ArchiCode

Dokumentacja Języka Programowania

Szymon Kowalski, Michał Król

1.Wstęp

ArchiCode to edukacyjny, imperatywny język programowania o strukturze blokowej, który łączy cechy popularnych języków (takich jak C czy Python) z prostą, jednoznaczną składnią. ArchiCode obsługuje zmienne, funkcje ("blueprinty"), zakresy (scopy), pętle, instrukcje warunkowe oraz kolekcje danych.

2.Zmienne i typy danych

Zmienne deklaruje się przy użyciu instrukcji `define`, opcjonalnie z określeniem typu:

```
define int x = 10
```

Dostępne typy danych:

- `int` – liczby całkowite
- `float` – liczby zmiennoprzecinkowe
- `bool` – wartości logiczne: `true`, `false`
- `char` – pojedyncze znaki
- `string` – ciągi znaków

Zmienne można nadpisywać instrukcją:

x = 5

3.Zakresy (scope)

Zmienne obowiązują w ramach zakresu blokowego (czyli pomiędzy { a }):

```
{
    define int x = 3
    show x
}
```

Zmienne mogą być przesłaniane, ale dostęp do nadrzędnych jest możliwy przez:

show x@1 // odczytuje zmienną x z 1 poziomu wyżej

4.Operacje arytmetyczne

Działania:

- Dodawanie: +
- Odejmowanie: -
- Mnożenie: *
- Dzielenie: /

Przykład:

```
define int a = 2 + 3 * 5
```

Promocja typów:

- `int + float` daje `float`
- Operacje typu `bool + int` są niedozwolone

5.Operacje logiczne i porównania

Logiczne:

- and, or, not

Porównania:

- ==, !=, <, <=, >, >=

Przykład:

check x > 10 then { show "duży" } otherwise { show "mały" }

6.Instrukcje warunkowe

Instrukcja `check`:

```
check warunek then {
    // kod
} otherwise {
    // kod alternatywny
}
```

Obsługuje zagnieżdżone `check` oraz skrócone wersje bez `otherwise`.

7.Pętle

Dostępne są dwa typy:

Pętla o znanej liczbie powtórzeń:

```
repeat 5 { ... }
repeat 2 10 { ... } // od 2 do 9
```

Pętla warunkowa:

```
repeat until warunek { ... }
```

8.Funkcje (Blueprinty)

Funkcje definiuje się słowem `blueprint`:

```
blueprint Nazwa(int a, int b) delivers int wynik {
    wynik = a + b
}
```

Wywołanie:

```
show Nazwa 5 10
```

Blueprint może:

- zwracać wartość (przez `delivers typ nazwa`)
- mieć przeciążenia (parametry rozróżnia sygnatura)

9.Główna funkcja (punkt wejścia)

Program musi zawierać dokładnie jeden blok:

```
Core() delivers int wynik {
    // kod główny programu
}
```

To odpowiednik funkcji `main()`.

10.Diagnostyka i błędy

Język zapewnia:

- Informacje o błędach z numerem linii i znakiem w linii
- Wskazywanie błędów semantycznych (np. brak typu, zły typ)
- Wsparcie dla komunikatów typu: "czy chodziło ci o..."

11.Przykład

```
define int x = 5


blueprint Dodaj(int a, int b) delivers int wynik {
   wynik = a + b
}
```

```
Core() delivers int result {
   result = Dodaj 3 4
   show result
}
```

12.Komentarze

ArchiCode obsługuje komentarze:

- Jednolinijkowe: `>> komentarz`
- Wielolinijkowe: `>>> komentarz <<<`
- Inline (z zamknięciem w tej samej linii): `>> komentarz <<`

Przykład:

```
To jest komentarz

To jest wielolinijkowy komentarz <<<
```

13.Uwagi

- Nazwy funkcji z wielkiej litery, zmienne z małej.
- Funkcja `show` wypisuje na standardowe wyjście.

- Słowo kluczowe `step` dostępne w pętlach.

14.Rezerwa nazw

Zastrzeżone słowa kluczowe:

```
define, blueprint, delivers, Core, show, check, then,
otherwise,
repeat, until, for, in, box, chain, catalog, int, float, bool,
char, string,
true, false, step, request
```

Nie mogą być używane jako nazwy zmiennych/funkcji.