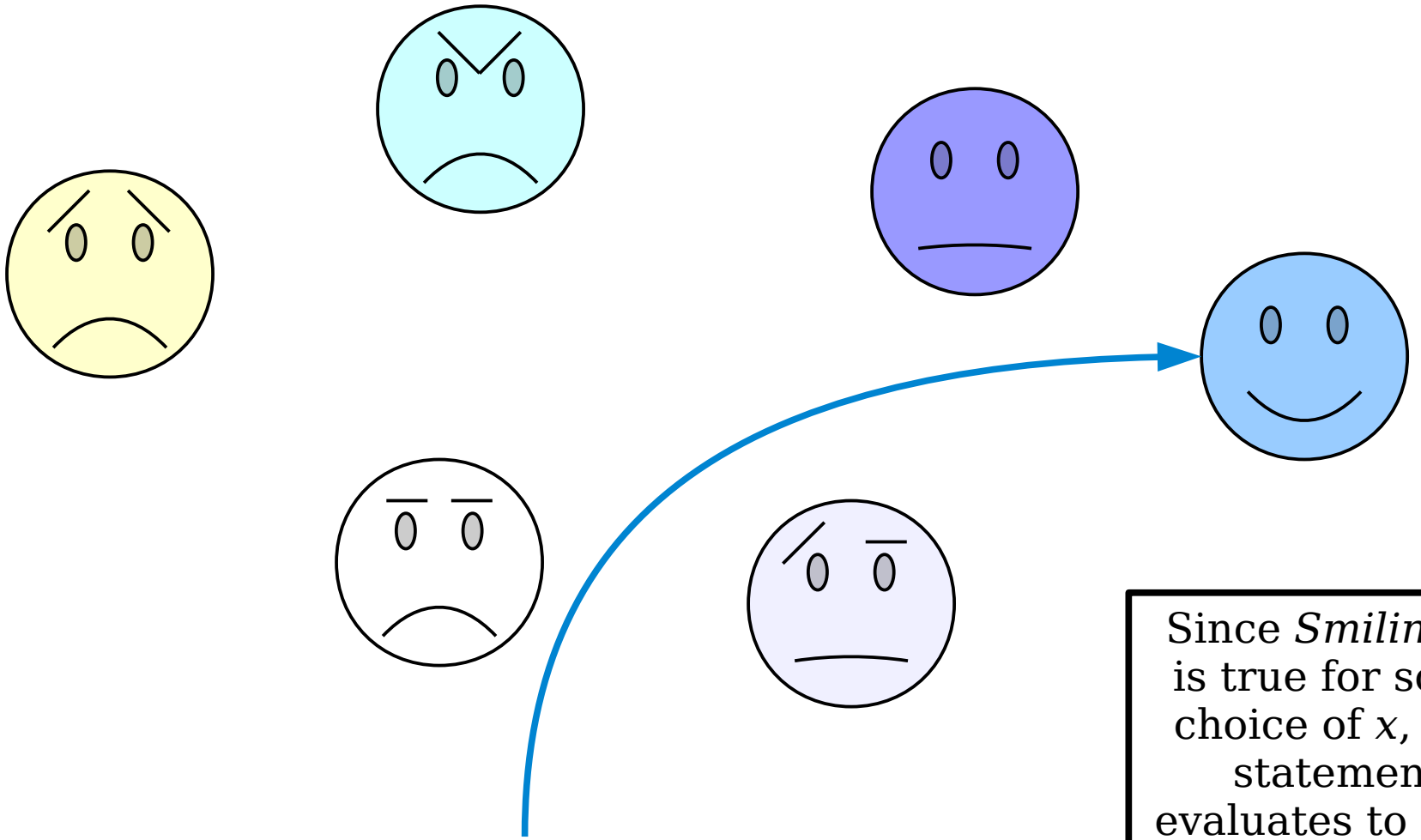
The Existential Quantifier



$\exists x.\ Smiling(x)$

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

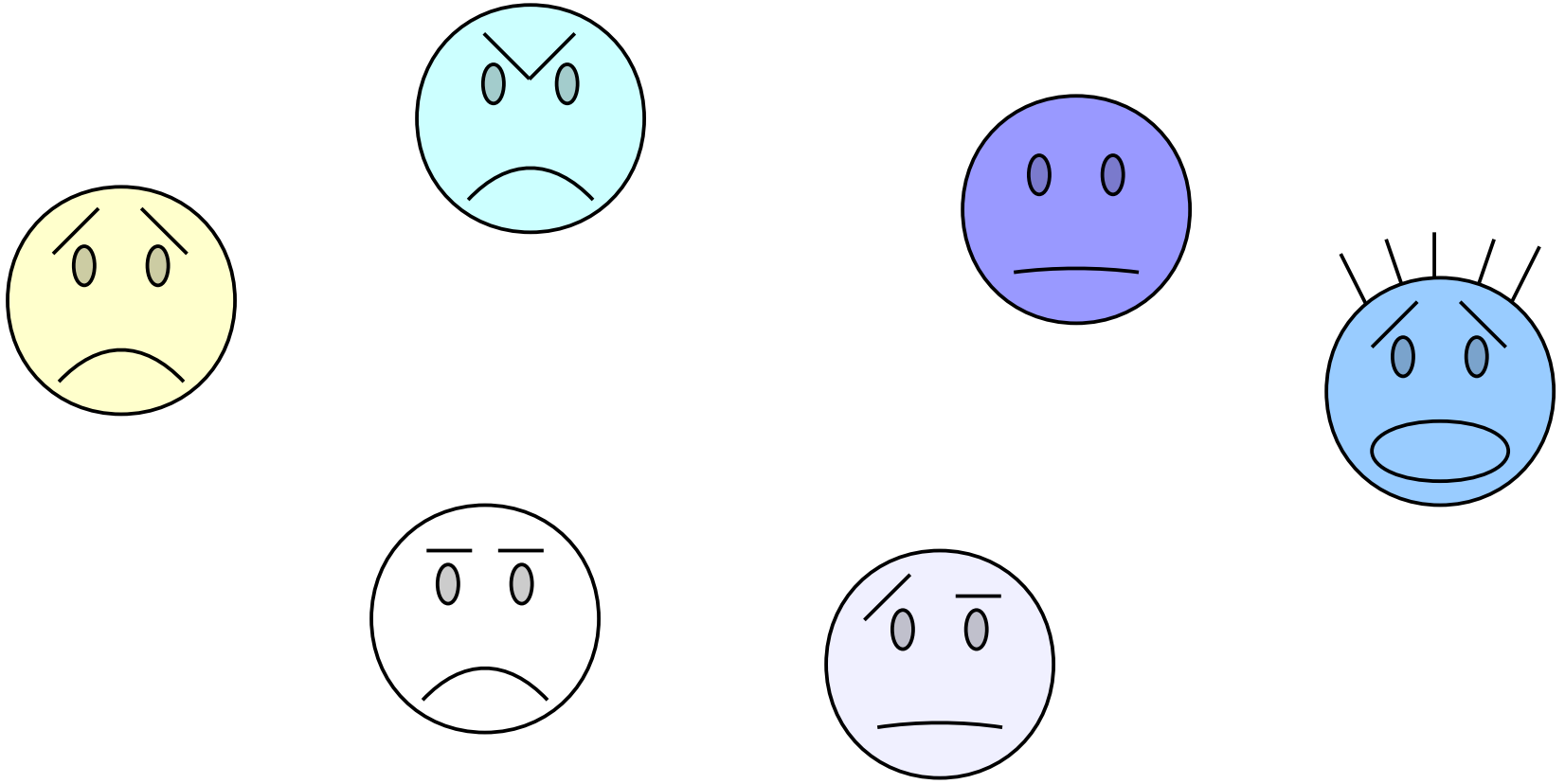
The Existential Quantifier



$\exists x.\ Smiling(x)$

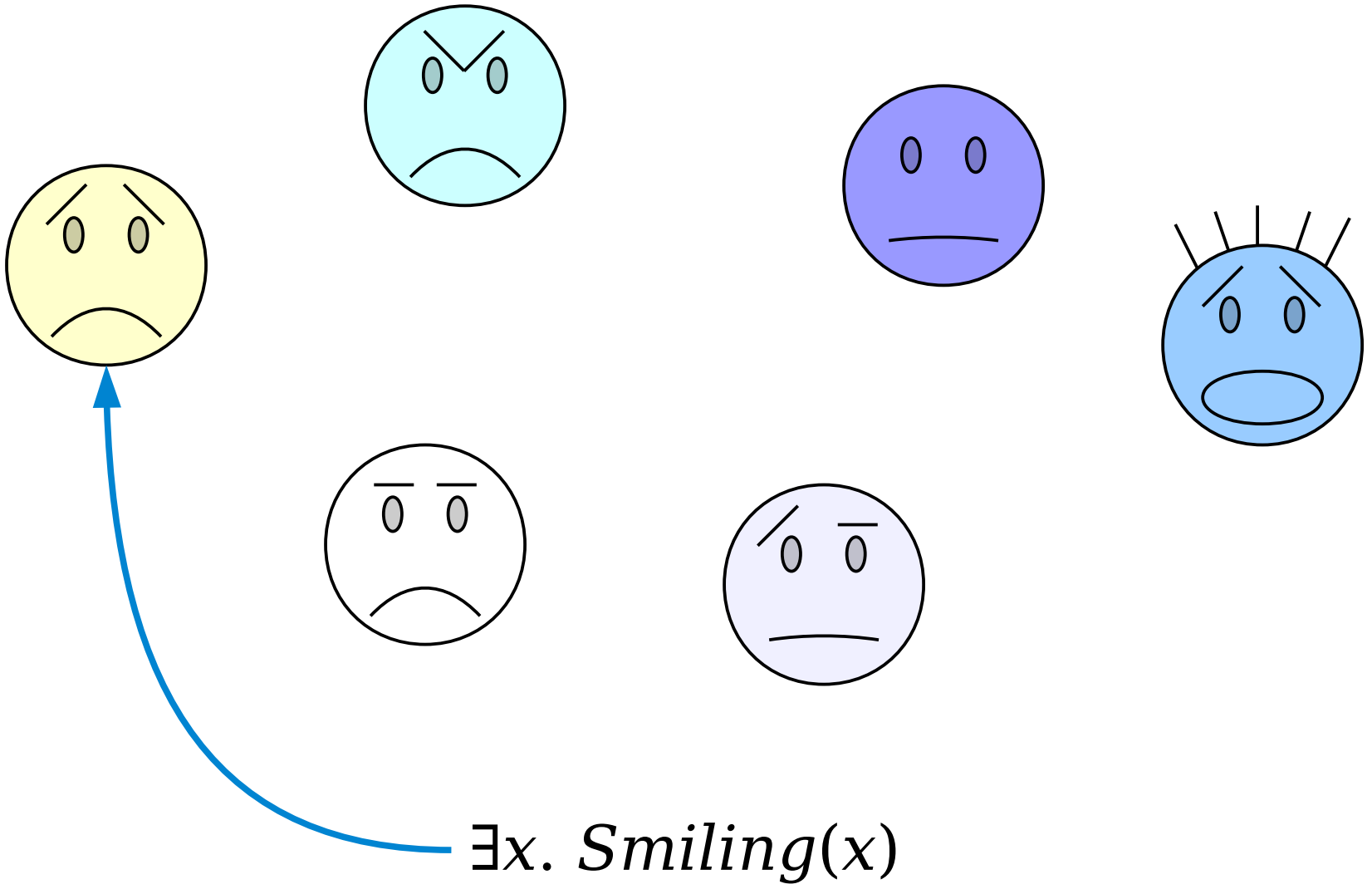
Since $Smiling(x)$ is true for some choice of x, this statement evaluates to true.

