

# 知識工学

第四回 論理表現と問題解決

# コメント



<http://papapac.com/post.php?room=知識情報工学citns>



コメントを投稿したい人

教えてもらった部屋名を入れてね

知識情報工学citns

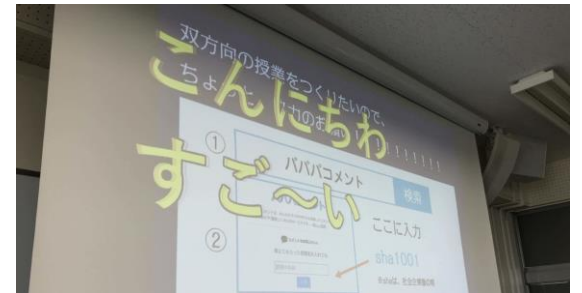
入室



知識情報工学citnsの部屋

コメント送信

コメントは部外者にも見られる可能性があります。個人情報などは送信しないでください。



面白い・興味がある  
(偶数)

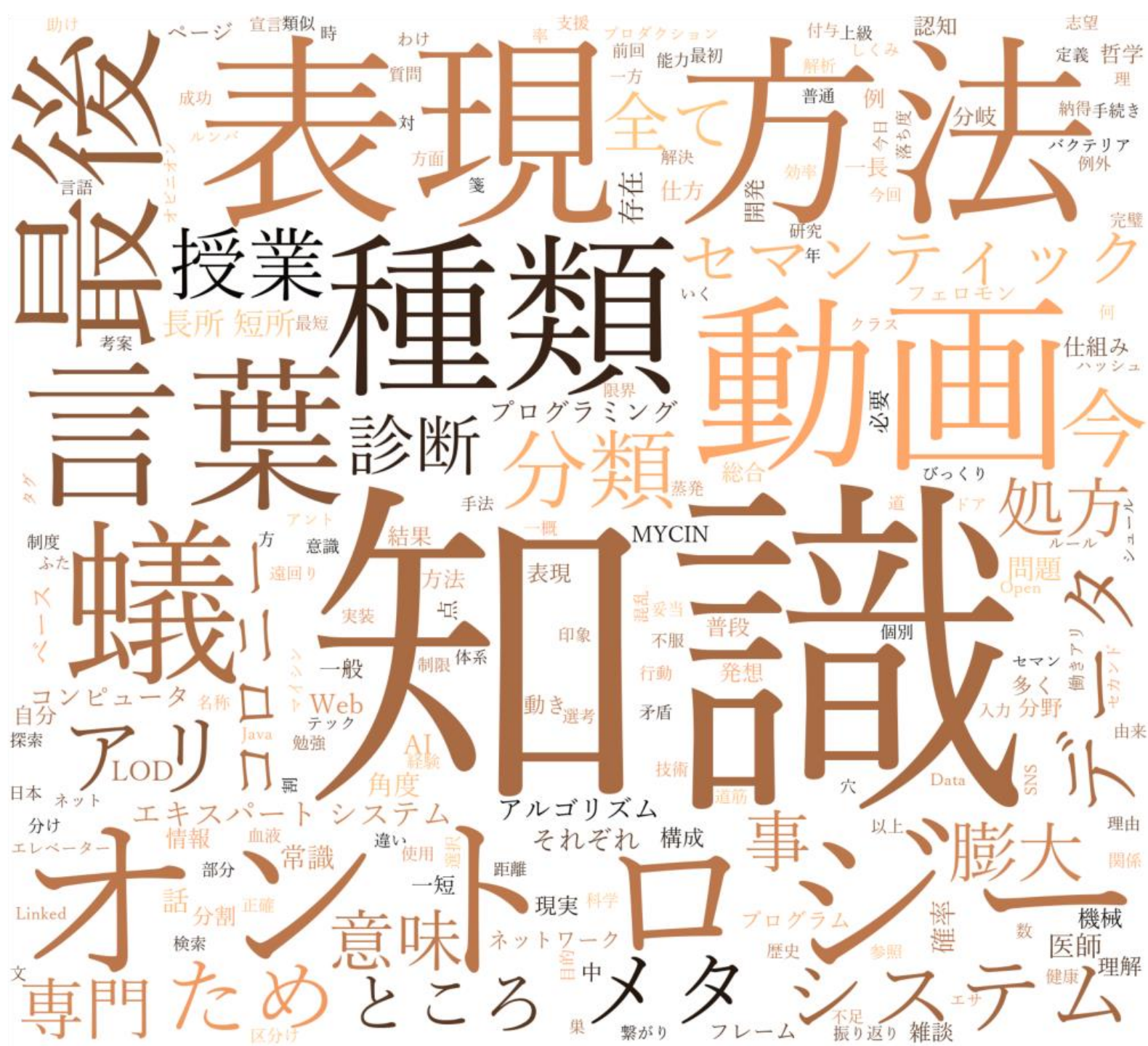


もっと知りたい  
(偶数)





面白い・興味がある  
(奇数)



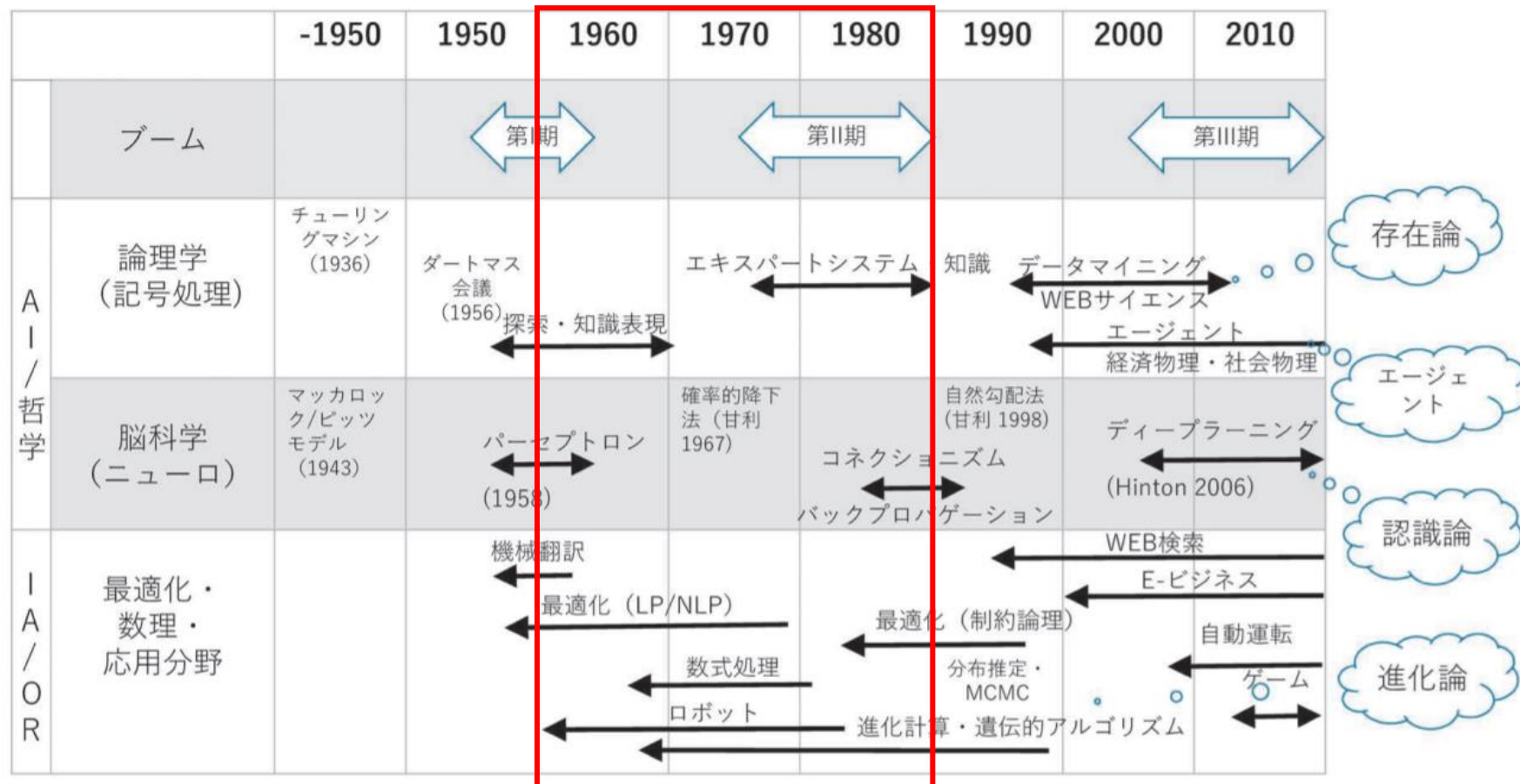
もっと知りたい  
(奇数)





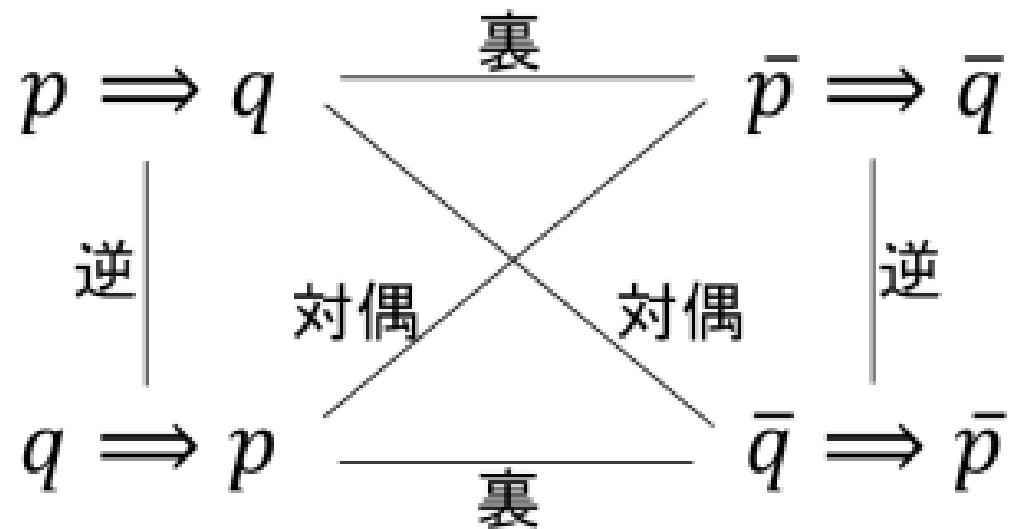
# 今日の内容

- 論理学を用いた推論
- ロボット等のプランニングにも使用される



# 論理学があつかう事柄

- 数学は数についての真理を探求するが，論理学は真理そのものを探求する．
- 言い換えれば，「～～が真であるとき，～～は真である」とは，どのようなときに成り立つのかを調べる．



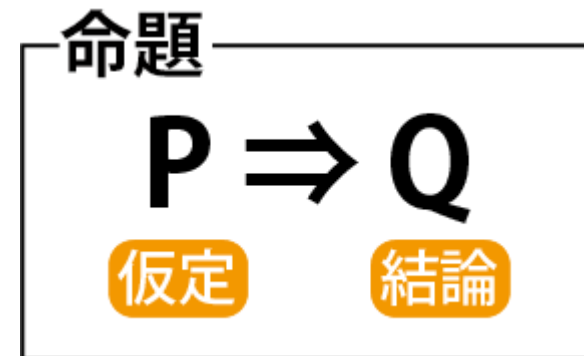


# 命題とは

- YesかNoかがはっきりと決まる文章のこと
- pやqといった記号で表す.

