Piotr Janowiak
252481

Podręcznik użykownika

Informacje ogólne

W Języku jedna linia kodu odpowiada jednej instrukcji.

```
#include "test2.txt" //pierwsza instrukcja
int z = 10 //druga instrukcja
float y = 10.1 //trzecia instrukcja
```

Instrukcja `#include "nazwa_pliku.rozszerzenie"` pozwala na łączenie plików.

Podwójny znak *slash* (`//`) powoduje wykomentowanie tekstu znajdującego się po jego prawej stronie. Zagnieżdżone komentarze mają następującą składnię: `/* zagniezdzony komentarz */`.

```
int z = 10 //zwykly komentarz
float y = 1 /* komentarz zagniezdzony */ 0.1
```

Typy zmiennych

W Języku dostępne są 3 typy zmiennych: `int`, `float` oraz `string`. `Int` przechowuje liczby całkowite, `float` zmiennoprzecinkowe a `string` ciągi znaków.

```
int z //deklaracja zmiennej typu int
float pi = 3.14 //definicja zmiennej typu float
string hello = "hello world" //definicja ciagu znakow
```

Dostępne są również zmienne tablicowe dla typów `int` oraz `float`.

```
int[10] tab1 //deklaracja tablicy typu int o rozmiarze 10
tab1[2] = 1 //przypisanie tab1 o indeksie 2 wartosci 1
```

Zmienne tablicowe indeksowane są od 0. W przypadku przekroczenia maksymalnego indeksu tablicy indeks zmieniany jest na maksymalny możliwy.

Możliwe jest zadeklarowanie zmiennych globalnych. Zmienne globalne widoczne są w każdym miejscu programu (również w funkcjach). Zmienną globalną deklaruje się dodając słowo `global` przed zwykła deklaracją.

```
global int x //zmienna globalna
```

Operacje arytmetyczne

W Języku zdefiniowane są cztery podsatowe operacje arytmetyczne: dodawanie (+), odejmowanie (-), mnożenie (*) i dzielenie (/). Wyrażenia arytmetyczne można grupować za pomocą nawiasów.

```
int x = 10
float y = 3.2 * (10 + x)
```

Wejście/Wyjście

Do pobierania wartości od użytkownika służą instrukcje `getint` oraz `getfloat`.

```
int calk
float rzecz
```

```
calk = getint
rzecz = getfloat
```

W przypadku wpisania przez użytkownika liczby rzeczywistej podczas użycia instrukcji `getint`, wpisana liczba zostanie zrzutowana na typ `float`.

Do wypisywania wartości na ekran konsoli służą instrukcje `print` oraz `prints`. Instrukcja `print` służy do wypisywania liczb, natomiast instrukcja `prints` do wypisywania ciągów znaków.

```
int z = 10//definicja zmiennej typu int
float pi = 3.14 //definicja zmiennej typu float
string hello = "hello world" //definicja ciagu znakow
print z+pi
prints hello
```

Instrukcje sterujące przebiegiem programu

Do sterowania przebiegiem programu służą instrukcje `repeat` oraz `if`. Instrukcja `repeat` ma następującą składnię:

```
repeat ilosc_powtorzen
      //dowolne instrukcje
endrepeat
```

`Ilosc_powtorzen` może być zmienną lub dowolnym wyrażeniem arytmetycznym.

```
int z = 2
int x
x = getint
string hello = "hello"
repeat x+z
              prints hello
endrepeat
```

Instrukcja `if` ma następującą składnię:

```
if warunek
      //dowolne instrukcje
endif
```

Warunek może być dowolnym wyrażeniem logicznym w postaci: `wartosc (<|>|==|>=|<=) warosc`, gdzie wartosc to zmienna typu `int` lub `float`, bądź dowolne wyrażenie arytmetyczne (również zawierające zmienne).

```
int z = 10
float x
x = getfloat
if x > z+5
      print x
endif
```

Funkcje

Funkcje można definiować w dowolnym miejscu programu. Nie dozwolone jest jednak definiowanie funkcji w funkcjach.

Definicja funkcji ma następującą składnię:

```
function typ nazwa typ arg1 typ arg2
      //dowolne instrukcje
      ret typ (wartosc)?
endfunction
```

Typem może być `int`, `float` lub `void`. Argumentami `(typ arg1 typ arg2)` mogą być zmienne typów `int` i `float`. Instrukcja `ret` zwraca wartość funkcji i musi się pojawić zawsze na końcu definicji funkcji. Kiedy typem funkcji jest `int` lub `float` instrukcja `ret` wygląda np. w ten sposób: `ret float x`. Kiedy funkcja jest typu `void`, instrukcja return wygląda następująco : `ret void`.

```
int x
x = getint

function int power2 int x
      int val = x * x
      ret val
endfunction

print power2(x)
```