The Existential Quantifier

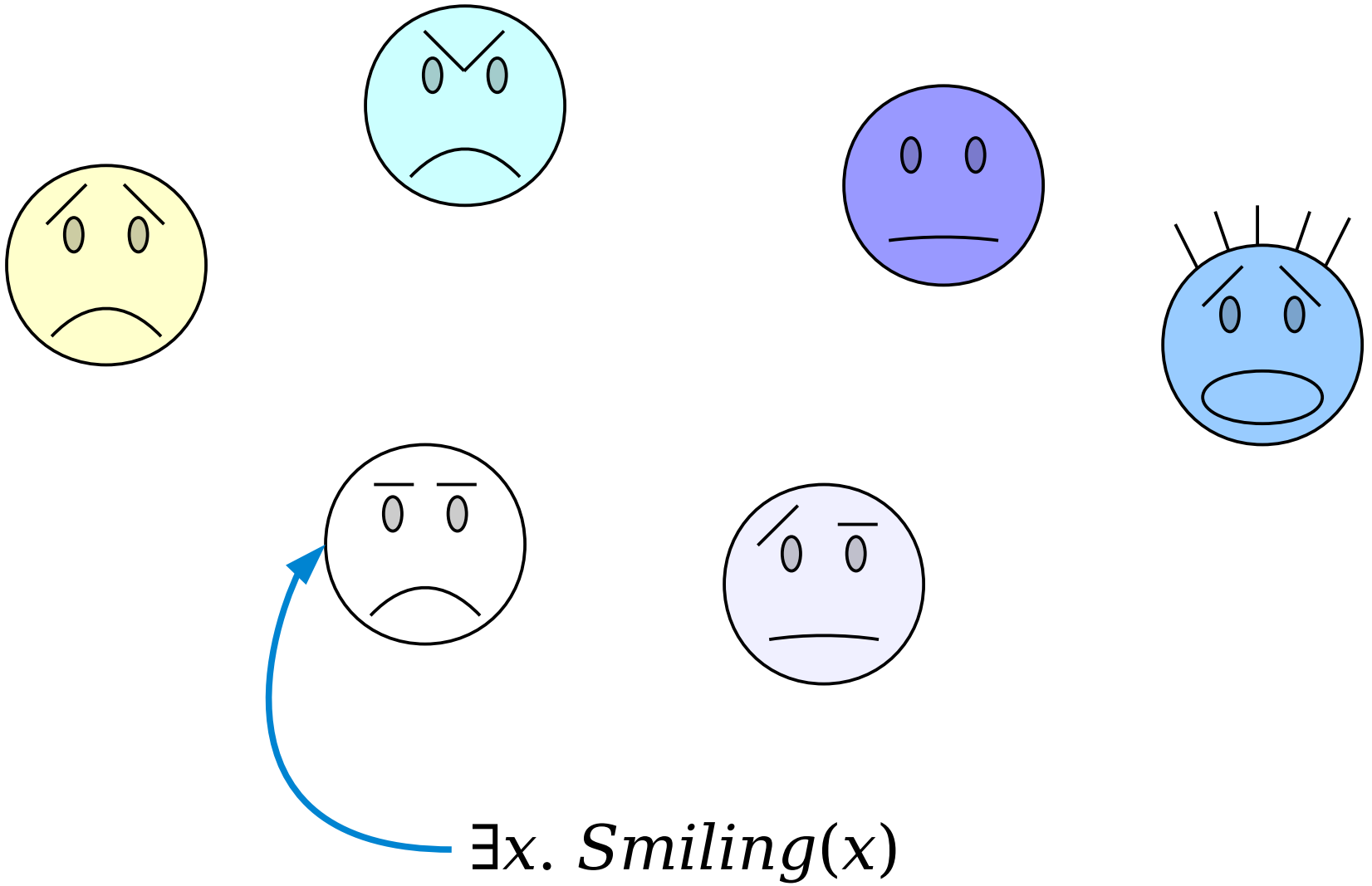


$\exists x.\ \mathit{Smiling}(x)$

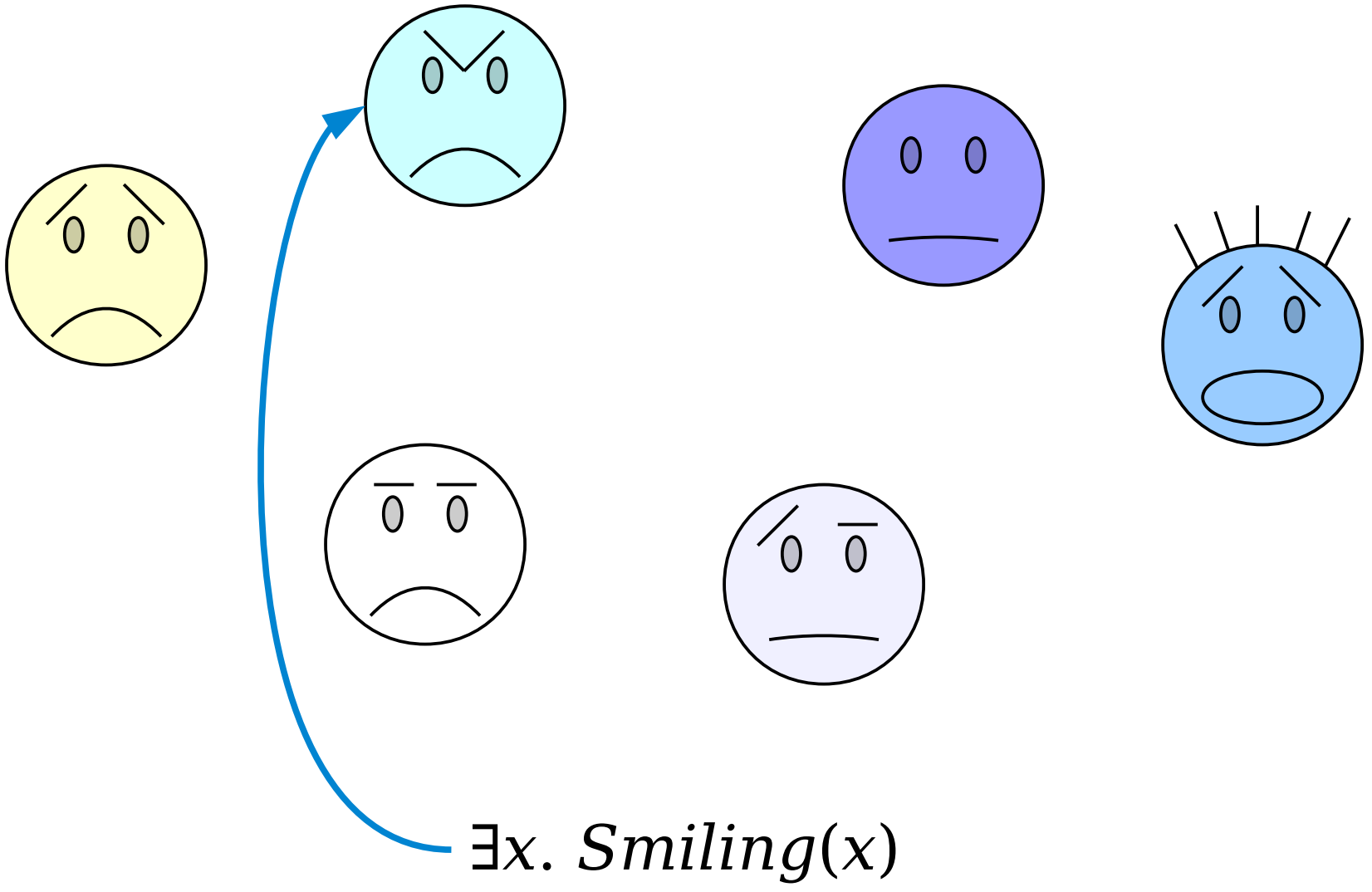
The Existential Quantifier



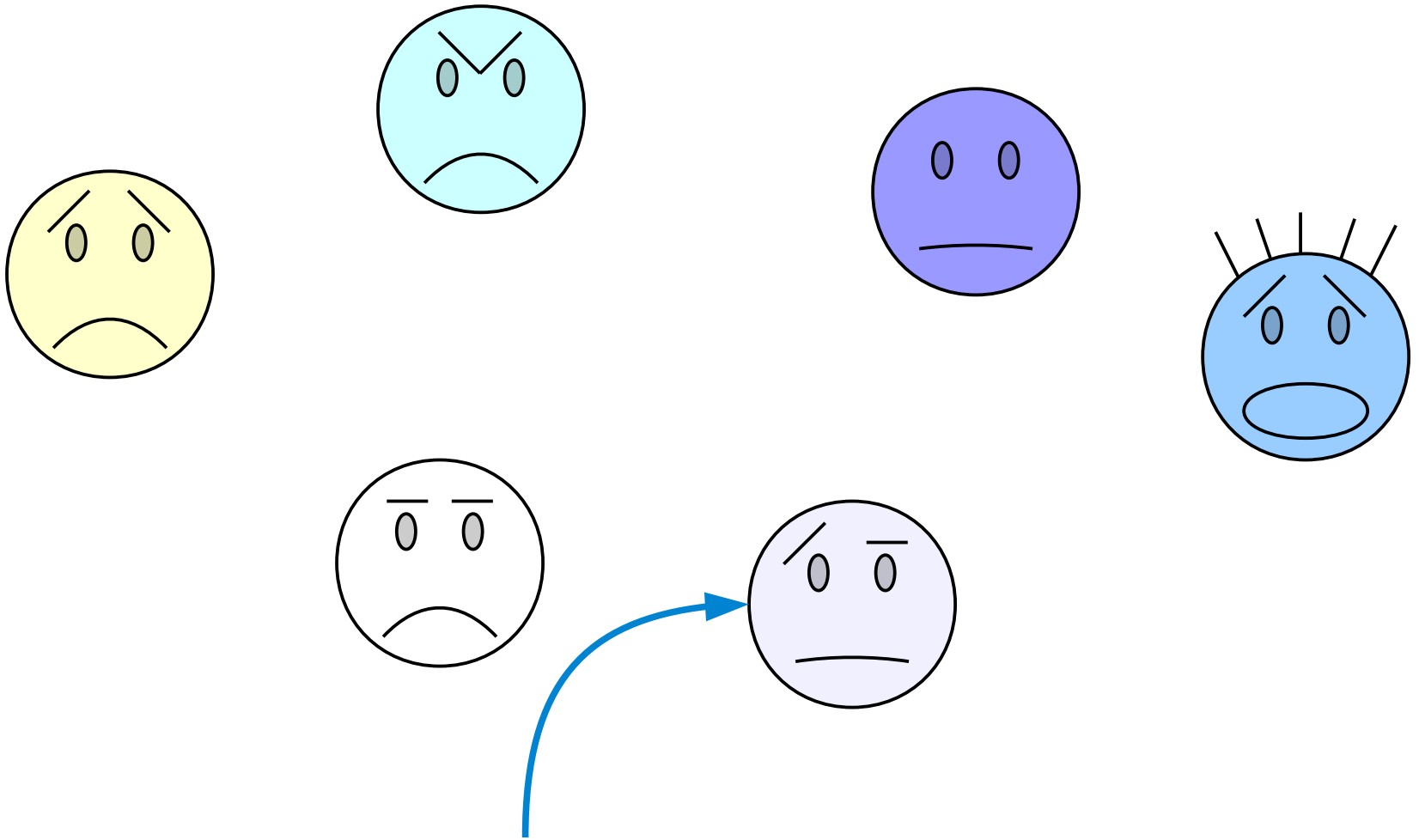
