Języki programowania

II UWr 2019/20

Alternatywne zadanie na pracownię

deadline: 20 stycznia 2020

Język W++

W zadaniu na pracownię rozważamy kilka rozszerzeń języka **W+** z listy 9. Podstawowym rozszerzeniem którym będziemy się zajmować jest dynamiczna alokacja pamięci (podobna do polecenia `malloc` znanego z C lub `new` pojawiającego się w Javie czy Pascalu). W tym celu rozszerzamy typy, wyrażenia i instrukcje naszego języka następująco:

$$\begin{aligned}\tau &::= \ldots \mid \mathsf{ptr}\ \tau \\ e &::= \ldots \mid *\, e \\ c &::= \ldots \mid \mathsf{var}\ x := \mathsf{new}\ e\ \mathsf{in}\ c \mid *\, x := e\end{aligned}$$

Intuicja za tymi rozszerzeniami jest następująca: do gramatyki typów (zawierających już, oprócz liczb naturalnych, rekordy i warianty) dodajemy nowy konstruktor ptr τ reprezentujący wskaźnik do wartości o typie τ. Wartość tego typu jest tworzona przez konstrukcję new e, która zapisuje wartość wyrażenia e w świeżo zaalokowanej komórce *sterty*; dla uproszczenia przyjmujemy że wartość ta musi od razu zostać zapamiętana jako nowa zmienna programu. Do komórki pamięci na którą wskazuje zmienna x o typie ptr τ możemy zapisać wartość wyrażenia e używając konstrukcji $* x := e$, a odczytać wartość przechowywaną w odpowiedniej komórce sterty możemy poprzez wyrażenie $* e$ (oczywiście wartością e musi być w tym celu wskaźnik). Zwróć uwagę że język nie pozwala na jawną *dealokację* pamięci (zakładamy że w implementacji pamięć byłaby automatycznie zwalniana przez interpreter języka), ale również że w języku nie ma wartości w rodzaju `null`, znanych z C czy Javy, która jest w tych językach wartością dowolnego typu ptr τ — ale za cenę bezpieczeństwa systemu typów.

Rozszerzenie: typy definiowane przez użytkownika

Żeby zbliżyć się jeszcze bardziej do praktycznych imperatywnych języków programowania możemy pozwolić w **W++** na deklarowanie własnych typów danych. Na przykład moglibyśmy chcieć zadeklarować typ list łączonych przechowujących liczby jako

$$t = (\mathsf{int} \times \mathsf{ptr}\ t) + ().$$

Zauważmy, że jest to typ rekurencyjny: t, które deklarujemy występuje również w swojej definicji. Przyjmujemy że takie definicje będą poprawne jeśli rekurencyjne wystąpienia pojawią się jedynie jako wskaźniki. Przyjmujemy że deklaracje takie mogą pojawić się jedynie na początku programu, tj. że typy i programy przyjmują następującą postać:

$$\begin{aligned}\tau &::= \ldots \mid t \\ d_{\mathrm{t}} &::= \mathsf{type}\ t = \tau \\ p &::= d_{\mathrm{t}}^{*}\ \mathsf{in}\ d_{\mathrm{f}}^{*}\ \mathsf{in}\ \mathsf{vars}\ (x := v)^{*}\ \mathsf{in}\ c,\end{aligned}$$

gdzie d_{f} to deklaracje funkcji (bez zmian względem **W+**).

Zadanie

Zadanie składa się z dwóch części: raportu pisemnego i implementacji języka. Raport pisemny dla w pełni zrealizowanego zadania powinien zawierać

- Definicję składni wybranego wariantu języka **W++**;
- Definicję systemu typów dla tego języka;
- Definicję semantyki operacyjnej małych kroków;
- Sformułowanie twierdzeń i ważniejszych lematów oraz wybrane fragmenty dowodu poprawności systemu typów;
- Opis ważniejszych decyzji projektowych podjętych w trakcie implementacji i nieformalny opis adekwatności implementacji względem systemu typów i semantyki operacyjnej zdefiniowanej w raporcie.

Implementacja powinna odnosić się do formalnych definicji z raportu, nie musi im jednak odpowiadać w stu procentach. W szczególności ewaluator może implementować semantykę w innym stylu niż zaprezentowany w raporcie, algorytm sprawdzania typów może implementować (w jakimś stopniu) inferencję typów, a implementowany język może wymagać większej liczby anotacji typowych niż rachunek zaprezentowany w raporcie. Są to jednak decyzje projektowe które powinny być opisane i uzasadnione w ostatniej części raportu pisemnego.

Uwaga: Pewne aspekty rozwiązania zadania mogą nie być oczywiste. W razie napotkania trudności w definicjach czy dowodach warto skontaktować się z prowadzącymi żeby je rozwiązać.

Ocena

Minimalne akceptowalne rozwiązanie musi zawierać: definicje składni, systemu typów i semantyki operacyjnej (potencjalnie w stylu ewaluacyjnym), implementację oraz opis odnoszący implementację do semantyki w wersji bez deklaracji typów.

Ponieważ zadanie nadal traktujemy jako alternatywę dla implementacji wybranego artykułu, przyjmujemy że rozwiązanie maksymalne (tj. pełny raport i implementacja dla wariantu języka z deklaracjami typów) ocenione będzie na stopień dobry+.