Tytuł

Patryk Wałach

January 2022

1 [Wstęp]

2 Ogólne wprowadzenie

2.1 Parsowanie i tokenkizowanie

2.2 Co to typy?

3 Inferencja typów w teori

3.1 System typów Hindley–Milner

3.2 Algorytm W

4 Rescript/ReasonML jako języki realizujące podobne zadania

5 Założenia i priorytety opracowanej aplikacji

Tworząc aplikację, chciałem, by język posiadał podstawowe typy danych (liczby, stringi, wartość logiczna), kilka typów generycznych (funkcje, tablice), typ 'Option', oraz możliwość tworzenia własnych typów. Dodatkowo nie powinno być potrzeby podawania typów zmiennych w większości przypadków, kompilator sam powinien wykrywać typy zmiennych na podstawie ich użycia.

Język poza zmiennymi, potrzebuje możliwości wykonywania operacji na danych, dlatego ważne było dla mnie, by zaimplementować operatory binarane, oraz unarne. Operatory te miały też spełniać ważną rolą w trakcie inferencji typów. W języku javascript operator '+' może być wykorzystywany do dodawania liczb jak i konkatenacji stringów, ważne więc było by stworzyć dwa oddzielne operatory.

Chciałem również by funkcje wieloargumentowe kompilowane były jako funkcje jednoargumentowe zwracające kolejne funkcje, co pozwala na wywoływanie funkcji bez wszystkich argumentów w celu zwrócenia funkcji przyjmującej resztę argumentów tzw. currying.

Jednym z ważniejszych elemenów każdego języka jest możliwość wykonywania różnego zbioru instrukcji, warunkowo. W tym celu planowałem zaimplementowanie instrukcji 'if', oraz 'case'. Instrukcja 'case' wykonywać ma dopasowanie do wzorca (tzw. pattern-matching), wykonywać, odpowiedni zbiór instrukcji zależnie od wprowadzonych danych. Kompilator, powinien ostrzegać, jeżeli ścieżka dla jednego z typów danych nie została zaimplementowana.

Kolejnym dość ważnym elementem języka jest brak wyrażenia 'return', które jest wykorzystywane do zwrócenia wartości z funkcji. Zamiast tego każdy bloku instrukcji powinien zwracać ostatnie wyrażenie. Pozwoli to na łatwiejsze inicjowanie zmiennych, w przypadku gdy inicializacja wymaga więcej niż jedenej linii kodu.

5.1 Opis formaly składni języka

6 Narzędzia

6.1 Język python

6.2 Parsowanie i tokenizowanie przy użyciu biblioteki sly

6.3 Środowisko nodejs do uruchomienia skompilowanego kodu

7 Implementacja

7.1 lexer

7.2 parser

7.3 inferencja typów

7.4 kompilacja

8 Opis działania

8.1 Co działa

8.2 Uwagi co do obsługi błędów

9 [Podsumowanie]

10 [spisy – rysunków, tabel, listingów ipt.]