1. Обзор языка ............................................................................................................................................................... 14

1.1. Начнем, пожалуй ................................................................................................................................................ 14

1.2. Переменные и арифметические выражения ................................................................................................... 16

1.3. Инструкция for .................................................................................................................................................... 21

1.4. Именованные константы ................................................................................................................................... 22

1.5. Ввод-вывод символов ........................................................................................................................................ 23

1.5.1. Копирование файла .................................................................................................................................... 23

1.5.2. Подсчет символов ....................................................................................................................................... 25

1.5.3. Подсчет строк .............................................................................................................................................. 26

1.5.4. Подсчет слов ................................................................................................................................................ 27

1.6. Массивы .............................................................................................................................................................. 29

1.7. Функции .............................................................................................................................................................. 31

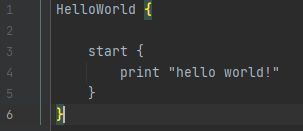
1.8. Аргументы. Вызов по значению ........................................................................................................................ 33

1.9. Символьные массивы ........................................................................................................................................ 34

1.10. Внешние переменные и область видимости ................................................................................................. 37

1. **Обзор языка**
   1. **Начнем, пожалуй**

WLang-класс, печатающая "hello world!", выглядит так:



В папке out\artifacts\the\_compiler.jar находится собранный проект the\_compiler.jar.

Пример команды для запуска: java -jar the\_compiler.jar D:\SumCalculator.w

По завершении выполнения программы мы получаем скомпилированный класс, который можно запустить по средствам JVM.

Пример команды для запуска скомпилированного класса: java SumCalculator

Если вы все сделали правильно, то программа напечатает

hello world!

Методы в WLang похожи на методы в Java.

Приведенная программа — это класс с методом start. Обычно вы вольны придумывать любые имена для своих методов, но "start" — особое имя: любой класс в WLang начинает свои вычисления с первой команды метода start.

Первая программа на WLang.

|  |  |
| --- | --- |
| HelloWorld { | *Название класса, его тело заключается в фигурные скобки* |
| start | *Определение метода с именем start* |
| { | *Метод start(как и любая другая) заключаются в фигурные скобки.* |
| print “hello world!” | *Метод start вызывает библиотечную функцию*  print *для печати заданной последовательности*  *символов;* |
| }  } | *Все блоки кода начинаются и заканчиваются фигурными скобками* |

Команды метода заключаются в фигурные скобки {}. Метод start содержит только одну команду

print "hello world!”

Функция вызывается по имени, после которого, перечисляются аргументы. Таким образом,

приведенная выше строка — это вызов функции print с аргументом "hello world!”. Функция

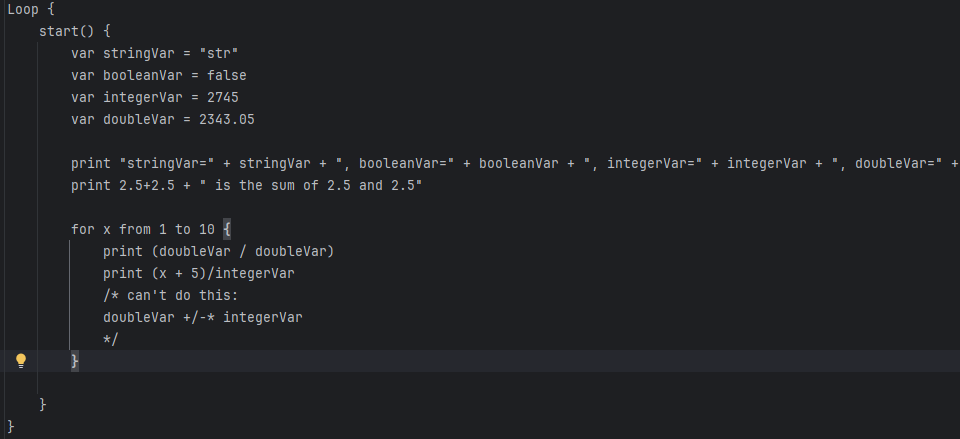
print — это библиотечная функция, которая в данном случае напечатает последовательность символов,

заключенную в двойные кавычки.

Последовательность символов в двойных кавычках, такая как "hello world!”, называется строкой символов, или строковой константой.

* 1. Переменные и арифметические выражения

Приведенная ниже программа четкого смысла не имеет, она лишь показывает, как объявляются переменный и арифметические возможности Wlang.



Любая переменная объявляется раньше, чем используется, даже если она является итерационной в цикле for. Для объявления обычной переменной в начале указывается ключевое слово “var”, например:

var stringVar = “str”

Тип переменной определяется автоматически в зависимости от ее значения.

