Zadanie nr 3 na pracownię

Pewien język funkcyjny pierwszego rzędu Rozważmy język programowania, którego składnia konkretna zadana jest następującą gramatyką, opisującą pewien podzbiór S-wyrażeń typu `S-Exp` w języku Plait:

$$
\begin{array}{rcl}
p & ::= & \{\texttt{define}\ \{d_1 \ldots d_k\}\ \texttt{for}\ e\} \\
d & ::= & [\texttt{fun}\ f\ (x_1 \ldots x_l) = e] \\
e & ::= & n \\
 & | & x \\
 & | & \{e_1\ \oplus\ e_2\} \\
 & | & \{\texttt{ifz}\ e_0\ \texttt{then}\ e_1\ \texttt{else}\ e_2\} \\
 & | & \{\texttt{let}\ x\ \texttt{be}\ e_1\ \texttt{in}\ e_2\} \\
 & | & \{f\ (e_1 \ldots e_l)\} \\
\oplus & ::= & \texttt{+} \mid \texttt{-} \mid \texttt{*} \mid \texttt{<=}
\end{array}
$$

Program p w naszym języku składa się z, być może pustego (zakładamy, że $k \geq 0$), ciągu globalnych definicji funkcji, oraz wyrażenia, które może używać zdefiniowanych funkcji. Definicja funkcji zawiera informację o *unikatowej* nazwie funkcji f, reprezentowanej jako symbol w składni konkretnej, ciągu parametrów formalnych $x_1, \ldots, x_l$, być może pustego (zakładamy, że $l \geq 0$), reprezentowanych jako *parami różne* symbole, oraz z wyrażenia stanowiącego ciało funkcji. Ciało funkcji może korzystać z parametrów formalnych funkcji oraz ze wszystkich pozostałych funkcji zdefiniowanych w programie. Definicje funkcji są zatem wzajemnie rekurencyjne. Gramatyka wyrażeń dopuszcza stałe liczbowe n, zmienne x (parametry formalne funkcji i zmienne związane przez `let`) operacje binarne $\oplus$, wyrażenia warunkowe, definicje lokalne oraz wywołanie funkcji $\{f\ (e_1 \ldots e_l)\}$, gdzie l jest liczbą parametrów formalnych w definicji funkcji f.

Interpretacja programu $\{\texttt{define}\ \{d_1 \ldots d_k\}\ \texttt{for}\ e\}$ polega na obliczeniu wartości wyrażenia e przy wykorzystaniu definicji funkcji d_1,…, d_k. Jedynymi wartościami do jakich wyrażenia są ewaluowane są wartości liczbowe. W tej sytuacji, interpretacja wyrażenia $\{\texttt{ifz}\ e_0\ \texttt{then}\ e_1\ \texttt{else}\ e_2\}$ polega na obliczeniu wartości wyrażenia e_0 i wybraniu do ewaluacji wyrażenia e_1, gdy wartość e_0 jest równa 0, a e_2 w przeciwnym przypadku. Aplikacja funkcji jest gorliwa, tzn. argumenty są ewaluowane zanim funkcja zostanie wywołana. Semantyka pozostałych konstrukcji jest standardowa, z

wyjątkiem relacji <=, której zachodzenie daje wartość 0, a niezachodzenie daje dowolną (Ty wybierasz) wartość niezerową.

Oto przykładowy program napisany w naszym języku, którego wykonanie powinno dać wartość 120:

```
{define
  {[fun fact (n) = {ifz n then 1 else {n * {fact ({n - 1})}}}]}
 for
  {fact (5)}}
```

A tu mamy inny przykład (oczekiwana wartość to 0):

```
{define
  {[fun even (n) = {ifz n then 0 else {odd ({n - 1})}}]
   [fun odd (n) = {ifz n then 42 else {even ({n - 1})}}]}
 for
  {even (1024)}}
```

I jeszcze jeden (spodziewamy się wartości 9):

```
{define
  {[fun gcd (m n) = {ifz n
                     then m
                     else {ifz {m <= n}
                           then {gcd (m {n - m})}
                           else {gcd ({m - n} n)}}}]}
 for
  {gcd (81 63)}}
```

Zadanie Zaimplementuj wyżej opisany język w języku Plait. W tym celu napisz dla niego parser oraz ewaluator. Parser powinien przyjmować S-wyrażenie i produkować drzewo składni abstrakcyjnej w odpowiednio do tego zdefiniowanym typie danych lub zgłaszać wyjątek, gdy dane S-wyrażenie nie reprezentuje składniowo poprawnego programu. Ewaluator powinien następnie przyjmować program w składni abstrakcyjnej, przetwarzać definicje funkcji i obliczać wartość wyrażenia zgodnie z wyżej opisaną nieformalną semantyką języka. Oczywiście, interpreter może posiłkować się dodatkowymi strukturami danych, takimi jak środowiska etc.

Wykorzystaj szablon rozwiązania dostępny na SKOSie. W szczególności, Twoje rozwiązanie musi implementować funkcję `run`, która przyjmuje wartość typu `S-Exp` i zwraca wynik typu `Value` (zdefiniowanego jako alias dla typu `Number`). Plik z rozwiązaniem, nazwany `solution.rkt`, zgłoś przez system Web-CAT (dostęp przez odnośnik na SKOSie) do dnia **15 czerwca 2023, godz. 6:00**.