The Existential Quantifier



The Existential Quantifier

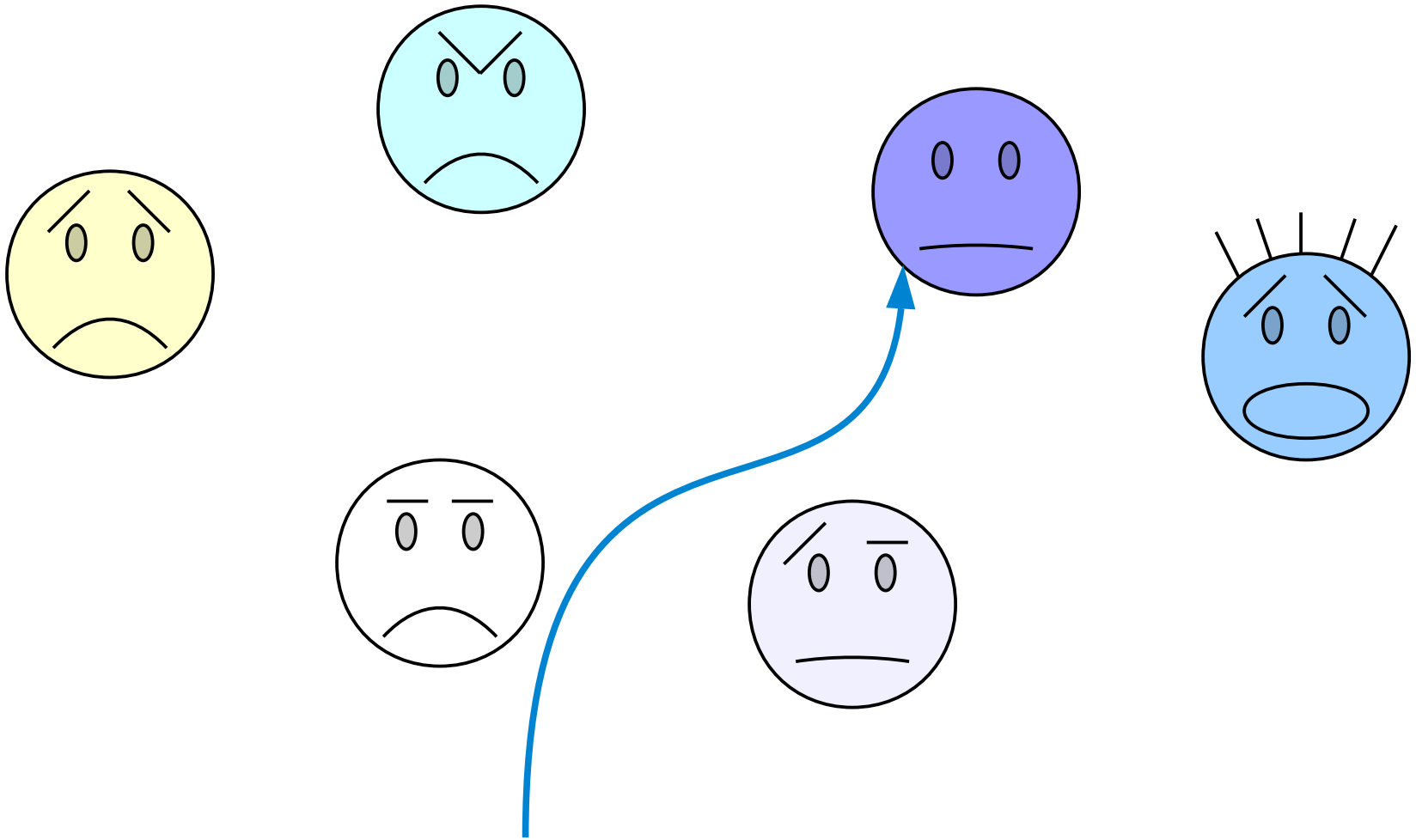


The Existential Quantifier



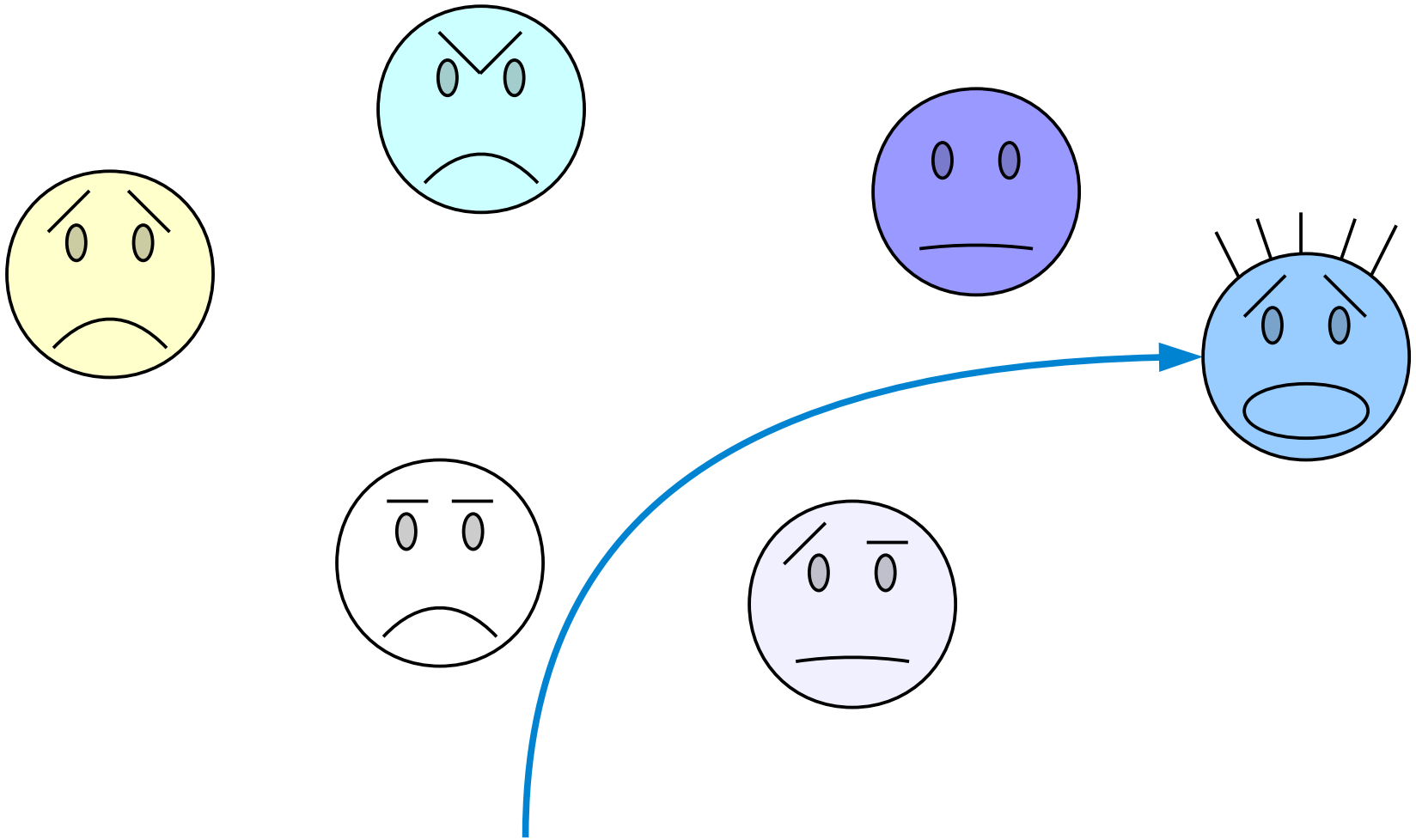
$\exists x.\ \mathit{Smiling}(x)$

The Existential Quantifier



