|  |
| --- |
| TKOM |
| Interpreter języka do operacji na walutach |
| Dokumentacja końcowa |

|  |
| --- |
| Grzegorz Aleksiuk  2020-05-08 |

Spis treści

[Opis funkcjonalny 2](#_Toc40033733)

[Instrukcja obsługi 2](#_Toc40033734)

[Szczegóły dotyczące języka 2](#_Toc40033735)

[Przykład języka 5](#_Toc40033736)

[Formalny opis gramatyki 6](#_Toc40033737)

[Opis techniczny realizacji 7](#_Toc40033738)

# Opis funkcjonalny

Projekt ma na celu stworzenie interpretera prostego języka, który ułatwi wykonywanie operacji na walutach dzięki wbudowanym typom walutowym (tj. EUR, USD, CHF, GBP, JPY, PLN)

**Język ma umożliwiać:**

* definiowanie własnych funkcji
* deklarowanie zmiennych walutowych
* wykonywanie działań na zmiennych walutowych (tj. dodawanie, odejmowanie, mnożenie przez liczbę rzeczywistą)
* konwersję z jednej waluty na drugą

# Instrukcja obsługi

Aby zbudować projekt należy przy użyciu narzędzia Maven w konsoli wpisać komendę:

*mvn clean install*

Do uruchomienia interpretera konieczny jest zainstalowany Java Runtime Environment. Interpreter uruchamia się poleceniem.

*java -jar tkom-currency-interpreter <plik z programem> <plik z walutami>*

Pliki należy podawać wraz ze ścieżką a także rozszerzeniem.

# Szczegóły dotyczące języka

W języku zaimplementowany jest typ całkowity (int), typ zmiennoprzecinkowy (double) a także typy zdefiniowane w pliku z walutami, wraz z ich przelicznikami na Euro

Plik źródłowy musi zawierać jedną bezargumentową funkcję „main” zwracającą int, jest ona funkcją od której zaczyna się wykonywanie programu. Wszystkie funkcje w tym języku muszą zwracać wartość przy użyciu return. Zmienne przekazywane są do funkcji poprzez wartość. Interpreter nie umożliwia deklarowania zmiennych globalnych.

W język wbudowana jest funkcja print() która umożliwia wypisanie zmiennych, liczb całkowitych, liczb zmiennoprzecinkowych, walut a także łańcuchów znaków. Po przecinku podawane są argumenty do wypisania. Przykład użycia:

*print(”test ”, zmienna);*

„Główną” walutą jest EUR, na niej wykonywane są wszystkie konwersje, dlatego jest wymagane aby była zdefiniowana w pliku z walutami jako „EUR”. A wszystkie pozostałe waluty miały podane przeliczniki z Euro. Deklarowanie zmiennych walutowych wygląda następująco:

*EUR zmienna = 12.5;*

Podczas deklaracji zmiennej walutowej możemy do niej przypisać inta, doubla lub inną walutę.

Instrukcje if, else oraz while mogą składać się z bloku lub pojedynczej instrukcji.

Niedozwolone jest przypisanie do zmiennej wartości o innym typie niż typ podanych podczas deklaracji zmiennej (wyjątkiem jest przypisanie inta do doubla a także waluty w jednym typie do zmiennej w walucie innego typu).

**Operacje arytmetyczne:**

* możliwe jest dzielenie i mnożenie waluty przez doubla i inta
* dodawanie walut do inta i doubla jest niedozwolone
* dzielenie inta, doubla i waluty przez walutę jest niedozwolone

**Operacje logiczne:**

* dwie waluty są równe jeśli ich wartości są równe (po przeliczeniu na EUR)
* nie można porównywać waluty do inta lub doubla.
* wynikiem operacji logicznej jest typ Bool

**Tokeny:**



A także własnoręcznie zdefiniowane waluty w pliku wejściowym o przykładowej strukturze:



# Przykład języka



# Formalny opis gramatyki



# Opis techniczny realizacji

Projekt jest zrealizowany w języku Java z wykorzystaniem narzędzia Maven.

Do testów jednostkowych została biblioteka JUnit.

Program jest uruchamiany z poziomu konsoli wraz z argumentami do wczytania pliku z programem napisanym w tworzonym języku oraz pliku z walutami.