$p \rightarrow q$

- pという条件を満たすとき, qという条件を満たす.
- 平たく言えば「pならばq」



# 命題の例

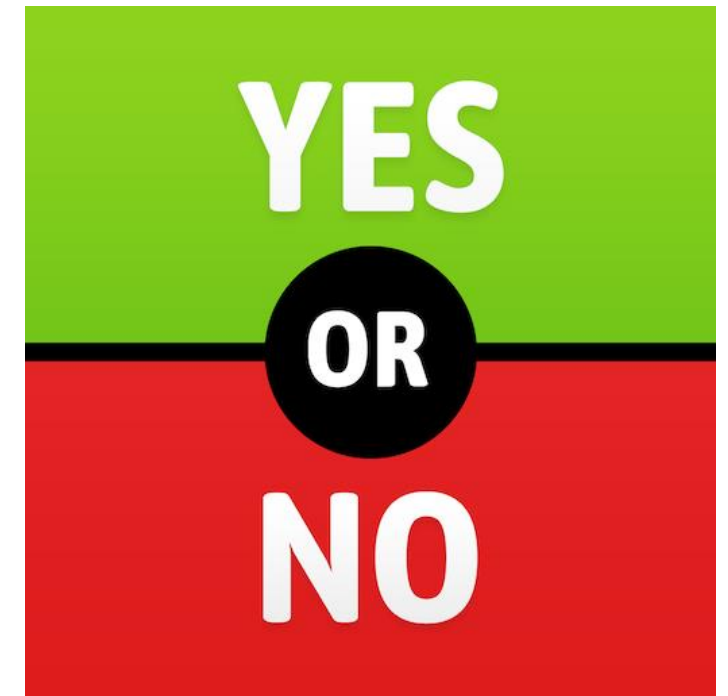
この科目名は知識工学である：真

この7号館は20階建てである：偽

成績評価で60点以上ならば単位がもらえる：真

## Quiz

1.  $x=0$  ならば,  $xy=0$ である.
2.  $x=0$  ならば,  $x+1 = 2$ である.
3.  $x^2=4$ ならば,  $x=2$ である.
4.  $x=2$  ならば,  $x^2=4$ である.



# 命題の例

この科目名は知識工学である：真

この7号館は20階建てである：偽

成績評価で60点以上ならば単位がもらえる：真

## Quiz

1.  $x=0$  ならば,  $xy=0$ である. 真
2.  $x=0$  ならば,  $x+1 = 2$ である.
3.  $x^2=4$ ならば,  $x=2$ である.
4.  $x=2$  ならば,  $x^2=4$ である.





# 命題の例

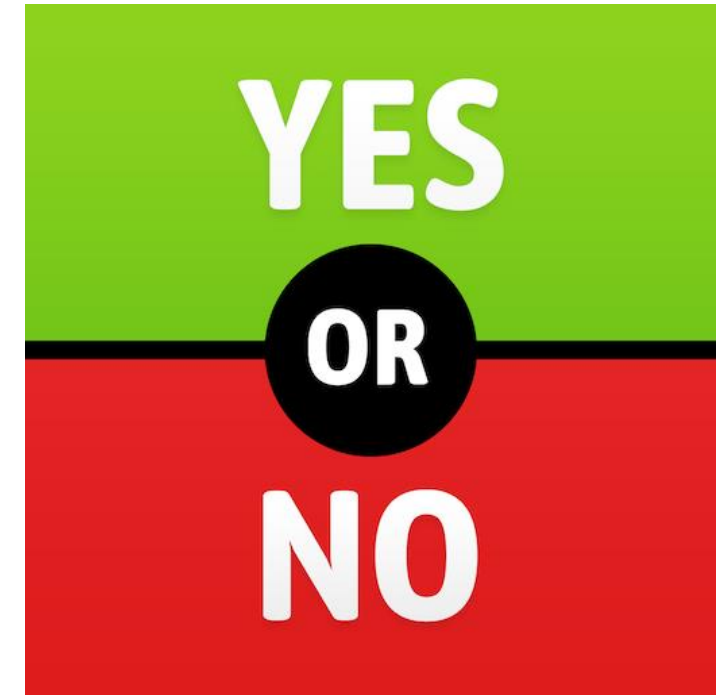
この科目名は知識工学である：真

この7号館は20階建てである：偽

成績評価で60点以上ならば単位がもらえる：真

## Quiz

1.  $x=0$  ならば,  $xy=0$ である. 真
2.  $x=0$  ならば,  $x+1 = 2$ である. 偽
3.  $x^2=4$ ならば,  $x=2$ である.
4.  $x=2$  ならば,  $x^2=4$ である.



# 命題の例

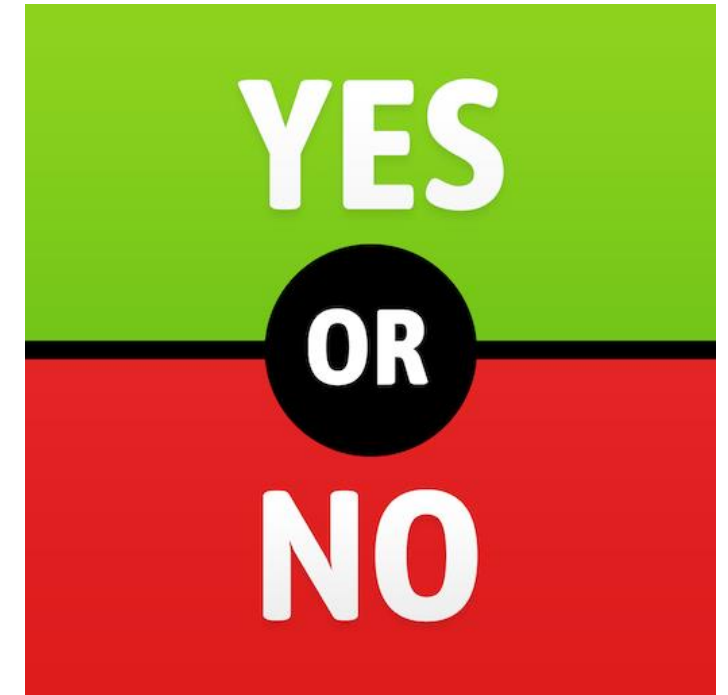
この科目名は知識工学である：真

この7号館は20階建てである：偽

成績評価で60点以上ならば単位がもらえる：真

## Quiz

1.  $x=0$  ならば,  $xy=0$ である. 真
2.  $x=0$  ならば,  $x+1 = 2$ である. 偽
3.  $x^2=4$ ならば,  $x=2$ である. 偽
4.  $x=2$  ならば,  $x^2=4$ である.



# 命題の例

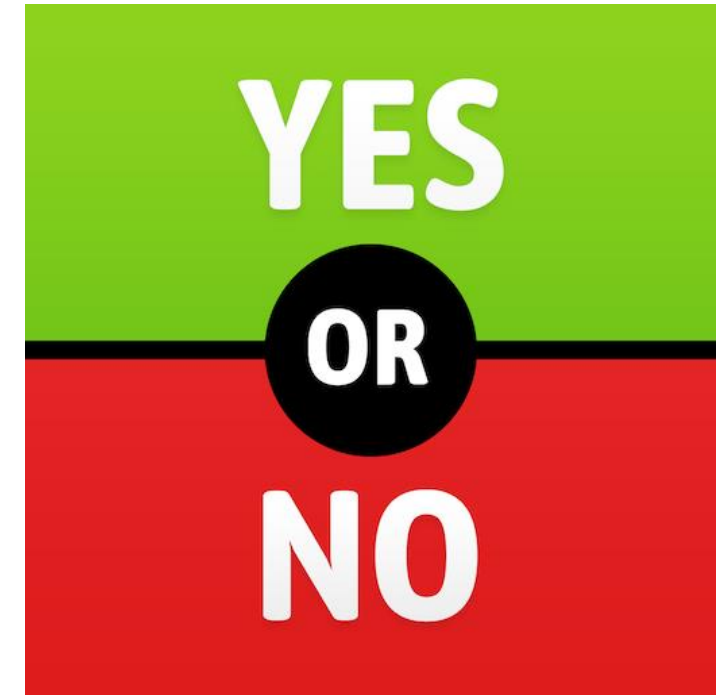
この科目名は知識工学である：真

この7号館は20階建てである：偽

成績評価で60点以上ならば単位がもらえる：真

## Quiz

1.  $x=0$  ならば,  $xy=0$ である. 真
2.  $x=0$  ならば,  $x+1 = 2$ である. 偽
3.  $x^2=4$ ならば,  $x=2$ である. 偽
4.  $x=2$  ならば,  $x^2=4$ である. 真





# 曖昧な点

真：  $x=2$  であるならば，  $x^2=4$  である．

$x = -2$  のときも，  $x^2=4$  である．

この場合はどうなるのか？



PとQは一対一ではない

P :  $x=2$

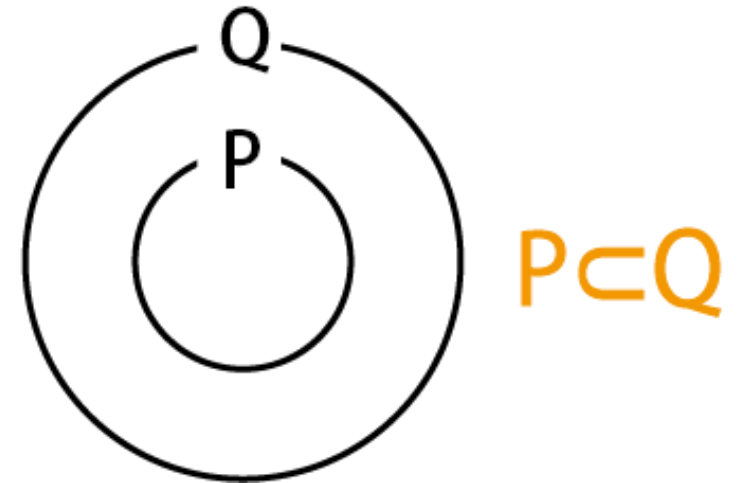
Q :  $x^2=4$

このとき、 $P \Rightarrow Q$ は真である.

ただしQは、 $x=-2$ のときも成立する.

つまり、PはQを満たす条件の1つでしかなく、他のQに至る可能性を排除したものではない.

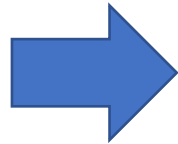
• 命題  $P \Rightarrow Q$  が真の時



# PとQのつながり

- 実は、PとQで意味のつながりは必要ない
  - 「円周率が3以上ならば、馬は動物である」は真である.
  - 言葉と記号の意味が合わないので、改良しようという動きもあるらしい

$\pi > 3$



: 真

| <i>A</i> | <i>B</i> | <i>A</i> → <i>B</i> |
|----------|----------|---------------------|
| <i>t</i> | <i>t</i> | <i>t</i>            |
| <i>t</i> | <i>f</i> | <i>f</i>            |
| <i>f</i> | <i>t</i> | <i>t</i>            |
| <i>f</i> | <i>f</i> | <i>t</i>            |



# PとQの繋がりに関する動画



# SPIのような問題

次の①～③はいずれも事実であるとする.

- ①お風呂が好きな人は、きれい好きである.
- ②歯みがきが好きな人は、お風呂が好きである.
- ③お酒が好きな人は、タバコが好きであり、しかも歯みがきが好きである.