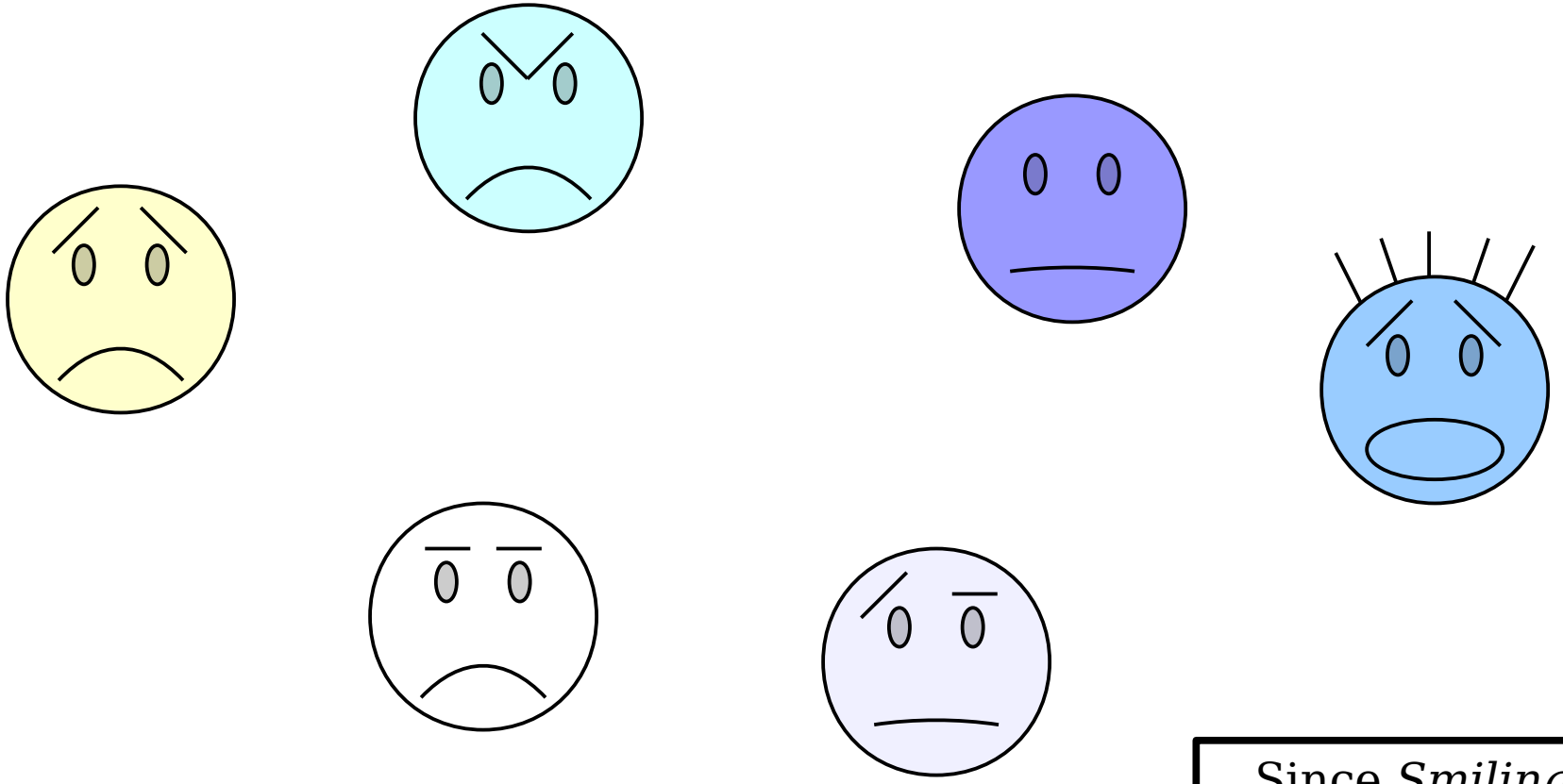
$\exists x.\ \mathit{Smiling}(x)$

The Existential Quantifier



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

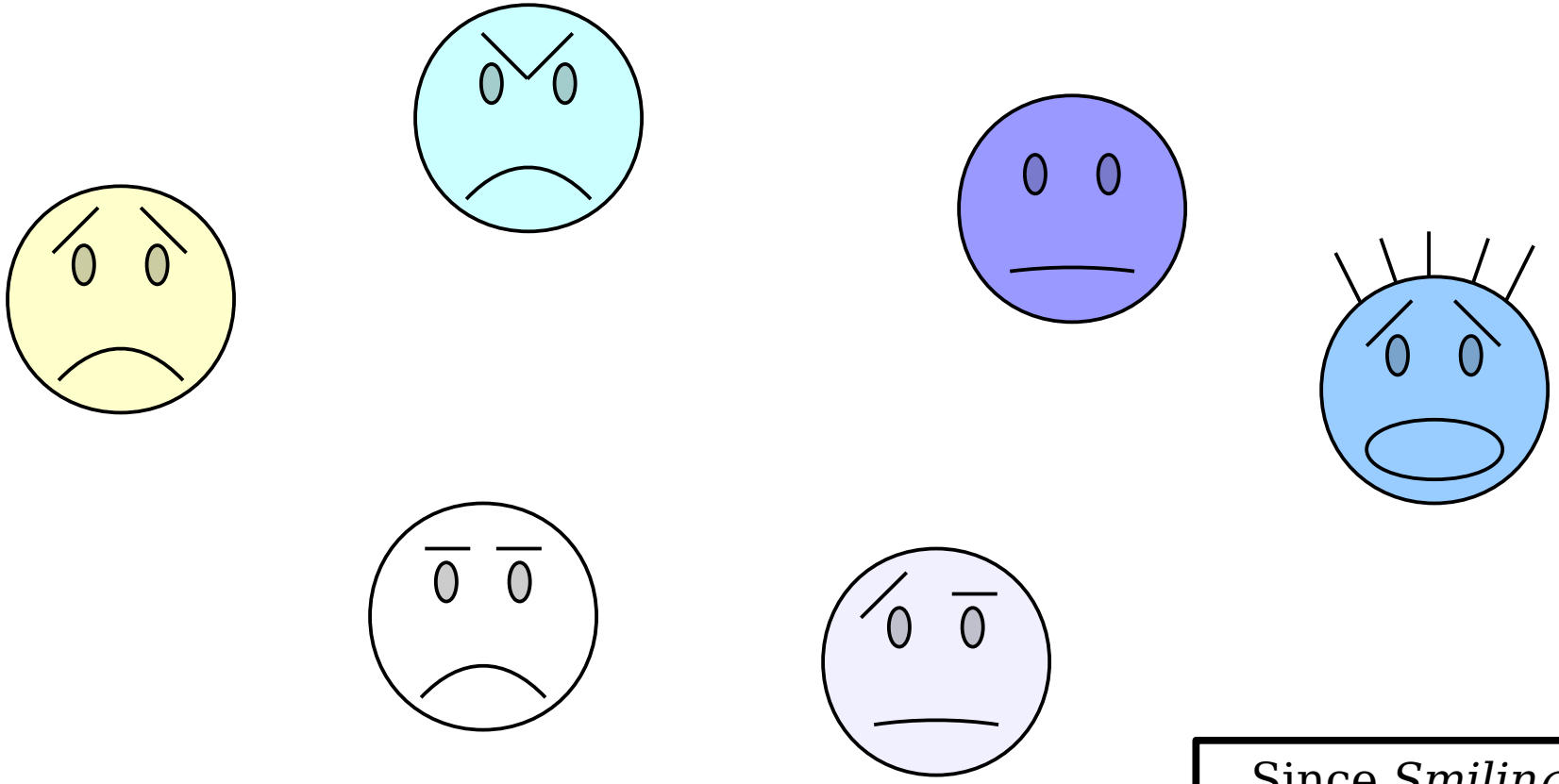
The Existential Quantifier



