

Deklaracja języka - Robert Suchocki

W ramach drugiego zadania zaliczeniowego na JPP planuję napisać interpreter wersji rozszerzonej imperatywnego języka Latte - Breve

Gramatyka tego języka znajduje się w pliku breve.cf w formacie zgodnym z BNFC

Cechy języka Breve

- Typy zmiennych: int, bool, str, void
- Złożone typy: array, dict
- Zmienne, operacje przypisania, działania arytmetyczne i logiczne, porównania, wyrażenia z efektami ubocznymi ('++', '--')
- Instrukcje if, if/else, while, "pascalowy" for
- Funkcje z parametrami przez wartość i zmienną, z rekurencją i zagnieżdżaniem (z przesłanianiem, z zachowaniem poprawności statycznego wiązania identyfikatorów)
- Statyczne typowanie
- Obsługa dynamicznych błędów wykonania, wbudowane funkcje do wypisywania na wyjście, funkcje rzutujące między typami int i str

Szczegóły konstrukcji rozszerzających Latte

- Array (indeksowany wartościami int)
 - Typ [typ_wartości]
 - Tworzenie array = [rozmiar]
 - Przypisanie array[indeks] = wartość
 - Dostęp array[indeks]
 - Funkcje length
- Dict (składnia do przemyślenia, powoduje konflikty shift/reduce)
 - Typ {typ_klucza, typ_wartości}
 - Tworzenie dict = {}
 - Przypisanie dict{klucz} = wartość
 - Dostęp dict{klucz}
 - Funkcje has_key, delete_key
- "Pascalowy" for
 - Składnia for i = first..last do statement
 - Zarówno inkrementacyjny jak i dekrementacyjny
- Przekazywanie parametrów do funkcji przez zmienną
 - Keyword var
 - W deklaracji funkcji zmieni sposób przekazania parametru z przez wartość na przez zmienną
- Zagnieżdżanie funkcji

Przykłady składni

Standardowe Latte

```
int add_values(int a, int b) {
    return a + b;
}
int main () {
    int a;
    a = 21;
    int b = 21;
    int c = add_values(a, b);
    bool equal = a + b == 42;
    str success = "success!";

    if (equal && c <= 42 && c >= 42) {
        c--;
        if (c > 42 || c < 42 && equal) {
            equal = 0 == 1;
            while (!equal) {
                print(success);
                # prints success! once
                equal = true;
            }
        }
    }
    return 0;
}
```

Tablice

```
int main () {
    [int] x;
    x = [10];
    x[0] = 42;
    x[9] = 10;
    if (length(x) == x[9]) {
        print("success!");
        # prints success!
    } else {
        x[10] = 0;
        # this would produce an error
    }
    return 0;
}
```

Słowniki

```
int main () {
    {str, int} in_words;
    in_words = {};
    in_words{"forty two"} = 42;
    if (has_key(in_words, "forty two")) {
        delete_key(in_words, "forty two");
        in_words{"success!"} = 42;
    }
    # would produce an error in case key is not in dict
    if (in_words{"success!"} == 42) {
        print("success!");
        # prints success! instead
    }
    return 0;
}
```

Pętla for

```
int main () {
    int x = 0;
    for i = 1..9 do {
        x = x + i;
    }
    int y = x - 3;
    for i = 10..1 do {
        if (y % i == 0) {
            y = y / i;
        }
        if (x - 3 == 42 && y == 6) {
            print("success!");
            # prints success! once when i == 7
        }
    }
    return 0;
}
```

Parametry funkcji

```
void zero_first(var int x, int y) {
    x = 0;
    y = 0;
}

int main () {
    int x = 42;
    int y = 42;
    zero_first(x, y);
    # x == 0 && y == 42
    if (x + y == 42) {
        print("success!");
        # prints success!
    }
    return 0;
}
```

Zagnieżdżanie funkcji

```
int main () {
    int squared(var int x) {
        int power(int x, int y) {
            int result = 1;
            if (y >= 1) {
                for i = 1..y do {
                    result = result * x;
                }
            }
            return result;
        }
        x = power(x, 2);
    }
    int x = 6;
    squared(x);
    # x == 36
    if (x + 6 == 42) {
        print("success!");
        # prints success!
    }
    return 0;
}
```

Punktacja

Wszystkie (nie rozwijające się na listę podpunktów do wyboru) wymagania na 24 punkty

Z punktu 6. (2 + 2 do spełnienia punktu 13.)

- a) dwa sposoby przekazywania parametrów (przez zmienną / przez wartość)
- b) pętla for w stylu Pascala
- c) typ str, literały napisowe, wbudowane funkcje pozwalające na rzutowanie między napisami a liczbami
- d) wyrażenia z efektami ubocznymi (przypisania, operatory języka C ++, += itd)

Z punktu 11. (2)

- b) tablice indeksowane int lub coś à la listy
- c) tablice/słowniki indeksowane dowolnymi porównywalnymi wartościami; typ klucza należy uwzględnić w typie słownika

Przewidywana ocena za całość: 24 punkty