このとき、次のア～オの中で確実にいえることをすべて選べ.

- ア. 歯みがきが好きな人は、タバコが好きである.
- イ. お酒が好きな人は、歯みがきが好きである.
- ウ. タバコが好きな人は、きれい好きである.
- エ. タバコが好きでない人は、お風呂が好きでない.
- オ. きれい好きでない人は、お酒が好きでない.

# 記号で表すと

- ①お風呂  $\rightarrow$  きれい
- ②歯みがき  $\rightarrow$  お風呂
- ③お酒  $\rightarrow$  タバコ  $\wedge$  歯みがき

ア. 歯みがき  $\rightarrow$  タバコ

イ. お酒  $\rightarrow$  歯みがき

ウ. タバコ  $\rightarrow$  きれい

エ.  $\neg$ タバコ  $\rightarrow$   $\neg$ お風呂

オ.  $\neg$ きれい  $\rightarrow$   $\neg$ お酒

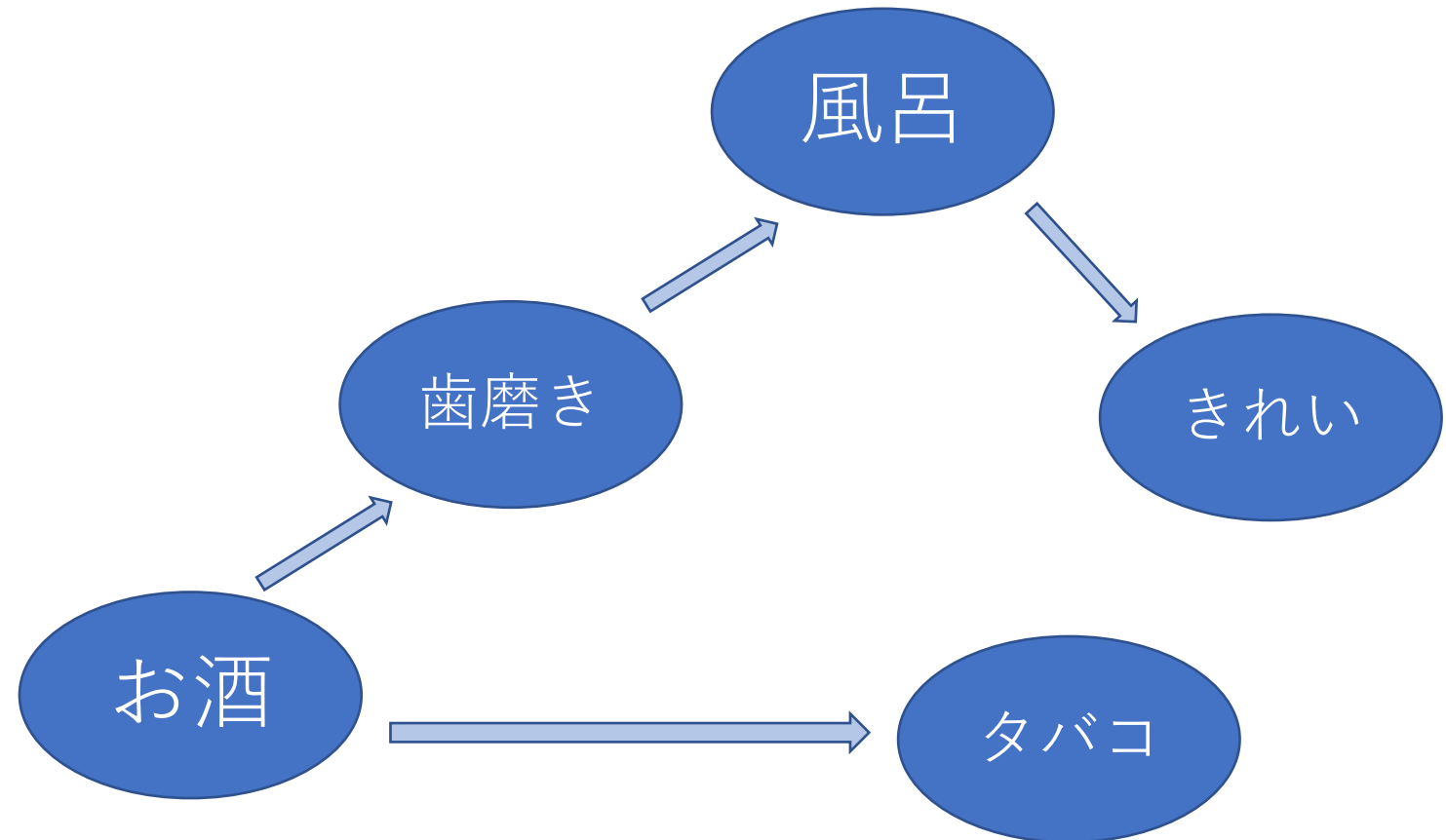
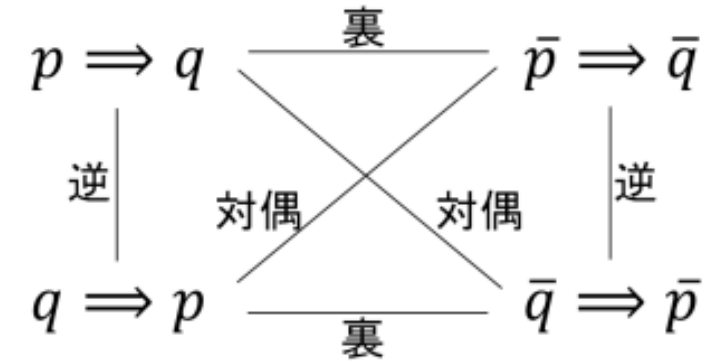
※  $\neg$ は否定を意味する

# 記号で表すと

- ①お風呂 → きれい
- ②歯みがき → お風呂
- ③お酒 → タバコ ∧ 歯みがき

- ア. 歯みがき → タバコ
- イ. お酒 → 歯みがき
- ウ. タバコ → きれい
- エ.  $\neg$ タバコ →  $\neg$ お風呂
- オ.  $\neg$ きれい →  $\neg$ お酒

命題とその対偶の真偽  
は必ず一致する



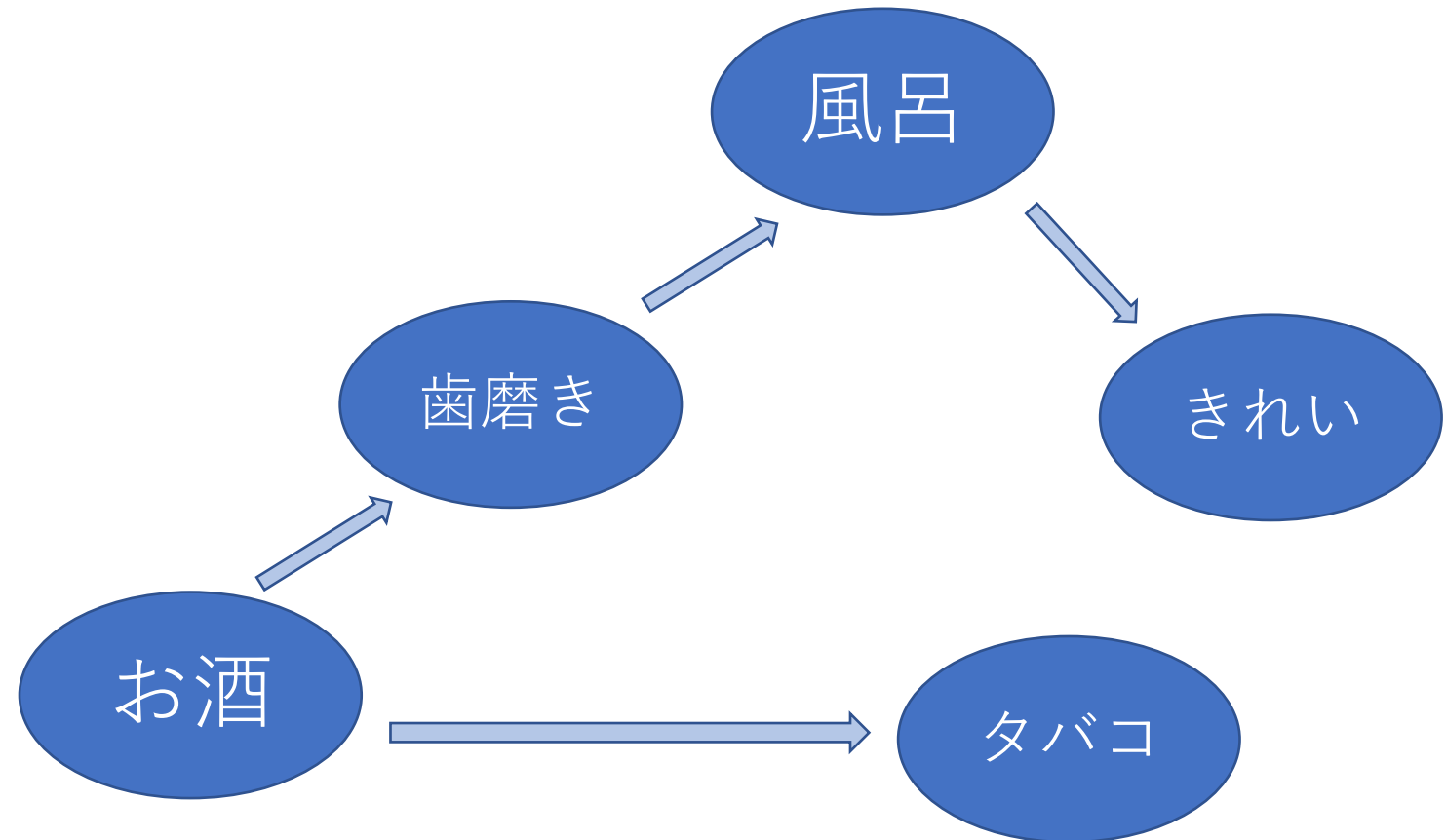
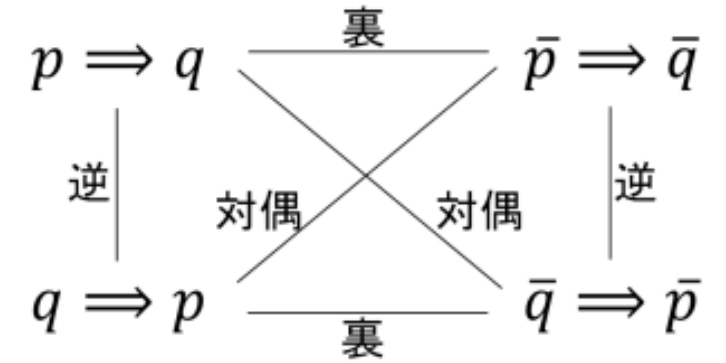


# 記号で表すと

- ①お風呂 → きれい
- ②歯みがき → お風呂
- ③お酒 → タバコ ∧ 歯みがき

- ア. 歯みがき → タバコ
- イ. お酒 → 歯みがき
- ウ. タバコ → きれい
- エ.  $\neg$ タバコ →  $\neg$ お風呂
- オ.  $\neg$ きれい →  $\neg$ お酒

命題とその対偶の真偽  
は必ず一致する



# 一階述語論理

- **述語論理**：命題を分解し，主語と述語を記号（変数）で表現する.
- 具体的には，命題「 $x$ は $P$ である」を $P(x)$ として表す.  
例）「多胡は器用である」を，器用という述語記号で示すと，器用（多胡）
- 一階があるからには二階もあるが，この辺は後で解説します.
  - どの程度までメタ的に考えるかによって階層が増えていく

# 述語論理での記号種類

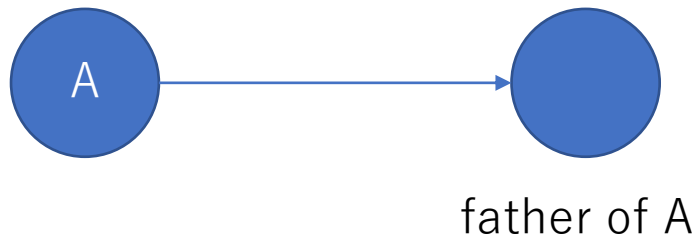
変数 :  $x, y, z, \dots$  (任意の個体を示す)

定数 :  $A, B, C, \dots$  (特定の個体を示す. ドナルド・トランプなど)

関数記号 :  $f, g, h, \dots$  (個体間の関係を示す.  $\text{father}(A)$  ならば  $A$  の父である. )

述語記号 :  $P, Q, R, \dots$  (個体の性質を示す.  $\text{Happy}(A)$  ならば  $A$  は幸せである)

演算子 :  $\neg$  (否定),  $\wedge$  (and),  $\vee$  (or),  $\rightarrow$  (含意),  $\forall$  (**すべて**),  $\exists$  (**ある**)



関数は写像として考え、任意の個体を与えられた場合にある一つの個体を指すことを意味する.