$\exists x.\ Smiling(x)$

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

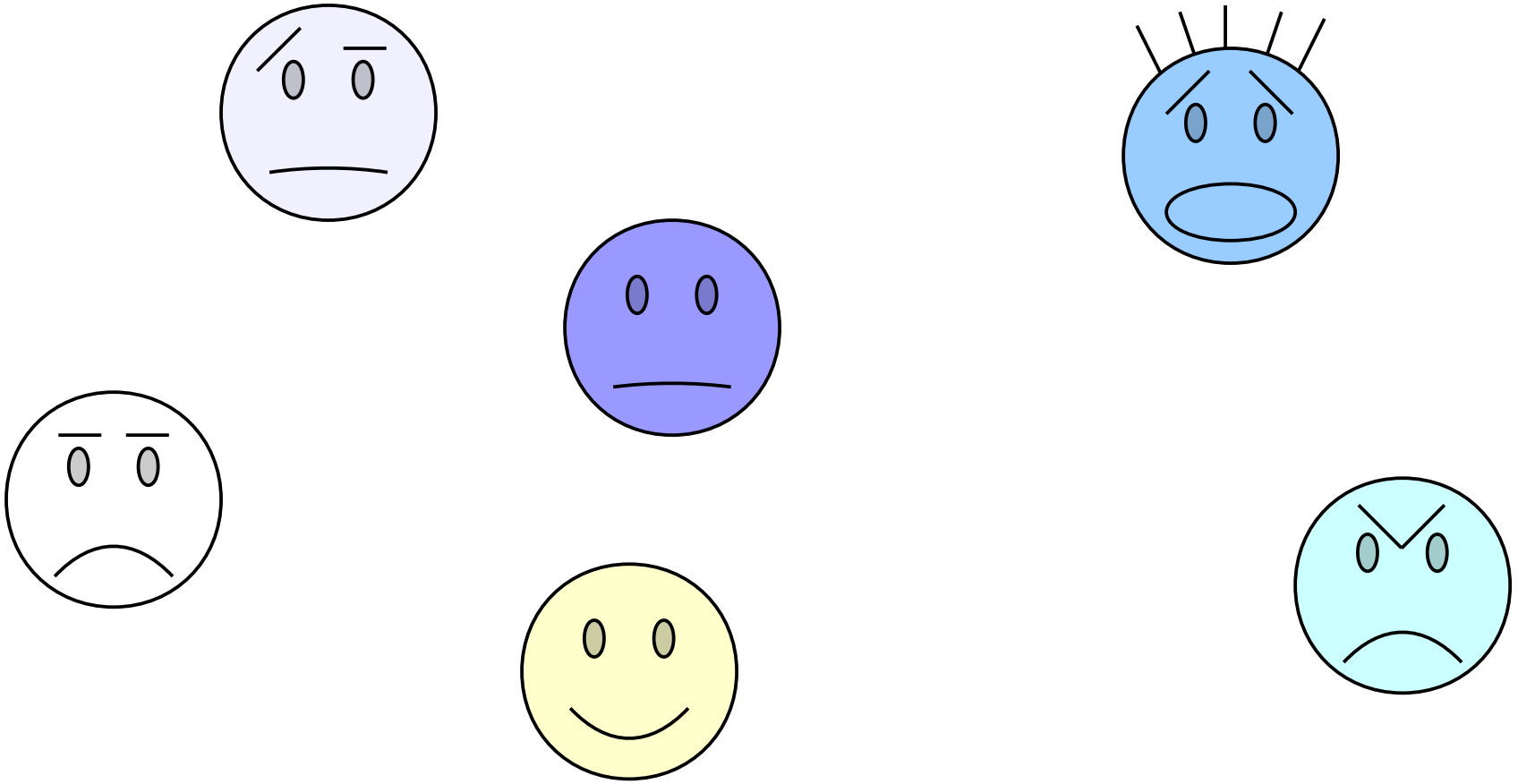
The Existential Quantifier



$\exists x.\ Smiling(x)$

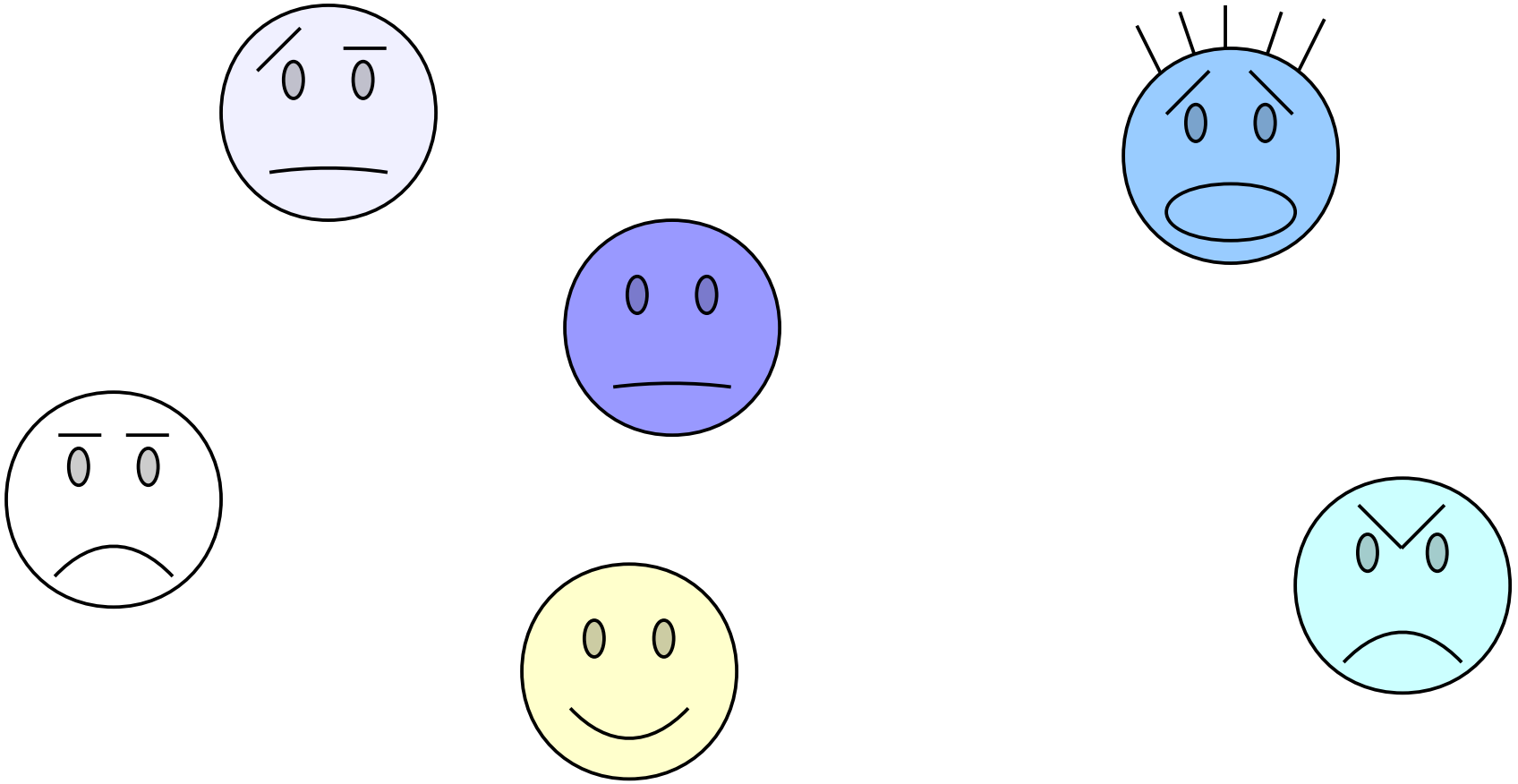
