# A 부록: 파이썬 IR의 요약된 문법 및 의미구조

## A.1 요약된 문법

$$
\begin{array}{rcl}
\mathit{Program} & ::= & \mathit{stmt} \\
\mathit{stmt} & ::= & \texttt{Pass} \\
& \mid & \texttt{Expr(}\mathit{expr}\texttt{)} \\
& \mid & \texttt{Seq(}\mathit{stmt}\ ,\ \mathit{stmt}\texttt{)} \\
& \mid & \texttt{Assign(}\mathit{left\text{-}val}\ ,\ \mathit{expr}\texttt{)} \\
& \mid & \texttt{If(}\mathit{expr}\ ,\ \mathit{stmt}\ ,\ \mathit{stmt}\texttt{)} \\
& \mid & \texttt{ForIn(}\mathit{left\text{-}val}\ ,\ \mathit{expr}\ ,\ \mathit{stmt}\texttt{)} \\
& \mid & \texttt{Return(}\mathit{expr}\texttt{)} \mid \texttt{Continue} \mid \texttt{Break} \\
& \mid & \texttt{Let(}\mathit{left\text{-}val},\ (\mathit{expr} \mid \texttt{Undef})\ ,\ \mathit{stmt}\texttt{)} \\
& \mid & \texttt{FuncDef(}\mathit{id}\ ,\ \mathit{expr}^*\ ,\ \mathit{stmt}\texttt{)} \\
& \mid & \texttt{Try(}\mathit{stmt},\ (\mathit{exceptionclass} \mid \texttt{Undef}\ ,\ \mathit{stmt})^*\ ,\ \mathit{stmt}\ ,\ \mathit{stmt}\texttt{)} \\
& \mid & \texttt{Raise(}\mathit{exceptionclass}\ ,\ (\mathit{expr} \mid \texttt{Undef})\texttt{)} \\
\mathit{expr} & ::= & n \in \mathbb{Z} \mid r \in \mathit{Float} \mid s \in \mathit{String} \mid \texttt{True} \mid \texttt{False} \mid \texttt{None} \\
& \mid & \mathit{object} \\
& \mid & \texttt{BinOp(}\mathit{Binary\text{-}op}\ ,\ \mathit{expr}\ ,\ \mathit{expr}\texttt{)} \\
& \mid & \texttt{UnaryOp(}\mathit{Unary\text{-}op}\ ,\ \mathit{expr}\texttt{)} \\
& \mid & \mathit{left\text{-}val} \\
& \mid & \texttt{Call(}\mathit{expr.id}\ ,\ \mathit{expr}^*\texttt{)} \\
& \mid & \texttt{LibCall(}\mathit{id}\ ,\ \mathit{expr}^*\texttt{)} \\
& \mid & \mathit{exceptionclass} \\
\mathit{left\text{-}val} & ::= & \texttt{Name(}\mathit{id}\texttt{)} \\
& \mid & \texttt{Attribute(}\mathit{expr}\ ,\ \mathit{id}\texttt{)} \\
& \mid & \texttt{Subscript(}\mathit{expr}\ ,\ \mathit{expr}\texttt{)} \\
\mathit{exceptionclass} & ::= & \texttt{Except(}\mathit{Exception\text{-}id}\texttt{)} \\
\mathit{id} & ::= & \mathit{string} \\
\mathit{object} & ::= & \mathit{id} \mid \texttt{List(}\mathit{expr}^*\texttt{)} \mid \texttt{Dictionary(}(\mathit{expr}\ ,\ \mathit{expr})^*\texttt{)} \mid \texttt{Tuple(}\mathit{expr}^*\texttt{)} \\
\mathit{Numeric\text{-}op} & ::= & \texttt{+} \mid \texttt{-} \mid \texttt{/} \mid \texttt{//} \mid \texttt{\%} \mid \texttt{**} \\
\mathit{Bool\text{-}op} & ::= & \texttt{and} \mid \texttt{or} \mid \texttt{is} \mid \texttt{is not} \mid \texttt{in} \mid \texttt{not in}
\end{array}
$$

$$
\begin{array}{rcl}
\textit{Compare-op} & ::= & \texttt{==} \mid \texttt{!=} \mid \texttt{<} \mid \texttt{>} \mid \texttt{<=} \mid \texttt{>=} \\
\textit{Binary-op} & ::= & \textit{Numeric-op} \mid \textit{Bool-op} \mid \textit{Compare-op} \mid \textit{Bit-op} \\
\textit{Bit-op} & ::= & \texttt{|} \mid \texttt{\&} \mid \texttt{\^{}} \mid \texttt{<<} \mid \texttt{>>} \\
\textit{Unary-op} & ::= & \texttt{not} \mid \texttt{-} \\
\textit{Exception-id} & ::= & \textit{id} \mid \texttt{Exception} \mid \texttt{IndexError} \mid \texttt{ZeroDivisionError} \\
& & \texttt{KeyError} \mid \texttt{AttributeError} \mid \texttt{NameError}
\end{array}
$$