# ∀を用いた例

- ∀は「全て」を意味し，任意の変数について成り立つことを表す.

$\forall x, \sin^2 x + \cos^2 x = 1$  (すべての $x$ について，式が成り立つ)

上記の式は常に成り立つため，この命題は真である.

$\forall$  NS教員，優しい (NS教員)

優しい( $x$ )は， $x$ が優しいことを示すとき，この命題は…

# ∃を用いた例

- ∃は「ある，任意の」を意味し，少なくとも1つ存在することを示す.

$\exists x \in \mathbb{R}, x^2=9$  （ある数 $x$ は実数に属しており，2乗すると9）

満たす $x$ が1つでも存在すれば，この命題は真となる.

$\exists$ 千葉工生，Youtuber（千葉工生）

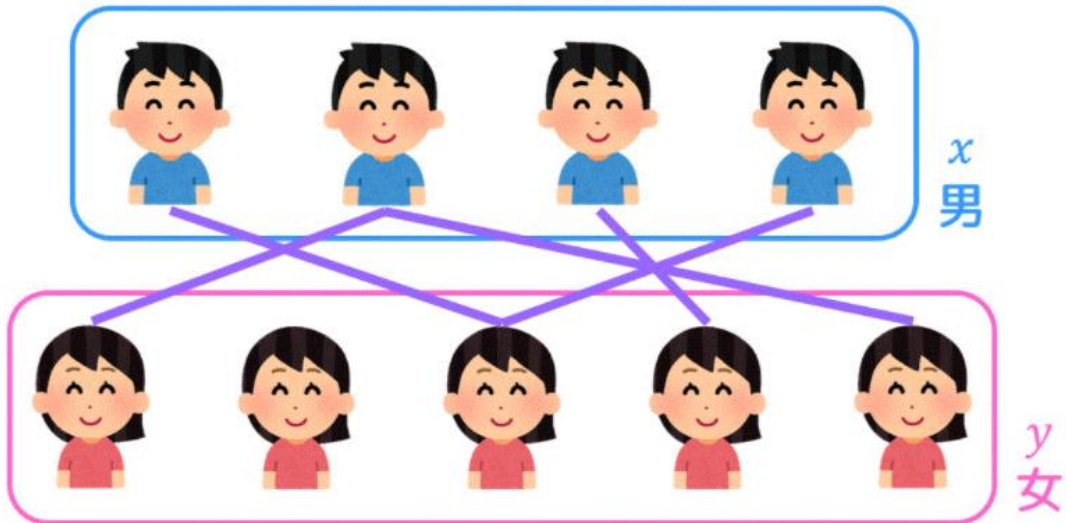
Youtuber( $x$ )は， $x$ がYoutuberであることを示すとき，この命題は…

# $\forall$ と $\exists$ を組み合わせた例

- 恋( $x, y$ )を,  $x$ は $y$ に恋している, とする.
- 男性たちを $x$ , 女性たちを $y$ とすると以下の例が成り立つ.

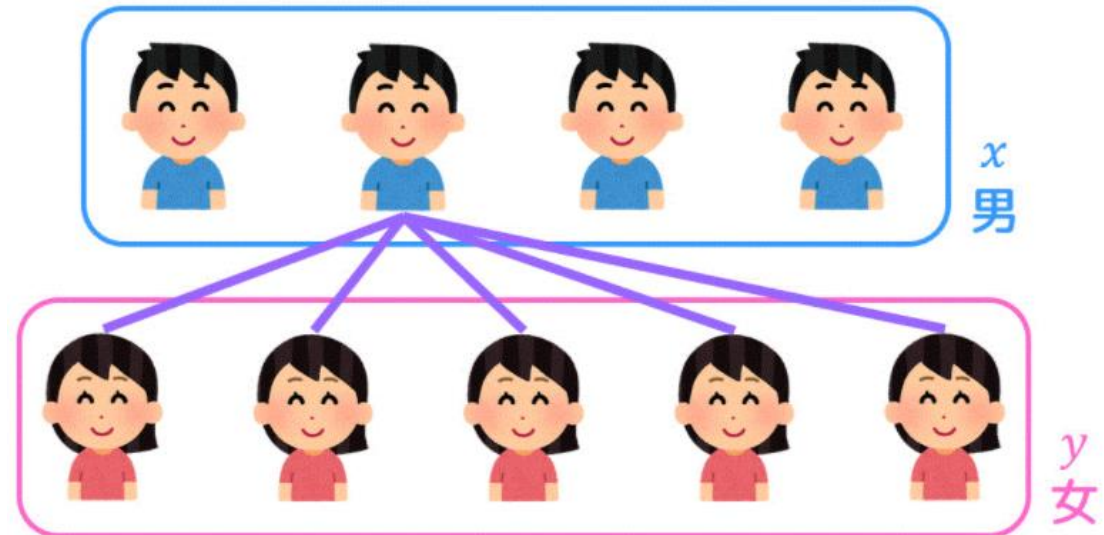
$$\forall x \exists y, \text{恋}(x, y)$$

全ての男性はある一人以上の女性に恋している.



$$\exists x \forall y, \text{恋}(x, y)$$

ある一人以上の男性は全ての女性に恋している.





# 全称記号( $\forall$ )の否定

$Fx$  は  $F(x)$  を意味し, 優秀であることを示す.  
 $x$  は学生を示す.

 $\forall x Fx$  $\neg \forall x Fx$  $\forall x \neg Fx$  $\neg \forall x \neg Fx$ 

~~すべての学生が優秀な学生である。~~  
すべての学生が優秀ではない。



優秀



優秀で  
はない

# 存在記号( $\exists$ )の否定

$Fx$  は  $F(x)$  を意味し, 優秀であることを示す.  
 $x$  は 学生を示す.

$$\exists x Fx$$

$$\neg \exists x Fx$$

$$\exists x \neg Fx$$

$$\neg \exists x \neg Fx$$

優秀な学生は存在しない。



優秀



優秀ではない

# 便利な論理式の変形

- 同値記号  $\equiv$  の除去
  - $P \equiv Q$  は,  $(P \rightarrow Q) \wedge (Q \rightarrow P)$  に変形できる.
- 含意記号  $\rightarrow$  の除去
  - 「 $A \rightarrow B$ 」は「 $\neg A \vee B$ 」と変換できる.
- ド・モルガンの法則
  - $\neg (P \wedge Q) \equiv \neg P \vee \neg Q$
  - $\neg (P \vee Q) \equiv \neg P \wedge \neg Q$

| $A$ | $B$ | $A \rightarrow B$ |
|-----|-----|-------------------|
| $t$ | $t$ | $t$               |
| $t$ | $f$ | $f$               |
| $f$ | $t$ | $t$               |
| $f$ | $f$ | $t$               |

| $A$ | $B$ | $\neg A$ | $\neg A \vee B$ |
|-----|-----|----------|-----------------|
| $t$ | $t$ | $f$      | $t$             |
| $t$ | $f$ | $f$      | $f$             |
| $f$ | $t$ | $t$      | $t$             |
| $f$ | $f$ | $t$      | $t$             |

# 二階述語論理

## 一階述語論理

- $X$ は犬である．  $\text{犬}(X)$
- 全ての犬にはしっぽがある．  $\forall \text{犬}, \text{しっぽ}(\text{犬})$

- 一階述語論理では， 個体（指差しで数えられるもの）しか量的に表せない．
- 二階述語論理では， メタ的な視点から関数， 述語自体も量的に表せる．
  - $\forall S \exists x, (x \in S \vee x \notin S)$
  - これは， 任意の $x$  が集合  $S$  に属する・属さないのどちらかであることを主張
  - 例）生物 $x$ は動物の集合 $S$ （哺乳類， 爬虫類， 魚類， 鳥類…）に属する・属さないかのどちらかである． この場合， 哺乳類や爬虫類といった述語をまとめて $S$ にしている．

休憩

# 知識システム内での論理の活用

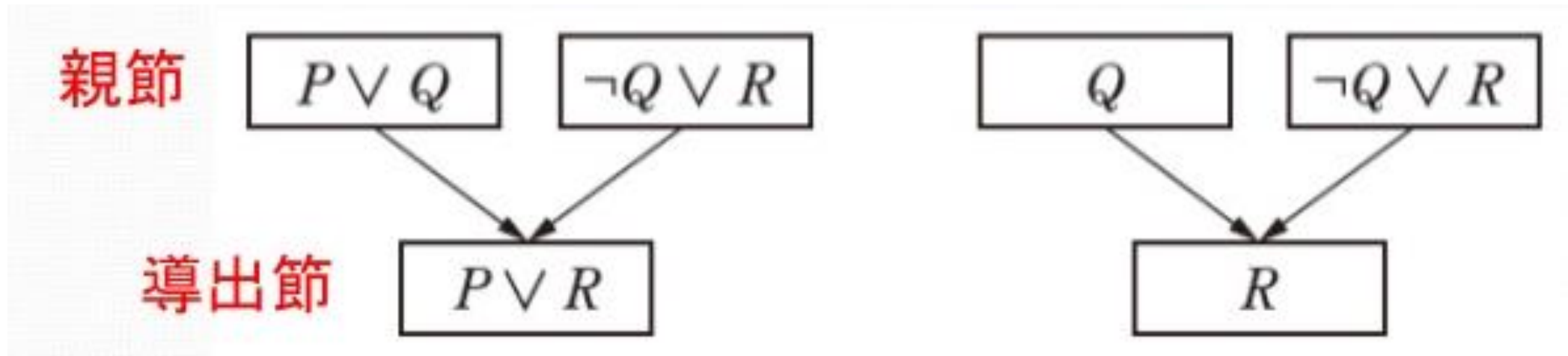
1. 知識の情報を論理式の有限集合（以下  $K$  とする）で表す.
2. 問い合わせ  $p$  を与える
3.  $p$  が  $K$  により矛盾するか，導出原理に基づく反駁法（背理法）で調べる.