Since $Smiling(x)$ is not true for any choice of x, this statement evaluates to false.

The Existential Quantifier



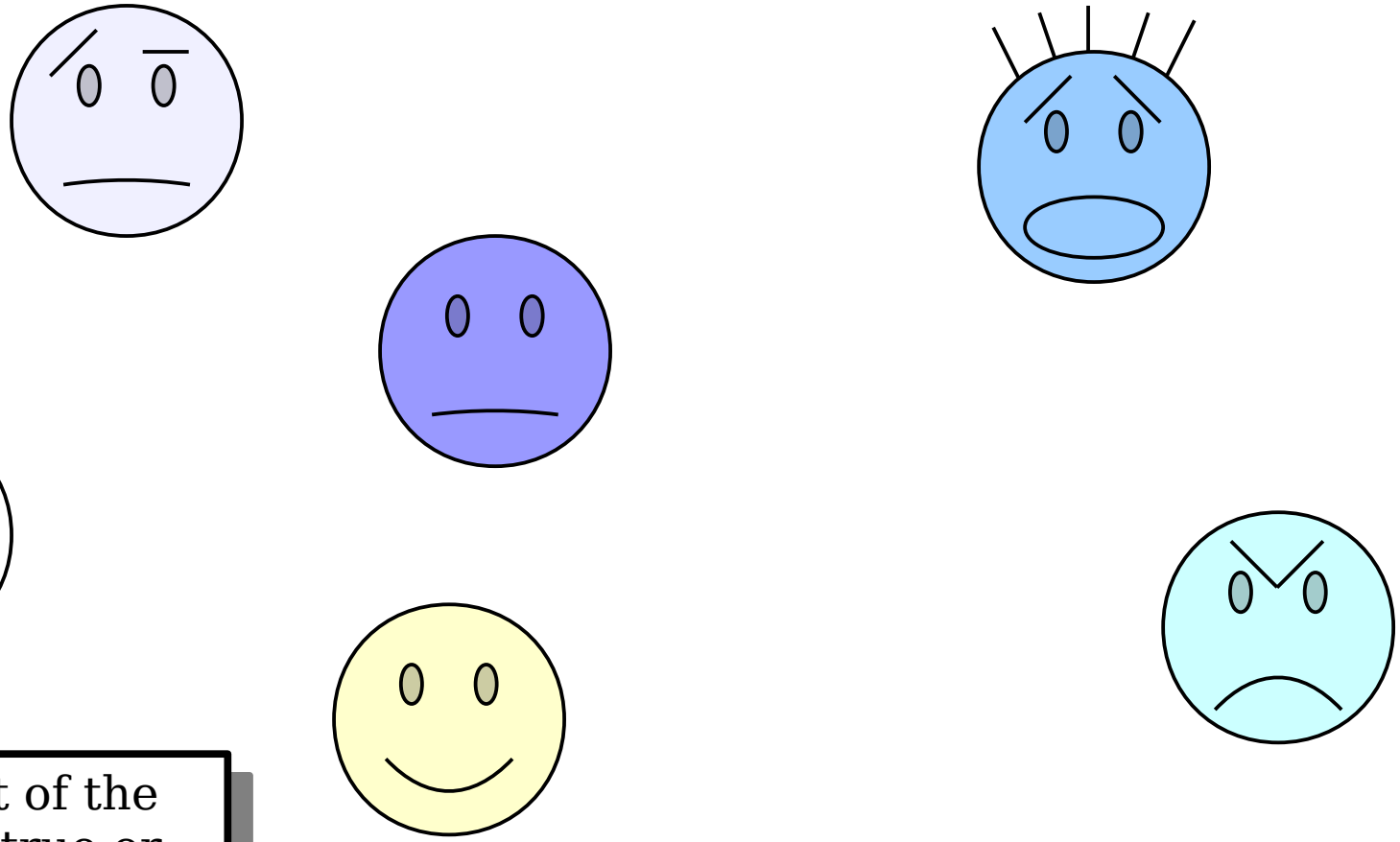
$(\exists x.\ \mathit{Smiling}(x)) \rightarrow (\exists y.\ \mathit{WearingHat}(y))$

