Prerekvizity

Sekvence, seznamy, řetězce

- Uspořádané posloupnosti prvků proměnlivé délky
- Např. $\langle 1, 2, 3 \rangle$
- Množinu všech sekvencí prvků z množiny A značíme A^*
- $\{a, b, c\}^* = \{\langle\rangle, \langle a \rangle, \langle b \rangle, \langle c \rangle, \langle a, a \rangle, \langle a, b \rangle, \langle a, c \rangle, \langle b, a \rangle, \dots\}$
- Prázdnou sekvenci budeme značit $\langle\rangle$ (někdy se značí ε)
- $n$tý prvek sekvence a píšeme jako $a(n)$

Základní množiny

Hodnoty

- Množina všech logických hodnot $bool = \{true, false\}$
- Množina všech čísel
  $num = \{\ldots, -2, \ldots, -1.5, \ldots, 0, \ldots, 0.1, \ldots, 1, 2, \ldots\}$
- Množina všech znaků $char = \{a, b, \ldots, ?, \$, \ldots\}$
- Množina všech adres $addr = \{\#1, \#2, \#3, \ldots\}$
- Množina všech hodnot $value = bool \cup num \cup char \cup addr \cup \{null\}$

Pozn.: řetězce jsou objekty skládající se z jednotlivých znaků. My je ale kvůli zjednodušení většinou budeme používat jako atomické entity.

Jména

- Množina všech jmen atributů $fieldName = \{a, b, \ldots, count, \ldots\}$
- Množina všech jmen lokálních proměnných
  $localName = \{this, a, b, \ldots, count, \ldots\}$
- Množina všech jmen proměnných $varName = fieldName \cup localName$
- Množina všech jmen metod
  $methodName = \{a, b, \ldots, f, \ldots, getSize, \ldots\}$
- Množina všech jmen tříd
  $className = \{a, b, \ldots, Expression, \ldots, java.util.List, \ldots\}$

"Užitečné" množiny

Objekty

- Dvojice složené ze jména třídy a mapování (funkce) jmen atributů na hodnoty
- Množina všech objektů $object = className \times (fieldName \to value)$
- Kvůli větší čitelnosti máme pro objekty upravenou notaci, např. $(Pair, \{(first, 2), (second, 3)\})$ zapisujeme jako $Pair(first = 2, second = 3)$
- Můžeme si nadefinovat přístupové metody $class : object \to className$ a $field : (object \times fieldName) \to value$
  - $class(o) = \mathrm{Pr}_1(o)$
  - $field(o, f) = (\mathrm{Pr}_2(o))(f)$

Haldy

- Halda je mapování z adres na objekty (zanedbáváme, že v reálném stroji se objekt do jedné paměťové buňky obvykle nevejde)
  $heap = addr \rightarrow object$
- Pokud máme haldu h a adresu a, pak objekt na této adrese získáme voláním $h(a)$

Zásobníky

- Náš zásobník je složitější než ten hardwarový; chceme zohlednit vnořená volání metod a vnořování bloků
- $stack = ((localName \rightarrow value)^*)^*$
- Kvůli čitelnosti opět používáme upravenou notaci
- Hodnotu (lokální) proměnné x na vrcholu zásobníku s získáme jako $s(1, i)(x)$, kde i je největší přirozené číslo takové, že $s(1, i)(x)$ je definováno

Zásobníky kódu

- Předpokládáme, že máme definovanou množinu příkazů *statement* (za chvíli uvidíme jak)
- Množina zásobníků kódu je $codeStack = (statement^*)^*$
- V zásobníku kódu *cs* se jako první vykoná příkaz $cs(1, 1)$

Oblasti statických proměnných

- Množina všech oblastí statických proměnných
  $global = (className \times fieldName) \rightarrow value$
- V oblasti statických proměnných g je hodnota proměnné $System.out$ rovna $g(System, out)$

Konfigurace virtuálního stroje

- Množina všech konfigurací virtuálního stroje
  $conf = heap \times stack \times codeStack \times global$

Příkazy

Výrazy

Pokud v je hodnota, x je proměnná, f atribut, m metoda, C třída a $e_0, \ldots, e_k$ výrazy, pak jsou výrazy i

- v,
- x,
- $!\ e_0$,
- $e_0\ \&\&\ e_1$,
- $e_0\ ||\ e_1$,
- $e_0\ ==\ e_1$,
- ...

Výrazy

- ...
- $(C)e_0$,
- $e_0.f$,
- $C.f$,
- $e_0.m(e_1, \ldots, e_k)$,
- $C.m(e_1, \ldots, e_k)$ a
- *new* $C(e_1, \ldots, e_k)$.

Cokoliv, co nevzniklo postupnou aplikací konečného počtu předchozích pravidel, výraz není. Množinu všech výrazů budeme označovat *expr*.

Příkazy

Pokud x je lokální proměnná, e_0 a e_1 výrazy, C třída a $s_1, \ldots, s_k$ příkazy, pak jsou příkazy i

- $C\ x$,
- $x = e_1$,
- $e_0.f = e_1$,
- $C.f = e_1$,
- ...

Příkazy

- ...
- e_0,
- $\{s_1, \ldots, s_k\}$,
- *if* (e_0) s_1 *else* s_2,
- *while* (e_0) s_1 a
- *return* e_0.

Cokoliv, co nevzniklo postupnou aplikací konečného počtu předchozích pravidel, příkaz není. Množinu všech příkazů budeme označovat *statement*.