**導出原理：2つ以上の命題から新たな命題を導き出したものが妥当であるという原理**



# 導出原理

- 命題に $Q$ と $\neg Q$ が含まれているとき、新たな命題が導き出せる



## モーダスポネンス (三段論法)

P 「ソクラテスは人間である」

Human(ソクラテス)

Q 「全ての人間は寿命がある」

$\forall x, \text{Human}(x) \rightarrow \text{Haslife}(x)$



R 「ソクラテスには寿命がある」

Haslife(ソクラテス)

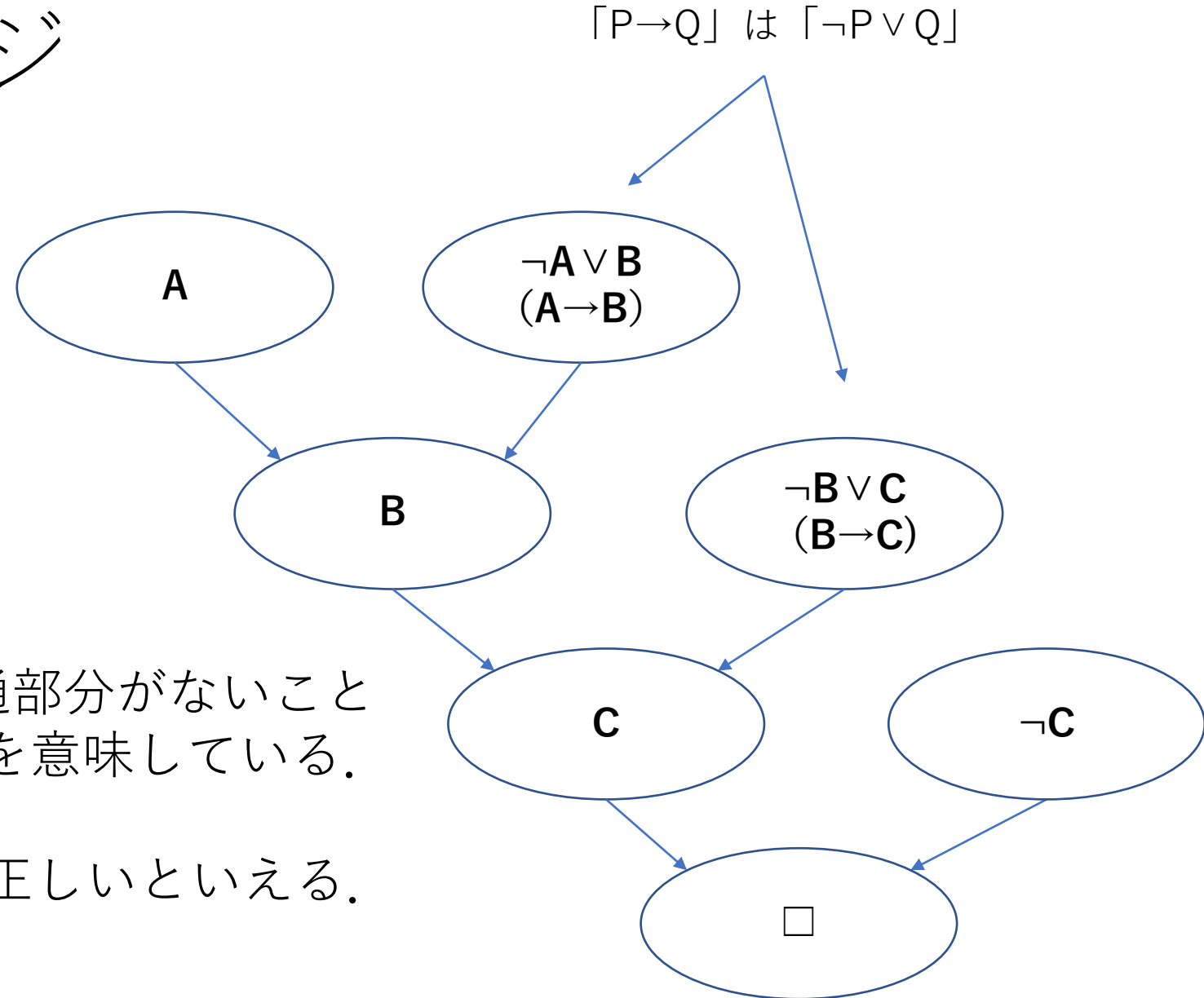


# 反駁法による検証

- 「Aであり， $A \rightarrow B$  かつ  $B \rightarrow C$  ならば Cである」という導出された命題を反駁法（背理法）で検証する.
- 反駁法では，結論を否定して「 $\neg C$ である」とする.
- 結論が正しくなければ，論理式は空集合になる.
  - 結論が間違っていたら空集合（null）になったので，正しい結論は「Cである」と考える.



# 反駁法のイメージ



$\square$  は空集合であり， $C$ と $\neg C$ に共通部分がないことから，要素を1つも持たないことを意味している．

ここから，結論は $\neg C$ ではなく $C$ が正しいといえる．

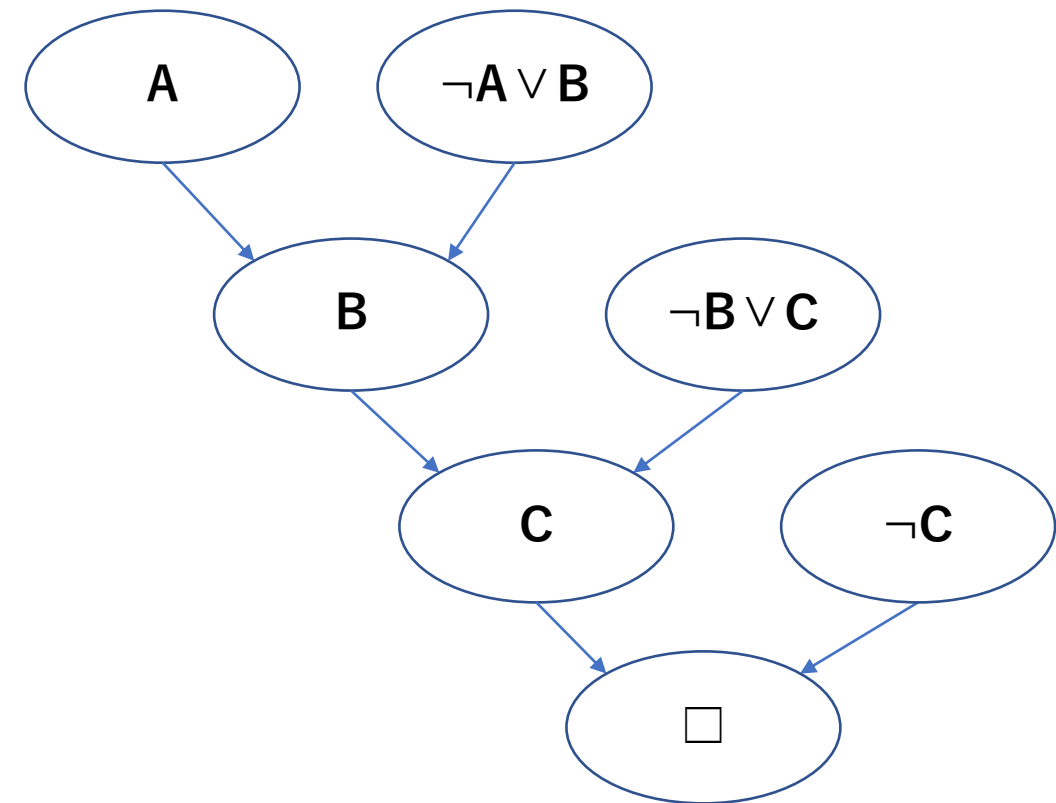
# 論理表現の長所と短所

## 長所

- 知識表現と推論手法が一体である
- 結果の健全性と完全性
- 理論的に整備されている

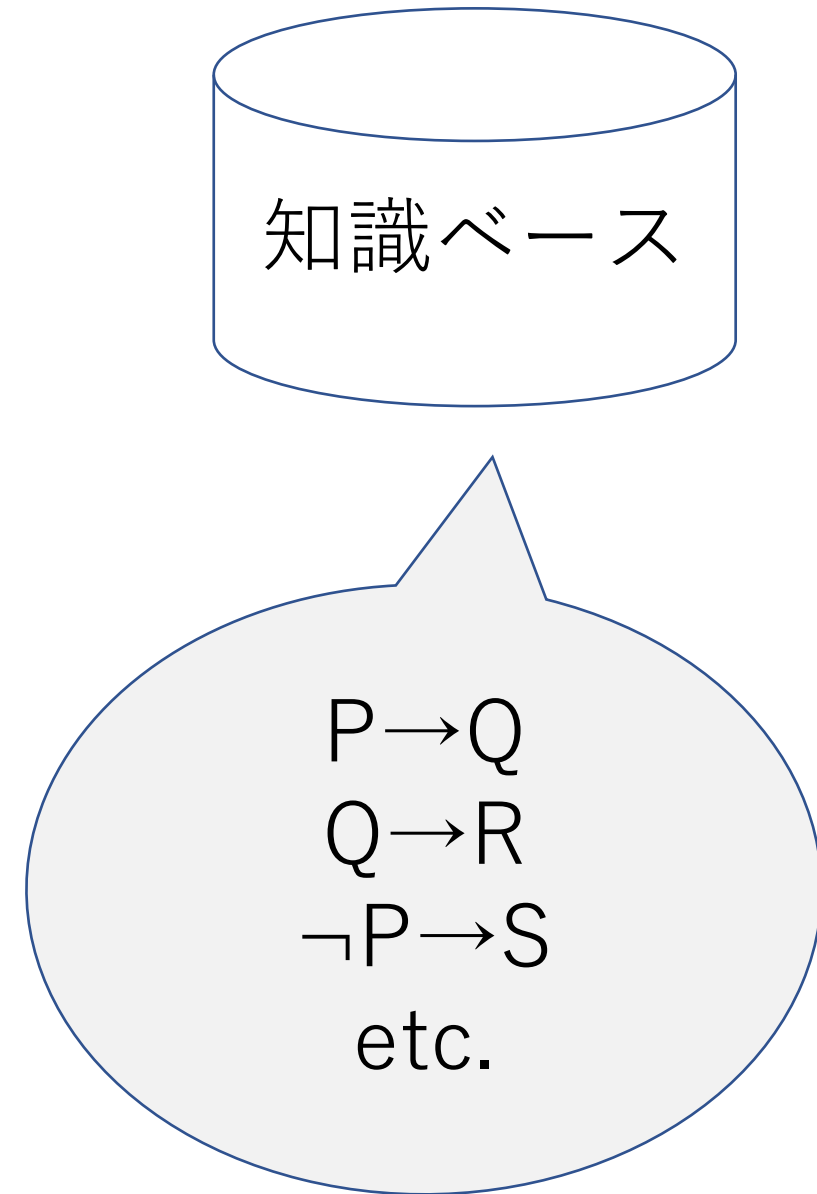
## 短所

- 導出に労力がかかる
- 階層的な表現は難しい
- 曖昧な表現は処理できない



# 知識ベースでの活用

知識を論理式のように入れておくことで、  
推論して結果を導くことができる。





# Prologによるプログラミング

導出原理を用いた定理証明や応答文作成のプログラミング言語として、Prologがある。

2011年に、本で取り上げられて再注目された。

ソフトバンクのPepperくんも内部にPrologベースのプログラムがある。

## Prologのプログラミング例

```
parent(Pam, Bob)
parent(Tom, Bob)
parent(Tom, Liz)
parent(Bob, Pat)
?- parent(Bob, Pat)
出力>> yes
?- parent(X, Liz)
出力>> Tom
```

# 導出原理の活用例

- これから、マス目を使ったダンジョン探検の例を紹介する.