The Existential Quantifier



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

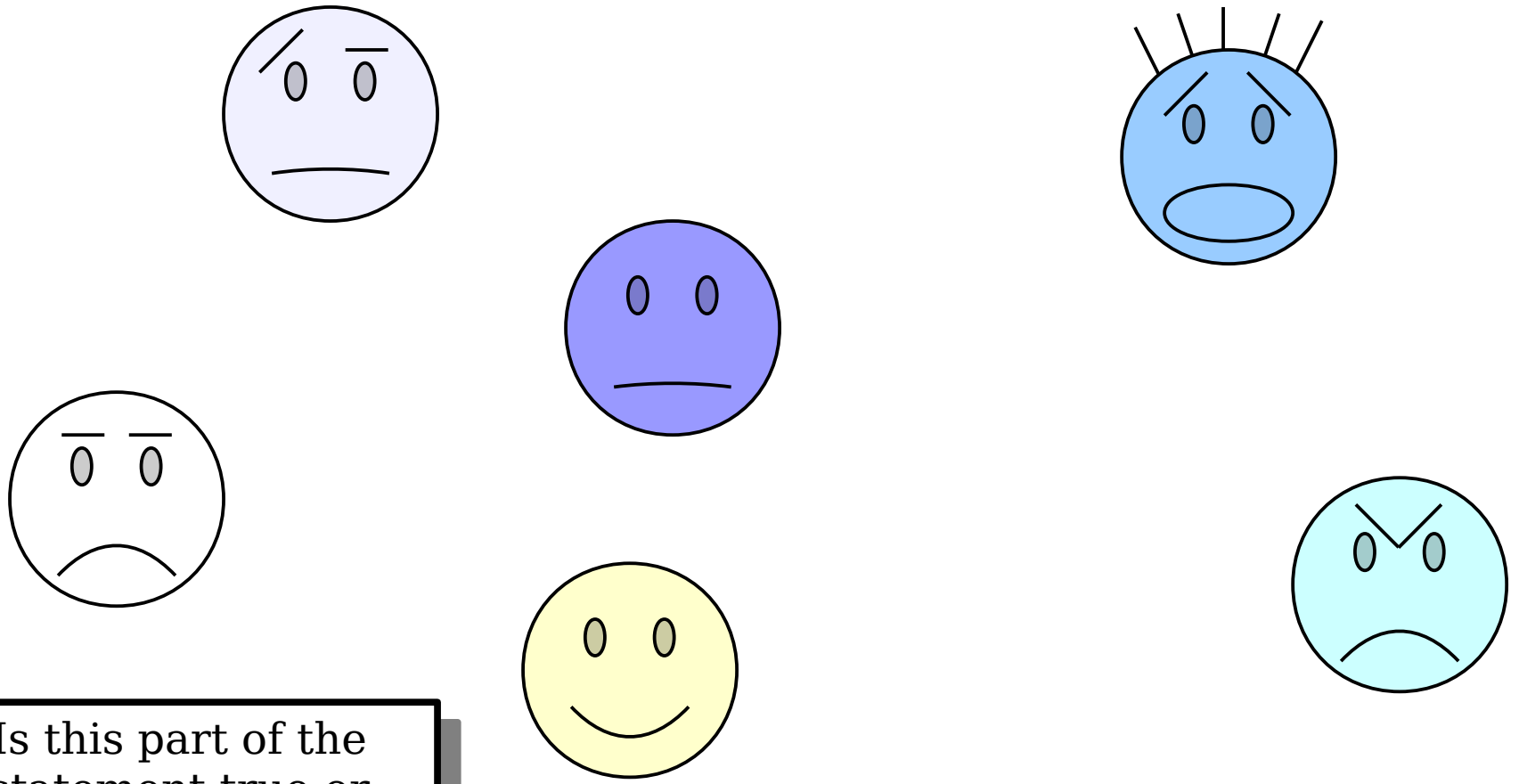
The Existential Quantifier



Is this part of the statement true or false?

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

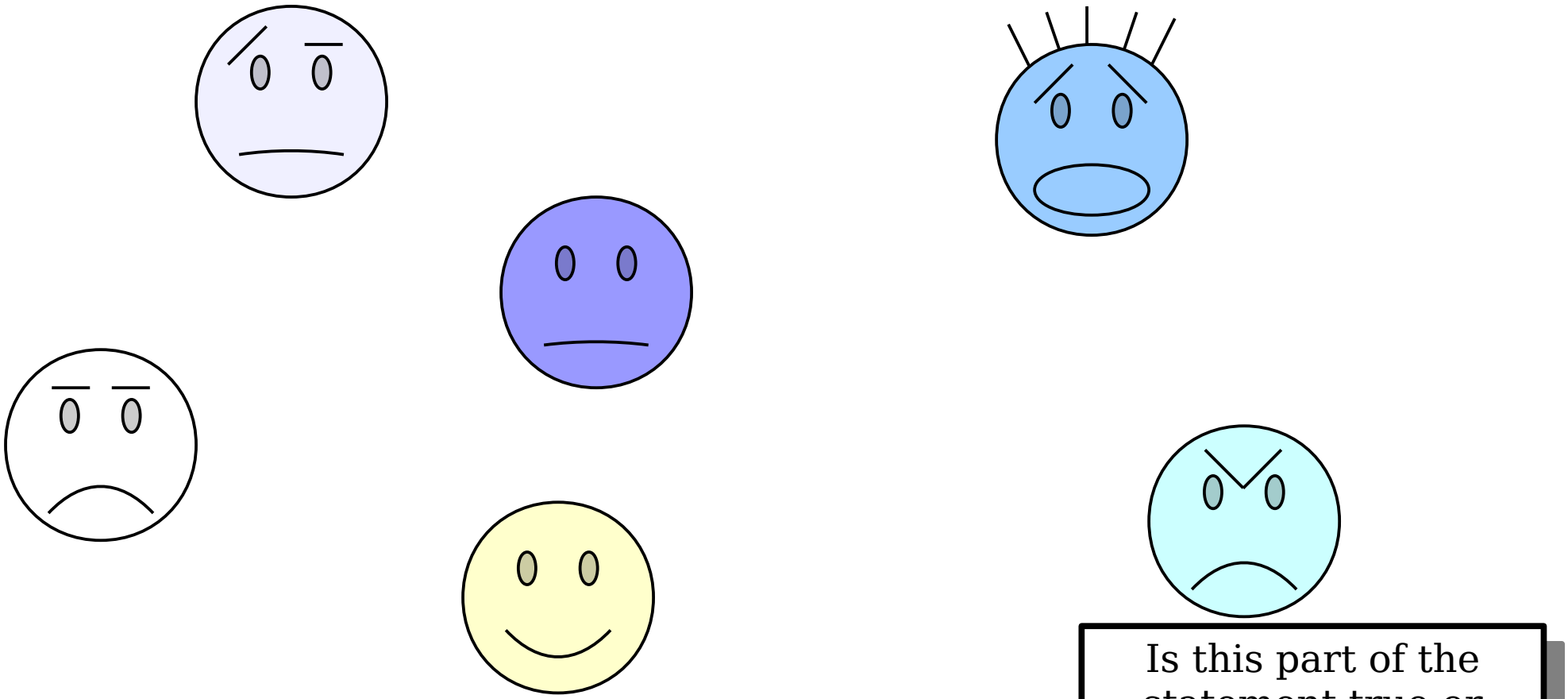
The Existential Quantifier



Is this part of the statement true or false?