## A.2 도메인

$$
\begin{array}{llrcl}
\sigma & \in & Env & = & Id \xrightarrow{fin} Addr \\
\mathcal{H} & \in & Heap & = & Addr \xrightarrow{fin} Value \\
v & \in & Value & = & ValueExpr + Addr + Object + Func + Exception \\
& & & & + \, Cont + \{ \texttt{ None, NotImpl, Undef } \} \\
s & \in & ValueExpr & = & String + Boolean + Float + Int \\
e & \in & Exception & = & Id + \{\texttt{Exception, NameError, AttributeError,} \\
& & & & \texttt{ZeroDivisionError, KeyError, IndexError}\} \\
o & \in & Object & = & (Id + Int + String) \xrightarrow{fin} Value \\
f & \in & Func & = & Id \times Id^{*} \times Stmt \times Env \\
\kappa & \in & Cont & = & \{ \texttt{run}, \texttt{cnt}, \texttt{ brk} \} \\
l & \in & Addr & = & Int(\text{address space}) \\
i, x & \in & Id & = & String(\text{identifier}) \\
stmt & \in & Stmt & = & \texttt{Pass} + \ldots
\end{array}
$$

$$
\begin{array}{lcc}
\texttt{"length"}, \texttt{""}(\text{empty string}), \ldots & \in & String \\
\_\_len\_\_, \_\_add\_\_, \_\_getitem\_\_, \ldots & \in & Id
\end{array}
$$

요약 환경 ($Env$) 는 식별자(Id)와 요약 메모리 주소 ($Addr$)의 매핑(mapping)이다. 또한, 요약 메모리 ($Heap$) 는 요약 메모리 주소 ($Addr$)와 요약 값 ($Value$)의 매핑

(mapping)이다. 요약 값은 String, Float, Int, Boolean 등의 값 ($ValueExpr$) 또는 요약 메모리 주소, 객체 ($Object$), 함수 ($Func$), 예외 클래스 ($Exception$), 프로그램 진행 값 ($Cont$), NULL 값 `None`, 구현 안된 값 `NotImpl`, 정의 안된 값 `Undef` 이다. 객체는 Id 또는 Int, String으로 부터 요약 값으로의 매핑이다. 함수는 함수 이름을 나타내는 식별자, 함수 인자를 나타내는 식별자, 함수 몸체 (($Cont$)), 요약 환경으로 구성된다. 프로그램 진행 값은 프로그램 진행(pass, `run`), continue(`cnt`), 프로그램 진행 중단(break, `brk`) 중 하나이다.

A.3의 의미구조에서 도메인 $Object$는 글씨체의 차이에 의해 구분된다. 도메인 $String$ 는 딕셔너리 또는 `__getitem__` 함수의 도메인이다 (e.g., `obj["key"]`). $Id$ 는 객체의 속성 및 `__getattr__` 함수의 도메인이다 (e.g., `obj.key`). $Int$ 는 리스트나 `__getitem__` 함수의 도메인이다 (e.g., `obj[0]`).

클래스, 리스트, 딕셔너리 등 모든 파이썬의 객체들은 $Object$으로 나타낼 수 있다. 최적화를 위해서, $Object$ 와 $Func$의 실제 구현에서는 함수의 디폴트 인자등의 추가적인 정보를 가질 수 있다.

```
class A:      # A.__mro__ = (A, object)
  pass
class B(A):   # B.__mro__ = (B, A)
  pass
b = B()       # b.__mro__ = (B, A)
```

그림 1: 클래스와 객체에서의 $\_\_mro\_\_$ 호출 결과의 예

각 $Object$의 값마다 두 개의 빌트인 속성이 있다. $length$속성은 리스트와 같은 객체들에서 원소의 수와 같은 객체의 길이를 나타낸다. $__mro__$는 파이썬에서 Method Resolution Order(MRO)를 의미한다. 본 연구에서는 각 객체의 $\_\_mro\_\_$ 속성마다 길이가 2인 튜플을 할당한다. 튜플의 첫번째 원소는 객체의 클래스이다. 객체가 클래스 일 때는 객체 자신이 첫 번째 원소가 된다. 두번째 원소는 첫번째 원소의 부모 클래스 이다. 실제 파이썬의 문법과는 다르게 요약된 문법에서는 다중 상속을 지원하지 않는다. 객체의 $\_\_mro\_\_$ 속성들이 클래스

상속 트리에서 간선이 된다는 의미이다.