- モンスターがいる. その周囲縦横のマスでは, 異臭を感じる

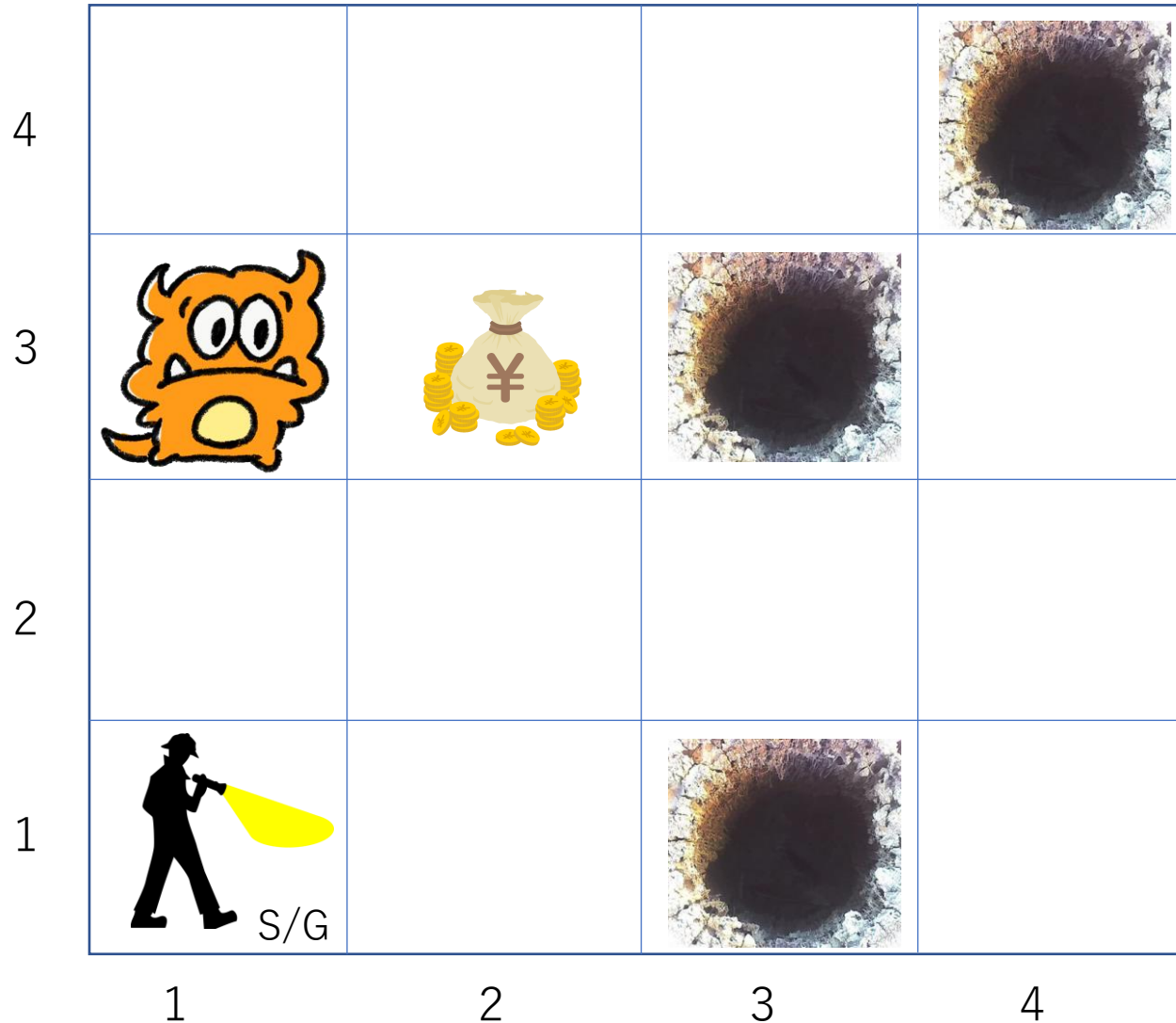


- 落とし穴がある. その周囲縦横のマスは, 風を感じる



- 財宝. 持って帰りたい

# 活用例：ダンジョン探検



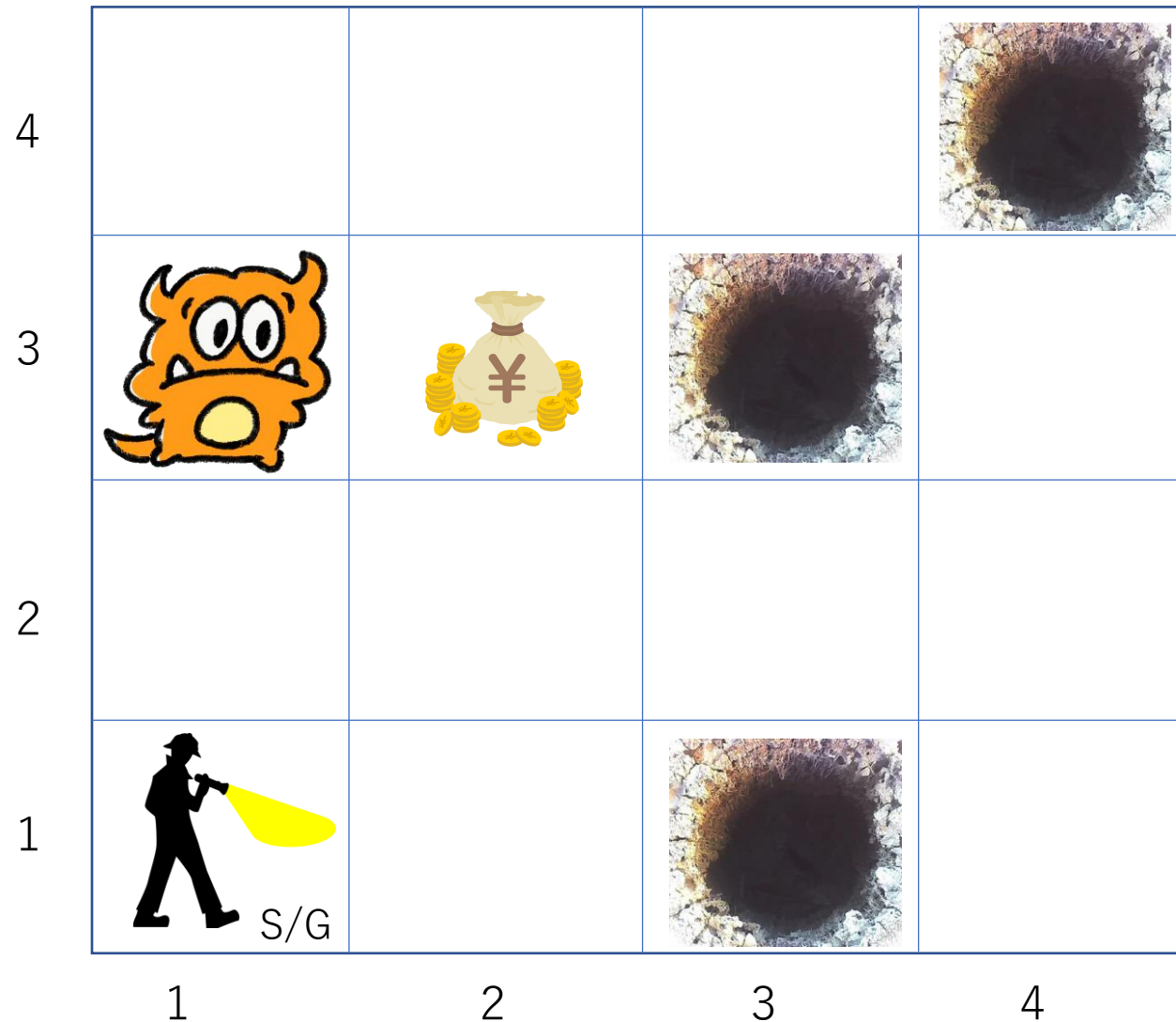
エージェントはスタート位置から財宝をとって元の位置に戻りたい.

ただし, モンスターと落とし穴の部屋を避ける必要がある.

場所は,  $x$  と  $y$  のように2つの数字で表現する.

例) 一番下段の落とし穴は(3, 1)

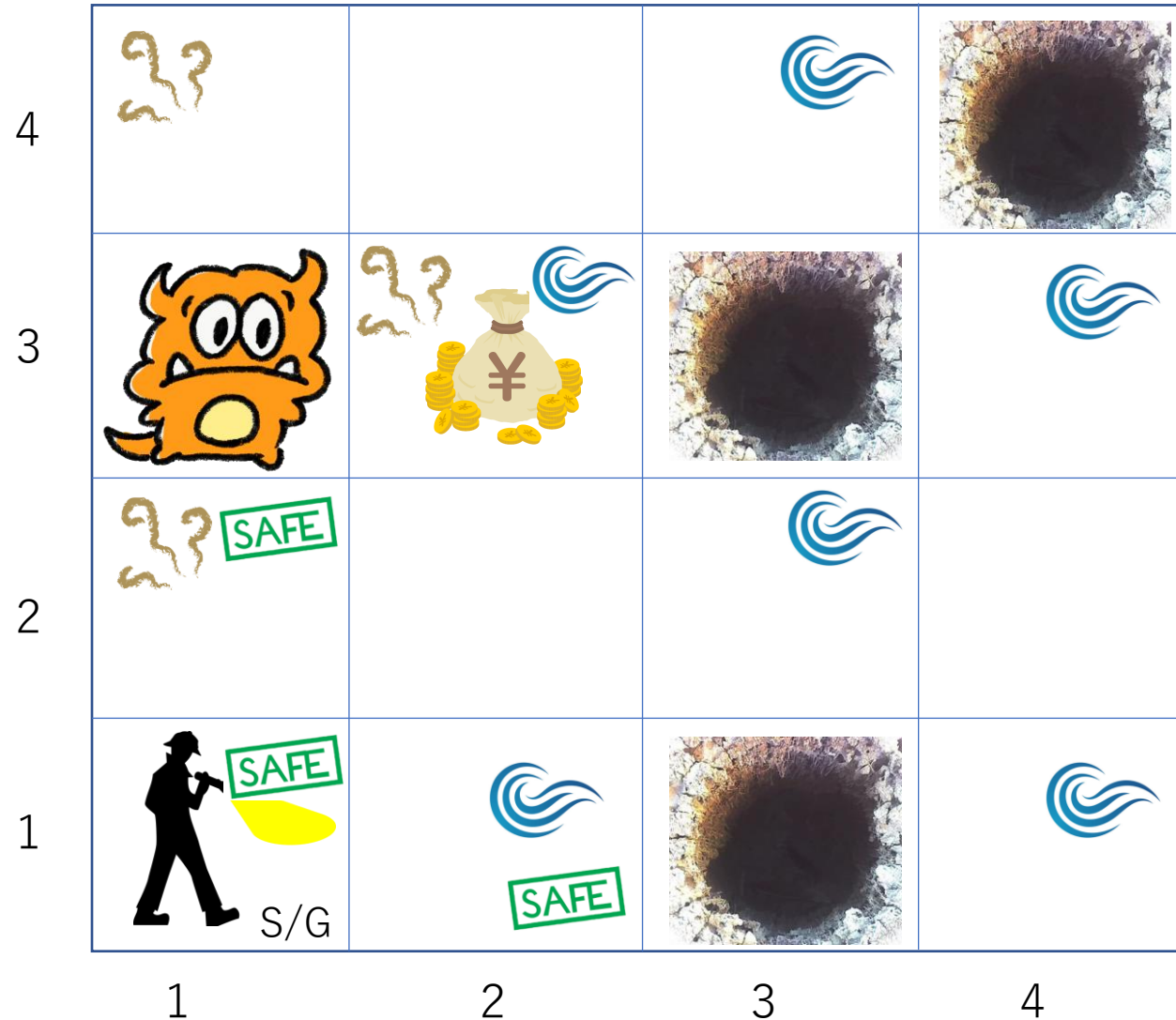
# 活用例：ダンジョン探検



- 背景知識：臭いとモンスター，風と落とし穴が関係するという予習した知識
- 事実：この部屋には臭いや風があるかどうかという，現場での知識





















これらを組み合わせてエージェントは適切な行動を決定する.