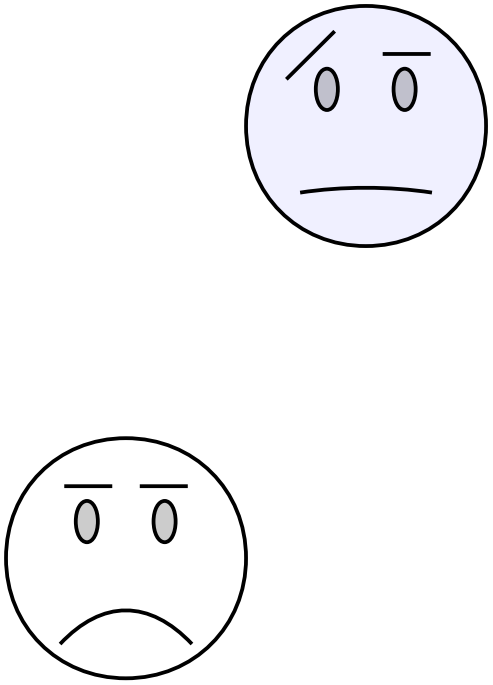
$(\exists x.\ \mathit{Smiling}(x)) \rightarrow (\exists y.\ \mathit{WearingHat}(y))$

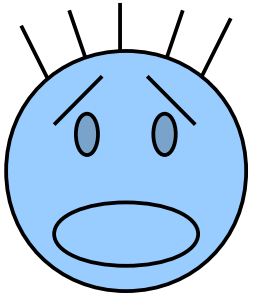
The Existential Quantifier



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

The Existential Quantifier

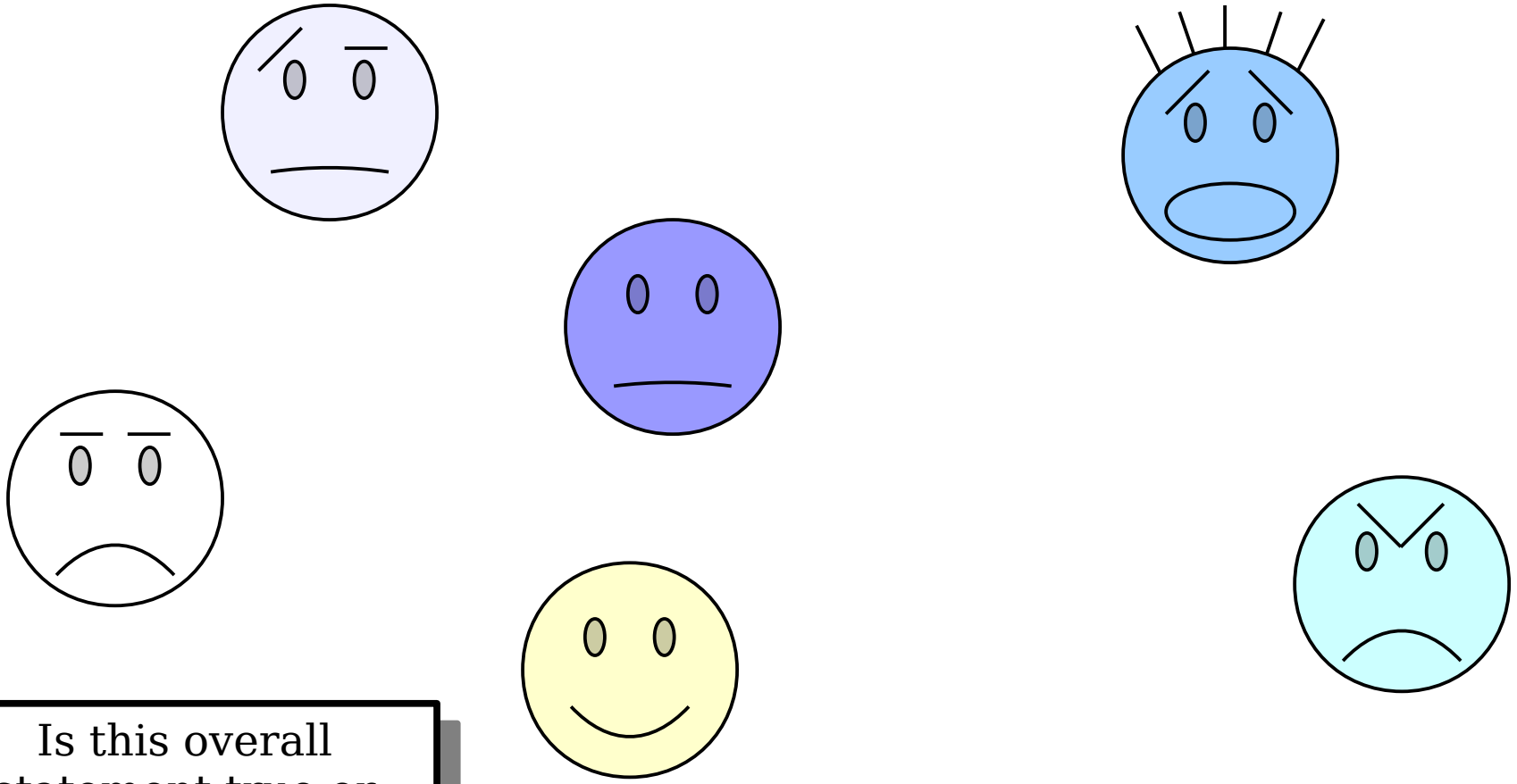




Is this part of the statement true or false?

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

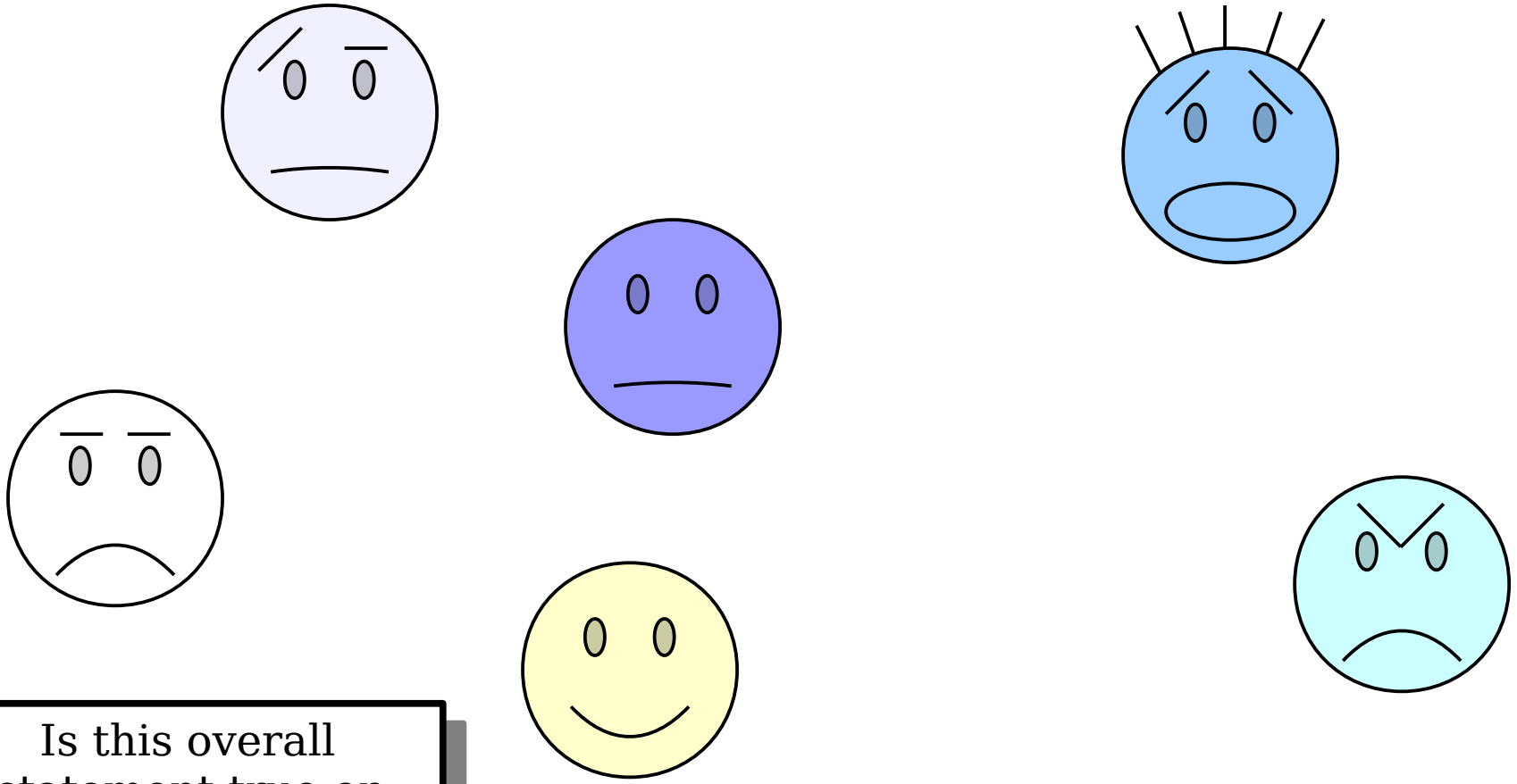
The Existential Quantifier



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 it's not possible to choose an object!

$\exists x.\ Smiling(x)$

Next Time

First-Order Logic

- Reasoning about groups of objects.

First-Order Translations

- Expressing yourself in symbolic math!