## A.3 의미구조

의미구조를 간략하게 나타내기 위해서 도우미 함수 $call$, $attr$, $item$를 아래와 같이 정의한다.

- $call(\mathcal{H}, f, v)$: 힙 공간 $\mathcal{H}$아래에서 인자 $v$ 로 함수 $f$를 계산한다 .

$$\frac{f = (x_f, x_1, S, \sigma_f) \qquad l_f, l_1 \notin Dom(\mathcal{H}) \qquad \sigma_f[x_f \mapsto l_f, x_1 \mapsto l_1], \mathcal{H}[l_f \mapsto f, l_1 \mapsto v_1] \vdash S \Rightarrow v', \mathcal{H}'}{call(\mathcal{H}, f, v) = v', \mathcal{H}'}$$

- $attr(\mathcal{H}, o, i)$: 힙 공간 $\mathcal{H}$아래에서 객체 $o$의 속성 $i$ 을 검색한다 . 속성 $i$가 객체$o$의 속성이 아니면, 함수 $__getattr__$에 인자 $i$를 주어 호출한다. 만약 함수 $__getattr__$가 $i$에 없으면, 함수$__mro__$를 활용해 기본 클래스 (base class)를 찾는다.
- $item(\mathcal{H}, o, i)$: 위와 같으나 $i$ 와 함수 $__getitem__$를 사용한다.

$$\frac{o(i) = v}{attr(\mathcal{H}, o, i) = v, \mathcal{H}} \ (\textsc{Attr-Direct})$$

$$\frac{i \notin Dom(o) \qquad o(\_\_getattr\_\_) = f \qquad call(\mathcal{H}, f, i) = v, \mathcal{H}'}{attr(\mathcal{H}, o, i) = v, \mathcal{H}'} \ (\textsc{Attr-Method})$$

$$\frac{i, \_\_getattr\_\_ \notin Dom(o) \qquad o(\_\_mro\_\_)(0) \neq o \qquad attr(\mathcal{H}, o(\_\_mro\_\_)(0), i) = v, \mathcal{H}'}{attr(\mathcal{H}, o, i) = v, \mathcal{H}'} \ (\textsc{Attr-Instance})$$

$$\frac{i, \_\_getattr\_\_ \notin Dom(o) \qquad o(\_\_mro\_\_)(0) = o \qquad attr(\mathcal{H}, o(\_\_mro\_\_)(1), i) = v, \mathcal{H}'}{attr(\mathcal{H}, o, i) = v, \mathcal{H}'} \ (\textsc{Attr-Class})$$

**Expression의 의미구조**

$$\boxed{\sigma, \mathcal{H} \vdash E \Rightarrow v, \mathcal{H}'}$$

$$\frac{E \in \mathbb{Z} + \mathit{Float} + \mathit{String} + \mathit{Bool} + \{\ \texttt{None}\ \}}{\sigma, \mathcal{H} \vdash E \Rightarrow c, \mathcal{H}} \quad (\textsc{Constraints})$$

$$\frac{\sigma, \mathcal{H} \vdash E \Rightarrow l, \mathcal{H}' \qquad \mathcal{H}'(l) = v, v \notin \{\mathit{Object}, \texttt{Undef}\}}{\sigma, \mathcal{H} \vdash E \Rightarrow v, \mathcal{H}'} \quad (\textsc{Address})$$

$$\frac{i \in \mathit{Dom}(\sigma)}{\sigma, \mathcal{H} \vdash \texttt{Name(}i\texttt{)} \Rightarrow \sigma(i), \mathcal{H}} \quad (\textsc{Id})$$

$$\frac{\sigma, \mathcal{H} \vdash E_1 \Rightarrow v_1, \mathcal{H}' \qquad \sigma, \mathcal{H}' \vdash E_2 \Rightarrow v_2, \mathcal{H}'' \qquad o = \{0 \mapsto v_1, 1 \mapsto v_2, \texttt{"\$length"} \mapsto 2\} \qquad l \notin \mathit{Dom}(\mathcal{H}'')}{\sigma, \mathcal{H} \vdash \texttt{List(}E_1, E_2\texttt{)} \Rightarrow l, \mathcal{H}''[l \mapsto o]} \quad (\textsc{List})$$