# 活用例：ダンジョン探検



- (1, 1)にはなにもないので  
(1, 2) と (2, 1)は安全
- 探索した結果をエージェント  
は記憶する


# 活用例：ダンジョン探検


|   |  |   |  |   |
|---|--|---|--|---|
| 4 |   |   |    |    |
| 3 |   | <br>  | <br>      |    |
| 2 | <br> |   | <br>      |   |
| 1 | <br>S/G   | <br><br> | <br> |  |
|   | 1  | 2   | 3  | 4   |






















- エージェントがランダムに (2, 1)に進んだとする.
- 風を感じるため, (2, 2)か (3, 1)に穴があると分かる



# 活用例：ダンジョン探検

 風での危険予想場所

 臭いでの危険予想場所

|   |   |   |   |   |
|---|---|---|---|---|
| 4 |    |   |   |  |
| 3 |     |    |    |   |
| 2 |    |    |   |   |
| 1 | <br>S/G  | <br>  |    |   |
|   | 1   | 2   | 3   | 4   |

- 一度戻り，(1, 1)を經由して(1, 2)へ
- 臭いがあるため，(1, 3)か(2, 2)にモンスターが居るとわかる.
- (2,1)では匂いを感じなかったので，(2,2)にモンスターはいないとわかる.
- 風を感じないため，(2, 2)には落とし穴がないとわかる

# 活用例：ダンジョン探検



風での危険予想場所



臭いでの危険予想場所

|   |         |      |   |   |
|---|---------|------|---|---|
| 4 |         |      |   |   |
| 3 |         |      |   |   |
| 2 |         |      |   |   |
| 1 | <br>S/G | <br> |   |   |
|   | 1       | 2    | 3 | 4 |

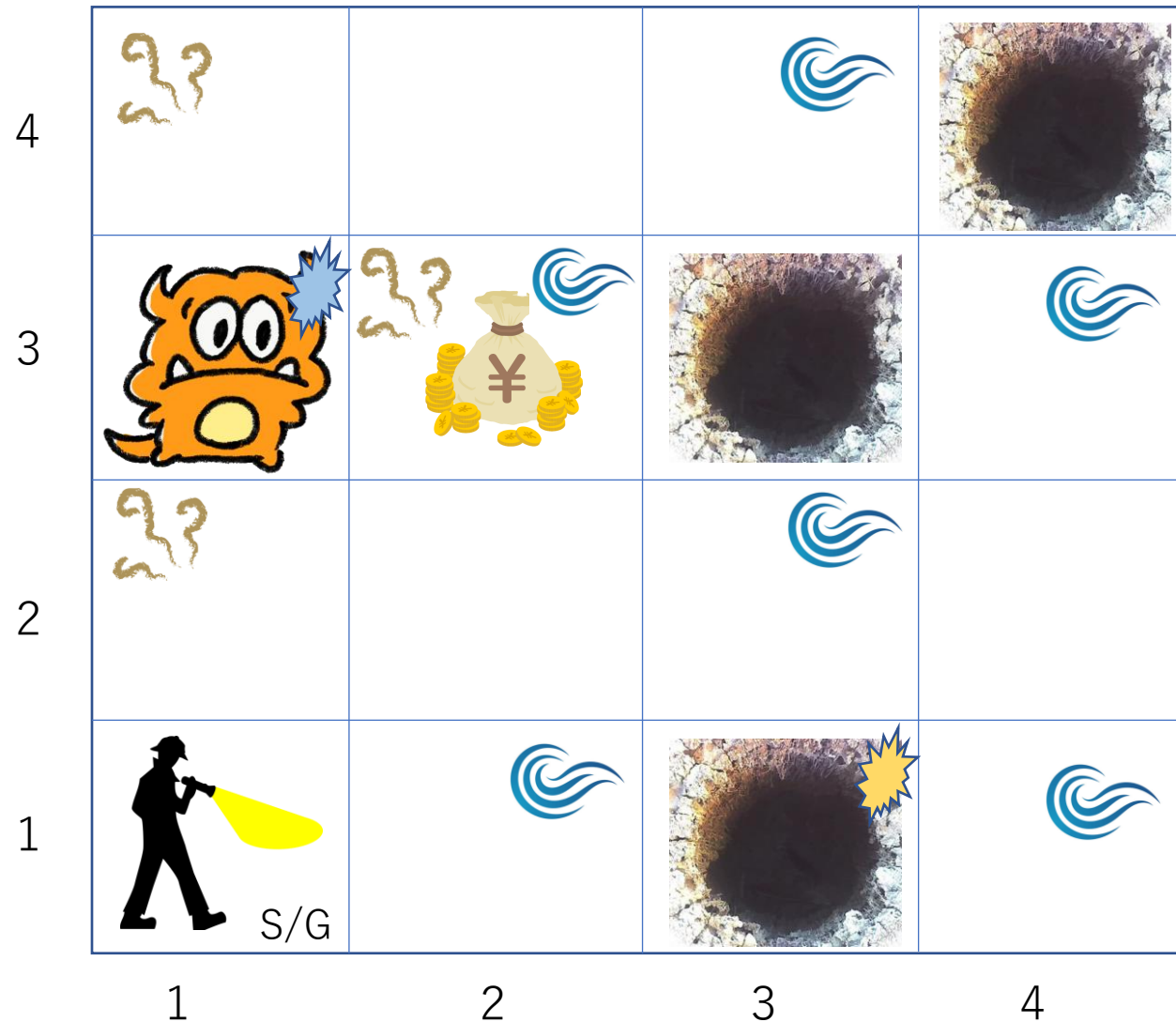
- (2, 2)が安全とわかったので行く
- (2, 1)で感じた風は(3, 1)であることが確定する
- この後, (2, 3) へ進み, お金を手に入れる. ( (3, 2)へ行ったとしても一度引き返す. )

# ダンジョン攻略

- エージェントは無事に帰ってこれました.
- この推論を，論理的に行うとどの様になるかを見ていきましょう.



# 活用例：ダンジョン探検



## 背景知識

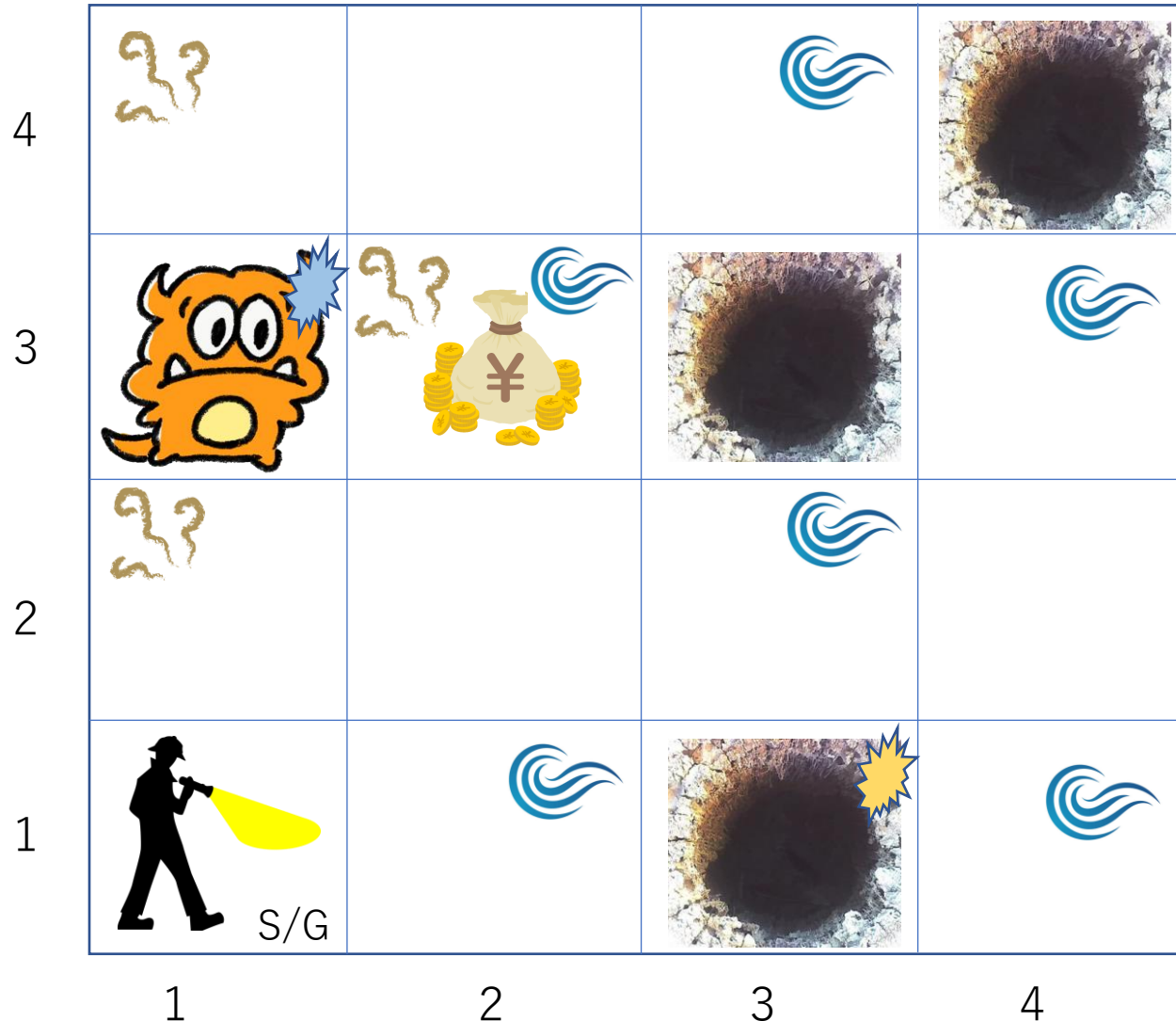
- (2,1)にモンスターがいれば, (1,1) に臭い  
 $\text{Monster}_{21} \rightarrow \text{Smell}_{11}$
- (2,1)に落とし穴があれば, (1,1) で風を感じる  
 $\text{Hole}_{21} \rightarrow \text{Wind}_{11}$
- (2,1)は, モンスターも落とし穴も無ければ安全  
 $\neg \text{Monster}_{21} \wedge \neg \text{Hole}_{21} \rightarrow \text{Safe}_{21}$

## 事実

(1,1)には臭いが無い:  $\neg \text{Smell}_{11}$

(1,1)には風が無い:  $\neg \text{Wind}_{11}$

# 活用例：ダンジョン探検



Monster21  $\rightarrow$  Smell11  $\dots$ (1)

