Russel's Paradox

Sets

$\{1\}$

$\{1, 3, 5\}$

$\{1, 2, 3, \ldots\}$

$\{0, 2, -2, 4, -4, \ldots\}$

$\{1,$ "e", $\ddot{\smile}\}$

$\{\{1\}, \{1, 3, 5\}\}$

Infinitely Receeding Set

$\{\{\{\ldots\}\}\}$

$\{1, \{1, \{\ldots\}\}\}$

R is the set of all sets that contain themselves

$R \stackrel{?}{\in} R$

If $R \in R$ then $R \in R$.

If $R \notin R$ then $R \notin R$.

P is the set of all sets that do not contain themselves.

$P \stackrel{?}{\in} P$

Suppose $P \in P$. Then $P \notin P$.

Suppose $P \notin P$. Then $P \in P$.

((

$\{1\}$ Function $a \to a = 1$

$\{1, 3\}$ Function $a \to a = 1$ Or $a = 3$

$\{\{1\}, \{1, 3\}, \ldots\}$ (the set containing all sets containing 1) Function $a \to a\ 1$

P Function $a \to$ Not $(a\ a)$

$P \in P$?

(Function $a \to$ Not $(a\ a)$) (Function $a \to$ Not $(a\ a)$)

Not ((Function $a \to$ Not $(a\ a)$) (Function $a \to$ Not $(a\ a)$))

Not (Not ((Function $a \to$ Not $(a\ a)$) (Function $a \to$ Not $(a\ a)$)))

R Function $a \to a\ a$

(Function $a \to a\ a$) (Function $a \to a\ a$)

Summate $(n) = \sum_{i=1}^{n} i =$ If $n=0$ Then 0 Else summate $(n-1) + n$

Let summate (x) = Function $n \rightarrow$ If $n=0$ Then 0 Else summate $(n-1) + n$ (e_1)

In

summate 5 (e_2)

Let $a = a$ In a

Self-passing recursion

Let summate' =

Function self → Function $n \rightarrow$ If $n=0$ Then 0 Else self self $(n-1) + n$ (S')

In

Let summate = summate' summate' In

summate 5

(x = summate', e_1 = S')

Let summate' = S' In

Let summate = summate' summate' In

summate 5 (e_2)

Let summate = S' S' In

summate 5

Let summate =

Function $n \rightarrow$ If $n=0$ Then 0 Else S' S' $(n-1) + n$

In

summate 5

```
Let summate' = Function recurse → Function n →
                    If n=0 Then 0 Else recurse (n-1) + n
In
Let recurser =
    Function n →
        Let recurser' = Function self → Function k →
                            summate' (self self) k
        In
        (recurser' recurser') n
In
Let summate = summate' recurser In
summate 5
```

```
Let summate' = Function recurse → Function n →
                    If n=0 Then 0 Else recurse (n-1) + n
In
Let recurser =
    Function f →
        Function n →
            Let recurser' = Function self → Function k →
                                f (self self) k
            In
            (recurser' recurser') n
In
Let summate = recurser summate' In
summate 5
```

Y-combinator