$$\frac{\sigma, \mathcal{H} \vdash E_1 \Rightarrow v_1, \mathcal{H}' \qquad \sigma, \mathcal{H}' \vdash E_2 \Rightarrow v_2, \mathcal{H}'' \qquad o = \{0 \mapsto v_1, 1 \mapsto v_2, \texttt{"\$length"} \mapsto 2\} \qquad l \notin \mathit{Dom}(\mathcal{H}'')}{\sigma, \mathcal{H} \vdash \texttt{Tuple(}E_1, E_2\texttt{)} \Rightarrow l, \mathcal{H}''[l \mapsto o]} \quad (\textsc{Tuple})$$

$$\frac{\sigma, \mathcal{H} \vdash E_1 \Rightarrow v_1, \mathcal{H}' \qquad \sigma, \mathcal{H}' \vdash E_2 \Rightarrow v_2, \mathcal{H}'' \qquad \sigma, \mathcal{H} \vdash E_3 \Rightarrow v_3, \mathcal{H}''' \qquad \sigma, \mathcal{H}' \vdash E_4 \Rightarrow v_4, \mathcal{H}'''' \qquad o = \{0 \mapsto v_1, 1 \mapsto v_2, 2 \mapsto v_3, 3 \mapsto v_4, \texttt{"\$length"} \mapsto 2\} \qquad l \notin \mathit{Dom}(\mathcal{H}'''')}{\sigma, \mathcal{H} \vdash \texttt{Dictionary((}E_1, E_2\texttt{)}, \texttt{(}E_3, E_4\texttt{))} \Rightarrow l, \mathcal{H}''''[l \mapsto o]} \quad (\textsc{Dictionary})$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E_1 \Rightarrow v_1, \mathcal{H}' \\ \sigma, \mathcal{H}' \vdash E_2 \Rightarrow v_2, \mathcal{H}'' \\ v_1, v_2 \in \mathit{Int} + \mathit{Float} + \mathit{Bool}\end{array}}{\sigma, \mathcal{H} \vdash \texttt{BinOp(+,}\ E_1,\ E_2\texttt{)} \Rightarrow v_1 + v_2, \mathcal{H}''} \text{ (Binary Operation)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E_1 \Rightarrow l, \mathcal{H}' \\ \sigma, \mathcal{H}' \vdash E_2 \Rightarrow v_1, \mathcal{H}'' \\ \mathcal{H}''(l) = o, \mathit{attr}(\mathcal{H}'', o, \_\_\mathit{add}\_\_) = f, \mathcal{H}''' \\ \mathit{call}(\mathcal{H}''', f, v_1) = v, \mathcal{H}'''' \qquad v \neq \texttt{NotImpl}\end{array}}{\sigma, \mathcal{H} \vdash \texttt{BinOp(+,}\ E_1,\ E_2\texttt{)} \Rightarrow v, \mathcal{H}''''} \text{ (Binary Operation-LM)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E_1 \Rightarrow l_1, \mathcal{H}' \\ \sigma, \mathcal{H}' \vdash E_2 \Rightarrow l_2, \mathcal{H}'' \\ \mathcal{H}''(l_1) = o_1, \mathcal{H}''(l_2) = o_2, \_\_\mathit{add}\_\_ \notin \mathit{Dom}(o_1) \\ \mathit{attr}(\mathcal{H}'', o_2, \_\_\mathit{radd}\_\_) = f, \mathcal{H}''' \\ \mathit{call}(\mathcal{H}''', f, l_1) = v, \mathcal{H}''''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{BinOp(+,}\ E_1,\ E_2\texttt{)} \Rightarrow v, \mathcal{H}''''} \text{ (Binary Operation-RM)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E_1 \Rightarrow l_1, \mathcal{H}' \\ \sigma, \mathcal{H}' \vdash E_2 \Rightarrow l_2, \mathcal{H}'' \\ \mathcal{H}''(l_1) = o_1, \mathcal{H}''(l_2) = o_2, \mathit{attr}(\mathcal{H}'', o_1, \_\_\mathit{add}\_\_) = f_1, \mathcal{H}''' \\ \mathit{call}(\mathcal{H}''', f_1, l_2) = \texttt{NotImpl}, \mathcal{H}'''' \\ \mathcal{H}''''(l_2) = o_2', \mathit{attr}(\mathcal{H}'''', o_2', \_\_\mathit{radd}\_\_) = f_2, \mathcal{H}''''' \\ \mathit{call}(\mathcal{H}''''', f_2, l_1) = v, \mathcal{H}''''''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{BinOp(+,}\ E_1,\ E_2\texttt{)} \Rightarrow v, \mathcal{H}''''''}$$