Hole21  $\rightarrow$  Wind11  $\dots$ (2)

$\neg$ Monster21  $\wedge$   $\neg$ Hole21  $\rightarrow$  Safe21 $\dots$  (3)

$\neg$ Smell11  $\dots$ (4)

$\neg$ Wind11  $\dots$ (5)

$P \rightarrow Q$  は  $\neg P \vee Q$ なので, (1)を変形し, (4)と合わせる



















$(\neg \text{Monster21} \vee \text{Smell11}) \wedge \neg \text{Smell11}$

$\equiv \neg \text{Monster21} \dots$  (6)

同様に, (2)と (5) より,  $\neg \text{Hole21} \dots$  (7)

(3) (6) (7)より, Safe21 を導ける. 同様に(1,2)も

# 活用例：ダンジョン探検

|   |   |   |  |   |
|---|---|---|--|---|
| 4 |    |   |    |    |
| 3 |    |          |    |    |
| 2 |   |   |    |   |
| 1 | <br>S/G  |    |  |  |
|   | 1   | 2   | 3  | 4   |

## 背景知識

- 落とし穴があれば，風を感じる。

Hole31→Wind21

Hole22→Wind21

- モンスターが居れば，臭いを感じる

Monster31 → Smell21

Monster22 → Smell21

## 事実

ここでは風を感じる：Wind21

ここでは臭いを感じない：¬Smell21



# 活用例：ダンジョン探検



風での危険予想場所



臭いでの危険予想場所

|   |         |   |   |   |
|---|---------|---|---|---|
| 4 |         |   |   |   |
| 3 |         |   |   |   |
| 2 |         |   |   |   |
| 1 | <br>S/G |   |   |   |
|   | 1       | 2 | 3 | 4 |

Hole31→Wind21 …(8)

Hole22→Wind21 …(9)

Monster31 → Smell21 …(10)

Monster22 → Smell21 …(11)

Wind21 …(12)

¬Smell21 …(13)


(10), (11)の対偶より,


¬Smell21 → ¬Monster31…(14)






















¬Smell21 → ¬Monster22…(15)

(13), (14),(15)より, (2,2)と(3,1)にモンスターはいない. 落とし穴の位置は不明.

# 活用例：ダンジョン探検

 風での危険予想場所

 臭いでの危険予想場所

|   |   |   |   |   |
|---|---|---|---|---|
| 4 |    |   |   |    |
| 3 |     |    |   |    |
| 2 |    |    |   |   |
| 1 | <br>S/G  | <br>  |   |  |
|   | 1   | 2   | 3   | 4   |

## 背景知識

- 落とし穴があれば風を感じる。

Hole22→Wind12

Hole13→Wind12

- モンスターがいれば匂いを感じる。

Monster(1,3) → Smell(1,2)

Monster(2,2) → Smell(1,2)


## 事実


Smell12






















¬Wind12



# 活用例：ダンジョン探検

 風での危険予想場所

 臭いでの危険予想場所

|   |   |   |   |   |
|---|---|---|---|---|
| 4 |    |   |   |    |
| 3 |     |    |   |    |
| 2 |    |    |   |   |
| 1 | <br>S/G  |     |   |  |
|   | 1   | 2   | 3   | 4   |

Hole22→Wind12...(16)

Hole13→Wind12...(17)

Monster(1,3) → Smell(1,2) ...(18)

Monster(2,2) → Smell(1,2) ...(19)

Smell12...(20)

¬Wind12...(21)

(16)(17)の対偶より,



¬Wind12 → ¬Hole22...(22)




















¬Wind12 → ¬Hole13...(23)

(21)(22)(23)より, (2,2)と(1,3)に落とし穴はない.

(13)(15)より, (2,2)にモンスターもいない.

# 活用例：ダンジョン探検

-  風での危険予想場所
-  臭いでの危険予想場所

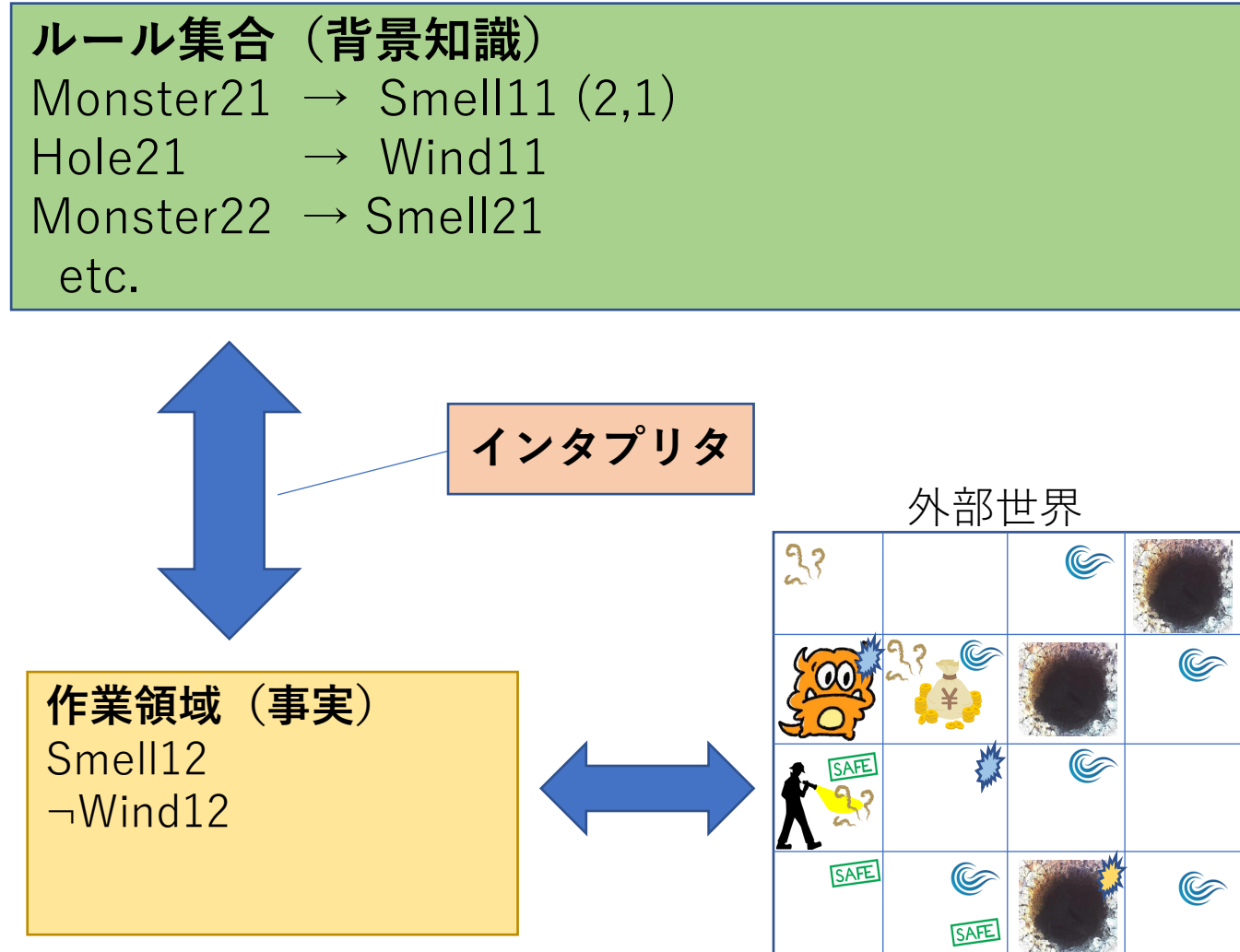
|   |  |   |  |   |
|---|--|---|--|---|
| 4 |   |   |    |    |
| 3 |   |    |    |    |
| 2 |   |    |    |   |
| 1 | <br>S/G   |     |  |  |
|   | 1  | 2   | 3  | 4   |

以下略

# インタプリタの役割

自分や皆さんが行ったことが、  
実はインタプリタの役割です。

1. 外部世界が記号化されたものを読み取り、どのルールを使うかをルール集合から持ってくる。
2. 作業領域から使って、結果を導き出す。

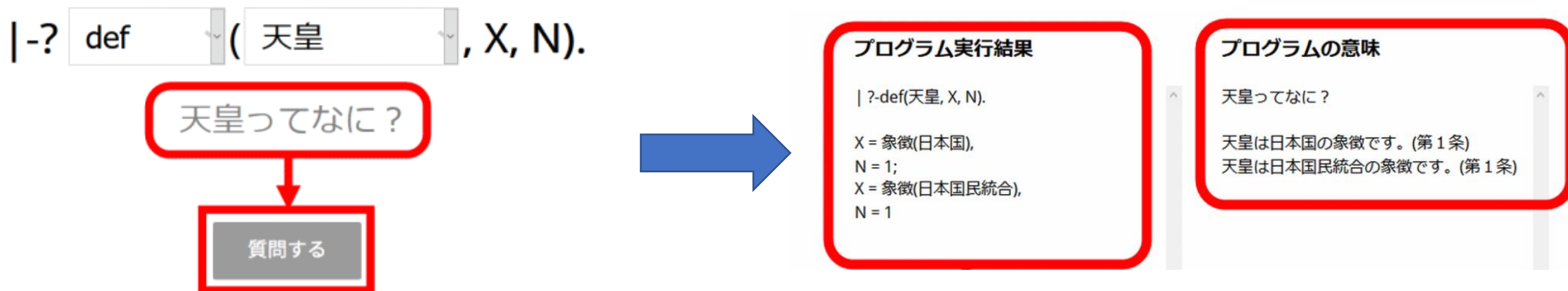


# 論理プログラミングのデモ

## 論理憲法

日本国憲法の条文を論理プログラミングを用いて表記し、日本国憲法の内容をQ & A方式で分かりやすく表示してくれるウェブアプリ。実行に関係するソースコードも見れる。

<http://bitlaw-jp.github.io/logicon-system/index.html>



# 論理プログラムのソースコード

```
from kanren import *
people = var()
kisoku = lall((eq, (var(), var(), var(), var()), people),
              (membero, ('Bob', var(), 'blue', var()), people),
              (membero, (var(), 'ITエンジニア', var(), 'Canada'), people),
              (membero, ('Zakosho', var(), var(), 'USA'), people),
              (membero, (var(), var(), 'black', 'Australia'), people),
              (membero, ('James', 'ITエンジニア', var(), var()), people),
              (membero, ('Kenji', var(), var(), 'Australia'), people),
              (membero, (var(), 'パイロット', var(), 'France'), people),
              (membero, (var(), '芸人', var(), var()), people),)
solver = run(0, people, kisoku)
output = [question for question in solver[0] if '芸人' in question][0][0]
print(output + ' is 芸人')
```

solver[0] #参考

Pythonで論理プログラミングを行う  
～芸人は誰かを推測する～

<https://qiita.com/pkaiky/items/95d54a4abad2314bfb53>

- Bobは青い服を着ている。
- ITエンジニアはカナダに住んでいる。
- Zakoshoはアメリカに住んでいる。
- 黒い服を着ている人はオーストラリアに住んでいる。
- JamesはITエンジニアである。
- Kenjiはオーストラリアに住んでいる。
- パイロットはフランスに住んでいる。

•芸人をしているのは誰だ？

# レビューシートの提出

- 今日の授業に関するレビューシートを，manabaから提出すること．

後日不明点があれば，多胡まで．

7号館5階 第9実験室内 第9研究室

tago@net.it-chiba.ac.jp