Mikołaj Szkaradek

Opis Języka

Szkarson to statycznie typowany język będący połączeniem pewnych fragmentów Rusta i C++.

Program:

Program jest listą definicji funkcji oraz zmiennych globalnych. W programie musi być funkcja `main`.

Typy:

Dostępne są 4 typy: `int`, `bool`, `string`, `void`. Literały liczbowe `int` to liczby całkowite `/([0-9])+`. Literały napisowe zapisujemy w cudzysłowach. Typ `bool` przyjmuje wartości `true` lub `false`. Typ `void` służy do przekazania funkcji informacji, że ma nic nie zwracać.

Operatory arytmetyczne i porównania:

Dostępne są operatory arytmetyczne: (`+,-,*,/,^`) oraz operatory porównawcze: (`==, !=, >, <, >=, <=`). Ich działanie jest standardowe jak w typowych językach programowania. Operatory arytmetyczne zwracają typ `int`, a porównania typ `bool`.

Zmienne i przypisania:

Możliwe jest tworzenie zmiennych o identyfikatorach takich jak w typowych językach, oraz przypisywanie do zmiennych wartości. Można przypisać wartość do zmiennej każdego typu, poza typem `void`.

Instrukcje print, while, if:

Instrukcje te są klasyczne jak w typowych językach programowania, dokładna składnia dostępna jest w gramatyce.

Funkcje:

W języku można definiować zagnieżdżone funkcje oraz korzystać z rekurencji.

Przekazywanie argumentów:

Argumenty można przekazywać przez wartość lub zmienną.

Tablice wielowymiarowe:

W języku dostępne są tablice wielowymiarowe `Type []...[] ident`. Można odwołać się do zmiennej jak w C++: `arr[i][j][k]`. Tablice są określonej długości.

Krotki:

Dostępny jest typ `tuple <[Type]>` który pozwala tworzyć dowolnie długie i zagnieżdżone krotki.

Można również dokonać przypisania <a,b,c> = <|1,true,"abc"|> które działa intuicyjnie.

Wbudowane funkcje:

W Szkarsonie dostępne są różne wbudowane funkcje do operacji na stringach.

`charAt(Expr)`, `reverse`, `append(String)`, `cut(Expr1,Expr2)`.

Oraz do operacji na tablicach:

`.len(Expr)` `.dim` zwracające odpowiednio długość tablicy w Expr-tym wymiarze, oraz liczbę wymiarów tablicy.

W gramatyce mogą pojawić się, a nawet prawdopodobnie pojawią się jeszcze pewne zmiany potrzebne w czasie implementacji.

Niektóre rzeczy mogą ulec lekkim modyfikacjom.

```
Na 15 punktów
+ 01 (trzy typy)
+ 02 (literały, arytmetyka, porównania)
+ 03 (zmienne, przypisanie)
+ 04 (print)
+ 05 (while, if)
+ 06 (funkcje lub procedury, rekurencja)
+ 07 (przez zmienną / przez wartość / in/out)
Na 20 punktów
+ 09 (przesłanianie i statyczne wiązanie)
+ 10 (obsługa błędów wykonania)
+ 11 (funkcje zwracające wartość)
Na 30 punktów
+ 12 (4) (statyczne typowanie)
+ 13 (2) (funkcje zagnieżdżone ze statycznym wiązaniem)
+ 14 (1/2) (rekordy/listy/tablice/tablice wielowymiarowe)
+ 15 (2) (krotki z przypisaniem)
+ 16 (1) (break, continue)
```

Chciałbym celować w 30 punktów z interpretera.