(Binary Operation-RMN)

$$\frac{\sigma, \mathcal{H} \vdash E_1 \Rightarrow v_1, \mathcal{H}' \qquad v_1 \in \mathit{Int} + \mathit{Float} + \mathit{Bool}}{\sigma, \mathcal{H} \vdash \texttt{UnryOp(-,}E_1\texttt{)} \Rightarrow -v_1, \mathcal{H}'} \quad (\textsc{Unary Operation})$$

$$\frac{\sigma, \mathcal{H} \vdash E_1 \Rightarrow l, \mathcal{H}' \qquad \mathcal{H}'(l) = o, \mathit{attr}(\mathcal{H}', o, \_\_\mathit{neg}\_\_) = f, \mathcal{H}'' \qquad \mathit{call}(\mathcal{H}'', f, l) = v, \mathcal{H}''' \quad v \neq \texttt{NotImpl}}{\sigma, \mathcal{H} \vdash \texttt{UnryOp(-, }E_1\texttt{)} \Rightarrow -v, \mathcal{H}'''} \quad (\textsc{Unary Operation-O})$$

$$\frac{\sigma, \mathcal{H} \vdash E_1 \Rightarrow f, \mathcal{H}' \quad \sigma, \mathcal{H}' \vdash E_2 \Rightarrow v, \mathcal{H}'' \qquad \mathit{call}(\mathcal{H}'', f, v) = v, \mathcal{H}'''}{\sigma, \mathcal{H} \vdash \texttt{Call(}E_1\texttt{, }E_2\texttt{)} \Rightarrow v, \mathcal{H}'''} \quad (\textsc{Call})$$

$$\frac{\sigma, \mathcal{H} \vdash E_1 \Rightarrow o, \mathcal{H}' \quad \sigma, \mathcal{H}' \vdash E_2 \Rightarrow v, \mathcal{H}'' \qquad \mathit{attr}(\mathcal{H}'', o, \_\_\mathit{call}\_\_) = f, \mathcal{H}''' \qquad \mathit{call}(\mathcal{H}''', f, v) = v, \mathcal{H}''''}{\sigma, \mathcal{H} \vdash \texttt{Call(}E_1\texttt{, }E_2\texttt{)} \Rightarrow v, \mathcal{H}''''} \quad (\textsc{Call-M})$$

$$\frac{\sigma, \mathcal{H} \vdash E_1 \Rightarrow o, \mathcal{H}' \quad \sigma, \mathcal{H}' \vdash E_2 \Rightarrow v, \mathcal{H}'' \qquad \mathit{attr}(\mathcal{H}'', o, \_\_\mathit{call}\_\_) = f, \mathcal{H}''' \qquad \mathit{call}(\mathcal{H}''', f, v) = v, \mathcal{H}''''}{\sigma, \mathcal{H} \vdash \texttt{LibCall(}E_1\texttt{, }E_2\texttt{)} \Rightarrow v, \mathcal{H}''''} \quad (\textsc{LibCall})$$

$$\frac{\sigma, \mathcal{H} \vdash E \Rightarrow l, \mathcal{H}' \qquad \mathcal{H}'(l) = o, o(i) = v}{\sigma, \mathcal{H} \vdash \texttt{Attribute(}E\texttt{, }i\texttt{)} \Rightarrow v, \mathcal{H}'} \quad (\textsc{Attribute})$$

$$\frac{\sigma, \mathcal{H} \vdash E \Rightarrow l, \mathcal{H}' \qquad \mathcal{H}'(l) = o, \mathit{attr}(\mathcal{H}', o, i) = v, \mathcal{H}''}{\sigma, \mathcal{H} \vdash \texttt{Attribute(}E\texttt{, }i\texttt{)} \Rightarrow v, \mathcal{H}''} \quad (\textsc{Attribute-M})$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E_1 \Rightarrow l, \mathcal{H}' \\ \sigma, \mathcal{H}' \vdash E_2 \Rightarrow x, \mathcal{H}'' \\ \mathcal{H}''(l) = o,\ \_\_getitem\_\_ \notin Dom(o) \\ x \in Int + String,\ o(x) = v\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Subscript(}E_1,\ E_2\texttt{)} \Rightarrow v, \mathcal{H}''} \quad \text{(SUBSCRIPT)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E_1 \Rightarrow l, \mathcal{H}' \\ \sigma, \mathcal{H}' \vdash E_2 \Rightarrow v, \mathcal{H}'' \\ \mathcal{H}''(l) = o, attr(\mathcal{H}'', o, \_\_getitem\_\_) = f, \mathcal{H}''' \\ call(\mathcal{H}''', f, v) = v', \mathcal{H}''''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Subscript(}E_1,\ E_2\texttt{)} \Rightarrow v', \mathcal{H}''''} \quad \text{(SUBSCRIPT-M)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash e \Rightarrow v, \mathcal{H}' \\ v \in \{\texttt{Exception,NameError,AttributeError,} \\ \texttt{ZeroDivisionError,IndexError,KeyError}\}\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Exception(}e\texttt{)} \Rightarrow v} \quad \text{(EXCEPTIONCLASS)}$$