Вся основная арифметика выполняется в цикле for:

for x from 1 to 10 {

Он работает следующим образом, после ключевого слова for пишется название итерационной переменной, после идут начальная граница (from) и конечная граница(to) в которых эта переменная будет итерироваться с шагом 1. Итерационную переменную можно использовать в теле цикла как обычную, но за пределами тела цикла ее существовать не будет.

Две строки в теле цикла for:

/\* can't do this:  
doubleVar +/-\* integerVar  
\*/

Являются комментариями, которые в данном случае, кратко объясняют, что арифметические операции между разными типами (за исключением сложения со строками) переменных в WLang не работают. Все символы, помещенные между /\* и \*/, игнорируются компилятором, и этим можно свободно пользоваться, чтобы сделать программу более понятной.

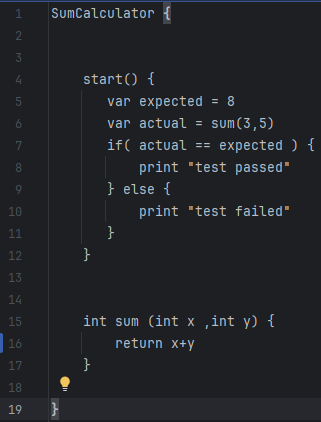
Две строки перед циклом for:

print "stringVar=" + stringVar + ", booleanVar=" + booleanVar + ", integerVar=" + integerVar + ", doubleVar=" + doubleVar  
print 2.5+2.5 + " is the sum of 2.5 and 2.5"

демонстрируют возможность сложения любого типа данных со строкой, результат такой операции всегда строка.

1.3 Функции

Итак, мы имеем функцию sum и главную функцию start, пользующуюся ее услугами, так что вся программа выглядит следующим образом:



Определение любой функции имеет следующий вид:

тип-результата имя-функции (список параметров, если он есть)

{

Тело функции

}

В 6 строке программы происходит вызов функции sum, главный метод start в ответ получает число и в данном случае сохраняет в переменную actual.

В первой строке определения sum:

int sum (int x ,int y)

указываются типы параметров, имя функции и тип результата. Имена параметров локальны внутри sum, это значит, что они скрыты для любой другой функции, так что остальные подпрограммы могут свободно пользоваться теми же именами для своих целей.

Значение, вычисляемое функцией sum, возвращается в start с помощью инструкции return. За словом return может следовать любое выражение:

return выражение

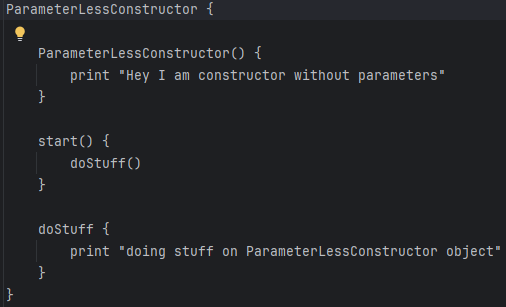
Функция не обязательно возвращает какое-нибудь значение. Инструкция return без выражения только передает управление в ту программу, которая ее вызвала, не передавая ей никакого результирующего значения. То же самое происходит, если в процессе вычислений мы выходим на конец функции, обозначенный в тексте последней закрывающей фигурной скобкой.

* 1. Конструкторы

В объектно-ориентированной архитектуре Java у классов присутствуют конструкторы для возможности использования функционала одного класса в другом, WLang тоже поддерживает такую возможность

Для начала разберем как объявлять конструкторы, а уже потом взаимодействия между классами.

Вот пример класса с конструктором без параметров:

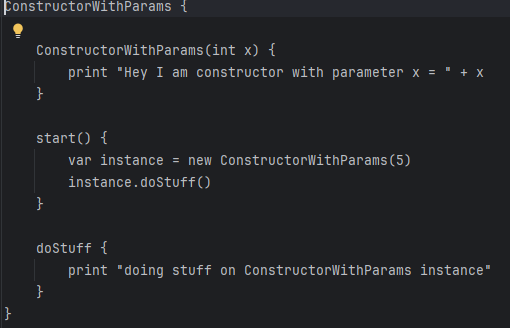


Конструктор отличается от метода несколькими правилами:

-название конструктора совпадает с название класса

-возвращаемый тип всегда null

Так как конструктор — это специфичный метод, то в него могут передаваться параметры:



Вызов конструктора в WLang похож на вызов конструктора в Java, сначала пишется ключевое слово “new” затем название конструктора и круглые скобки, куда передаются аргументы.

Конструктор возвращает экземпляр класса, и чтобы им воспользоваться, его можно, например сохранить в переменную, после обратиться к методу этого класса:

var instance = new ConstructorWithParams(5)  
instance.doStuff()

Как и в java если класс не имеет явно прописанного конструктора, то будет создан конструктор по умолчанию без параметров.