**Statement의 의미구조**

$$\boxed{\sigma, \mathcal{H} \vdash S \Rightarrow v, \mathcal{H}'}$$

$$\frac{S \in \{\texttt{Pass,Continue,Break}\}}{\sigma, \mathcal{H} \vdash S \Rightarrow \kappa, \mathcal{H}} \quad \text{(FLOW CONTROL)}$$

$$\frac{\sigma, \mathcal{H} \vdash E \Rightarrow v, \mathcal{H}'}{\sigma, \mathcal{H} \vdash \texttt{Expr}(E) \Rightarrow \texttt{run}, \mathcal{H}'} \quad \text{(EXCECUTE)}$$

$$\frac{\sigma, \mathcal{H} \vdash E \Rightarrow v, \mathcal{H}'}{\sigma, \mathcal{H} \vdash \texttt{Return}(E) \Rightarrow v, \mathcal{H}'} \quad \text{(RETURN)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash S_1 \Rightarrow \texttt{run}, \mathcal{H}' \\ \sigma, \mathcal{H}' \vdash S_2 \Rightarrow v, \mathcal{H}''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Seq}(S_1, S_2) \Rightarrow v, \mathcal{H}''} \quad \text{(SEQUENCE-R)}$$

$$\frac{\sigma, \mathcal{H} \vdash S_1 \Rightarrow \texttt{brk}, \mathcal{H}'}{\sigma, \mathcal{H} \vdash \texttt{Seq}(S_1, S_2) \Rightarrow \texttt{brk}, \mathcal{H}'} \quad \text{(Sequence-FB)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash S_1 \Rightarrow v, \mathcal{H}' \\ v \neq \texttt{brk} \qquad v \neq \texttt{cnt} \\ \sigma, \mathcal{H}' \vdash S_2 \Rightarrow \texttt{brk}, \mathcal{H}''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Seq}(S_1, S_2) \Rightarrow \texttt{brk}, \mathcal{H}''} \quad \text{(Sequence-SB)}$$

$$\frac{\sigma, \mathcal{H} \vdash S_1 \Rightarrow \texttt{cnt}, \mathcal{H}'}{\sigma, \mathcal{H} \vdash \texttt{Seq}(S_1, S_2) \Rightarrow \texttt{run}, \mathcal{H}'} \quad \text{(Sequence-FC)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash S_1 \Rightarrow v, \mathcal{H}' \\ v \neq \texttt{brk} \qquad v \neq \texttt{cnt} \\ \sigma, \mathcal{H}' \vdash S_2 \Rightarrow \texttt{cnt}, \mathcal{H}''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Seq}(S_1, S_2) \Rightarrow \texttt{cnt}, \mathcal{H}''} \quad \text{(Sequence-SC)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash S_1 \Rightarrow v, \mathcal{H}' \\ v \neq \texttt{run} \qquad v \neq \texttt{brk} \qquad v \neq \texttt{cnt} \\ \sigma, \mathcal{H}' \vdash S_2 \Rightarrow v', \mathcal{H}''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Seq}(S_1, S_2) \Rightarrow v', \mathcal{H}''} \quad \text{(Sequence)}$$

$$\frac{\begin{array}{c}f = (x_f, x_i, S_1, \sigma) \qquad l \notin \mathit{Dom}(\mathcal{H}) \\ \sigma[x_f \mapsto l], \mathcal{H}[l \mapsto f] \vdash S_2 \Rightarrow v, \mathcal{H}'\end{array}}{\sigma, \mathcal{H} \vdash \texttt{FunDef}(x_f, x_i, S_1, S_2) \Rightarrow v, \mathcal{H}'} \quad \text{(Def Function)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E \Rightarrow v, \mathcal{H}' \qquad l \notin \mathit{Dom}(\mathcal{H}') \\ \sigma[i \mapsto l], \mathcal{H}'[l \mapsto v] \vdash S \Rightarrow v', \mathcal{H}''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Let}(i, E, S) \Rightarrow v', \mathcal{H}''} \quad \text{(Let)}$$

$$\frac{\begin{array}{c}l \notin \mathit{Dom}(\mathcal{H}) \\ \sigma[i \mapsto l], \mathcal{H}[l \mapsto \texttt{Undef}] \vdash S \Rightarrow v, \mathcal{H}'\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Let}(i, \texttt{Undef}, S) \Rightarrow v, \mathcal{H}'} \quad \text{(Let-U)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E \Rightarrow v, \mathcal{H}' \\ v \in \mathit{Int} + \mathit{Float} + \mathit{Boolean} + \mathit{String} + \mathit{Object} + \mathit{Func} \\ v \notin \{\ \texttt{False},\texttt{None},\texttt{0},\texttt{0.0},\texttt{""}\ \} \\ \sigma, \mathcal{H}' \vdash S_t \Rightarrow v_t, \mathcal{H}''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{If(}E, S_t, S_f\texttt{)} \Rightarrow v_t, \mathcal{H}''} \quad \text{(If-TC)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E \Rightarrow v, \mathcal{H}' \\ v \in \{\ \texttt{False},\texttt{None},\texttt{0},\texttt{0.0},\texttt{""}\ \} \\ \sigma, \mathcal{H}' \vdash S_f \Rightarrow v_f, \mathcal{H}''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{If(}E, S_t, S_f\texttt{)} \Rightarrow v_f, \mathcal{H}''} \quad \text{(If-FC)}$$

$$\frac{\begin{array}{c}\sigma(i) = l \\ \sigma, \mathcal{H} \vdash E \Rightarrow v, \mathcal{H}'\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Assign(Name(}i\texttt{)}, E\texttt{)} \Rightarrow \texttt{run}, \mathcal{H}'[l \mapsto v]} \quad \text{(Assign-N)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E_1 \Rightarrow l, \mathcal{H}' \\ \sigma, \mathcal{H}' \vdash E_2 \Rightarrow v, \mathcal{H}'' \\ \mathcal{H}''(l) = o, o' = o[i \mapsto v]\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Assign(Attribute(}E_1, i\texttt{)}, E_2\texttt{)} \Rightarrow \texttt{run}, \mathcal{H}''[l \mapsto o']} \quad \text{(Assign-A)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E_1 \Rightarrow l, \mathcal{H}_1 \\ \sigma, \mathcal{H}_1 \vdash E_2 \Rightarrow v_i, \mathcal{H}_2 \\ \sigma, \mathcal{H}_2 \vdash E_3 \Rightarrow v, \mathcal{H}_3 \\ \mathcal{H}_3(l) = o, \mathit{attr}(\mathcal{H}_3, o, \_\_\mathit{setitem}\_\_) = \mathcal{H}_4, f \\ \mathit{call}(\mathcal{H}_4, f, (v_i, v)) = v', \mathcal{H}_5\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Assign(Subscript(}E_1, E_2\texttt{)}, E\texttt{)} \Rightarrow \texttt{run}, \mathcal{H}_5} \quad \text{(Assign-S)}$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E \Rightarrow l, \mathcal{H}' \\ \mathcal{H}'(l) = o \qquad \mathit{call}(\mathcal{H}', \_\_\mathit{len}\_\_, o) = n, \mathcal{H}_0 \\ l' \notin \mathit{Dom}(\mathcal{H}_0) \qquad \mathit{item}(\mathcal{H}_0, o, 0) = v_0, \mathcal{H}'_0 \\ \sigma[i \mapsto l'], \mathcal{H}'_0[l' \mapsto v_0] \vdash S \Rightarrow \kappa_1, \mathcal{H}_1 \\ \vdots \\ \mathit{item}(\mathcal{H}_{n-1}, o, n-1) = v_{n-1}, \mathcal{H}'_{n-1} \\ \sigma[i \mapsto l'], \mathcal{H}'_{n-1}[l' \mapsto v_{n-1}] \vdash S \Rightarrow \kappa_n, \mathcal{H}_n \\ \forall i, \kappa_i \in \{\mathsf{run}, \mathsf{cnt}\} \\ (\text{o가 S안에서 바뀌지 않는다고 가정}) \end{array}}{\sigma, \mathcal{H} \vdash \mathtt{ForIn(}i, E, S\mathtt{)} \Rightarrow \mathsf{run}, \mathcal{H}_n} \quad (\textsc{ForIn})$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E \Rightarrow l, \mathcal{H}' \\ \mathcal{H}'(l) = o \qquad \mathit{call}(\mathcal{H}', \_\_\mathit{len}\_\_, o) = n, \mathcal{H}_0 \\ l' \notin \mathit{Dom}(\mathcal{H}_0) \qquad \mathit{item}(\mathcal{H}_0, o, 0) = v_0, \mathcal{H}'_0 \\ \sigma[i \mapsto l'], \mathcal{H}'_0[l' \mapsto v_0] \vdash S \Rightarrow \kappa_1, \mathcal{H}_1 \\ \vdots \\ \mathit{item}(\mathcal{H}_{k-1}, o, k-1) = v_{k-1}, \mathcal{H}'_{k-1} \\ \sigma[i \mapsto l'], \mathcal{H}'_{k-1}[l' \mapsto v_{k-1}] \vdash S \Rightarrow \mathsf{brk}, \mathcal{H}_k \\ \forall i, \kappa_i \in \{\mathsf{run}, \mathsf{cnt}\} \\ (\text{o가 S안에서 바뀌지 않는다고 가정}, k \leq n) \end{array}}{\sigma, \mathcal{H} \vdash \mathtt{ForIn(}i, E, S\mathtt{)} \Rightarrow \mathsf{run}, \mathcal{H}_k} \quad (\textsc{ForIn-B})$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E \Rightarrow e, \mathcal{H}' \\ e \in \{\mathtt{Exception,NameError,AttributeError,} \\ \mathtt{ZeroDivisionError,IndexError,KeyError}\} \end{array}}{\sigma, \mathcal{H} \vdash \mathtt{Raise(}E, \mathtt{Undef)} \Rightarrow \mathsf{brk}, \mathcal{H}'} \quad (\textsc{Raise-U})$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E_1 \Rightarrow e, \mathcal{H}' \\ e \in \{\texttt{Exception}, \texttt{NameError}, \texttt{AttributeError}, \\ \texttt{ZeroDivisionError}, \texttt{IndexError}, \texttt{KeyError}\} \\ \sigma, \mathcal{H}' \vdash E_2 \Rightarrow v, \mathcal{H}''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Raise(}E_1, E_2\texttt{)} \Rightarrow \texttt{brk}, \mathcal{H}''} \quad (\textsc{Raise})$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E \Rightarrow e, \mathcal{H}' \qquad e \in \{\texttt{Undef}, \texttt{Exception}\} \\ \sigma, \mathcal{H}' \vdash S_1 \Rightarrow \texttt{brk}, \mathcal{H}'' \\ \sigma, \mathcal{H}'' \vdash S_2 \Rightarrow v', \mathcal{H}''' \\ \sigma, \mathcal{H}''' \vdash S_4 \Rightarrow v'', \mathcal{H}''''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Try(}S_1, E, S_2, S_3, S_4\texttt{)} \Rightarrow v'', \mathcal{H}''''} \quad (\textsc{Try-UX})$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E \Rightarrow e, \mathcal{H}' \qquad e \in \{\texttt{Undef}, \texttt{Exception}\} \\ \sigma, \mathcal{H}' \vdash S_1 \Rightarrow v, \mathcal{H}'' \\ \sigma, \mathcal{H}'' \vdash S_3 \Rightarrow v', \mathcal{H}''' \\ \sigma, \mathcal{H}''' \vdash S_4 \Rightarrow v'', \mathcal{H}''''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Try(}S_1, E, S_2, S_3, S_4\texttt{)} \Rightarrow v'', \mathcal{H}''''} \quad (\textsc{Try-UNX})$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E \Rightarrow e, \mathcal{H}' \qquad e \notin \{\texttt{Undef}, \texttt{Exception}\} \\ \sigma, \mathcal{H}' \vdash S_1 \Rightarrow \texttt{brk}, \mathcal{H}'' \\ \sigma, \mathcal{H}'' \vdash S_2 \Rightarrow v', \mathcal{H}''' \\ \sigma, \mathcal{H}''' \vdash S_4 \Rightarrow v'', \mathcal{H}''''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Try(}S_1, E, S_2, S_3, S_4\texttt{)} \Rightarrow v'', \mathcal{H}''''} \quad (\textsc{Try-X})$$

$$\frac{\begin{array}{c}\sigma, \mathcal{H} \vdash E \Rightarrow e, \mathcal{H}' \qquad e \notin \{\texttt{Undef}, \texttt{Exception}\} \\ \sigma, \mathcal{H}' \vdash S_1 \Rightarrow v, \mathcal{H}'' \\ \sigma, \mathcal{H}'' \vdash S_3 \Rightarrow v', \mathcal{H}''' \\ \sigma, \mathcal{H}''' \vdash S_4 \Rightarrow v'', \mathcal{H}''''\end{array}}{\sigma, \mathcal{H} \vdash \texttt{Try(}S_1, E, S_2, S_3, S_4\texttt{)} \Rightarrow v'', \mathcal{H}''''} \quad (\textsc{Try-NX})$$