**Министерство образования и науки Российской Федерации**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Владимирский государственный университет имени Александра Григорьевича

и Николая Григорьевича Столетовых» (ВлГУ)

Институт информационных технологий и электроники

Кафедра информатики и защиты информации

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к программе**

**«Компилятор»**

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ к.т.н. доцент кафедры ИЗИ

подпись, дата Ю.М. Монахов

Исполнитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ студент группы ИБ-122

подпись, дата Д.С. Чугров

Владимир 2025

Оглавление

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | |  |  |  |  | Стр. |
| 1. Краткое описание программы | | | | | | | 3 |
| 2. Описание модулей | | | | | | | 4 |
|  | | 2.1. Лексический анализатор | | | | | 4 |
|  | | 2.2. Синтаксический анализатор | | | | | 4 |
|  | | 2.3. Семантический анализатор | | | | | 5 |
|  | | 2.4. Оптимизатор | | | | | 6 |
|  | | 2.5. Транслятор в целевой код | | | | | 6 |
| 3. IDE | | | | | | | 6 |

1. Краткое описание программы

Компилятор — это программа, которая переводит исходный код, написанный на высокоуровневом языке программирования, в машинный код или промежуточный код. Данный процесс называется компиляцией.

Трансляция исходного языка производится в эквивалентный целевой язык - байт-код. Соответственно, результатом работы компилятора - программа на специально созданном низкоуровневом языке двоично-кодовых команд, выполняемых виртуальной машиной JVM.

За основу исходного языка взят объектно- ориентированный язык программирования Oxygene с некоторыми изменениями.

Пример программы:

namespace ConsoleApplication;

interface

Program = class

public

class method main();

class method getNumber:Integer;

end;

implementation

class method Program.getNumber:Integer;

begin

int result = 2025;

return result;

end;

class method Program.Main;

begin

print(“Hello “ + Program.getNumber);

end;

end.

Структура программы

Компилятор состоит из следующих частей:

* Лексический анализатор
* Синтаксический анализатор
* Семантический анализатор
* Оптимизатор
* Транслятор в целевой код

2. Описание модулей

2.1. Лексический анализатор

Лексический анализатор - первая стадия компиляции. Он выполняет обработку исходного кода программы и разбивает его на лексемы: ключевые слова, идентификаторы, литералы, операторы и разделители.

Лексический анализатор реализован с помощью генераторов парсеров ANTLR. Соответственно грамматика языка написана в файле Tokens.g4. На ее основе сгенерированы лексер и парсер, необходимые для остальных частей компилятора, и находятся в папке gen.

grammar Tokens;  
  
@header {  
package ru.Compiler;  
}  
*// Основная структура*program  
 : namespaceDeclaration? interfaceDeclaration implementationDeclaration END\_P ;  
  
namespaceDeclaration  
 : NAMESPACE ID\_UP PC;  
  
interfaceDeclaration  
 : INTERFACE classDeclaration\*;  
  
classDeclaration  
 : ID\_UP EQUALLY CLASS classBody END\_M;

Листинг. 1. Часть грамматики языка Token.g4

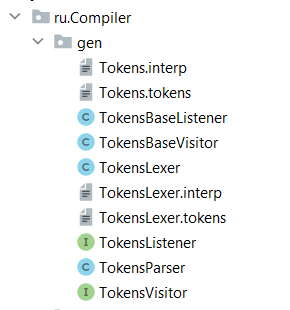


Рис. 1. Содержимое папки gen, необходимое для лексера и парсера

2.2. Синтаксический анализатор

Синтаксический анализатор - следующая часть компилятора. Он принимает поток токенов от лексического анализатора для проверки на соответствие синтаксису языка программирования.

Реализовано с помощью сгенерированных файлов на основе грамматики.

2.3. Семантический анализатор

Семантический анализатор - третья стадия компиляции. Он отвечает за проверку смысловой корректности программы в рамках правил языка.

На этом этапе осуществляется следующие:

- Проверка типов данных;

- Проверка объявления переменных;

- Проверка совместимости;

- Формируется таблица символов и область их видимости.

Результатом семантического анализатора является трехадресный код.

Namespace: ConsoleApplication

Interface

Program = class

Program.main(): void

Program.getNumber(int a): Integer

Implementation

Program.getNumber(int a): Integer

begin

t0 = 1

t1 = a

t2 = t0 + t1

f = t2

return f

end;

Program.main: void

begin

t3 = 5

push t3

t4 = call Program.getNumber(t3)

result = t4

t5 = result

print(t5)

end;

end.

Листинг 2. Трёхадресный код

Семантическая часть компилятора находится в папке Semantic и состоит из следующих java - классов:

1. SemanticAnalyzer - необходим для построения и обхода AST - дерева;

2. ClassInfo - собирает информацию классах и методах в них;

3. MethodInfo - хранит информацию о методе класса, его типе, его модификаторе доступа и параметрах

4. ParameterInfo - собирает информацию о параметрах и переменных в методе

5. TypedValue - вспомогательный класс, необходимый для корректной работы команды print();

6. ErrorListener - используется для сбора семантических ошибок в процессе компиляции;

7. CodeGenerator - класс, необходимый для построения трёхадресного кода.

2.4. Оптимизатор

Оптимизатор компилятора - следующая часть компилятора. Он преобразует трёхадресный код таким образом, что позволяет улучшить его работу: повысить производительность, снизить использование памяти и ресурсов и т.д.

Оптимизатор реализован с помощью класса CodeOptimizer и способен на следующие:

1. Удаление присваиваний вида x = x;

2. Свертка констант;

3. Удаление мертвого кода.

2.5. Транслятор в целевой код

Трансляция в целевой код - заключительный этап компиляции. На нем происходит преобразование промежуточного представления программы (трёхадресного кода) в низкоуровневый код целевой платформы (байт-код JVM). Итогом является файл .class, который можно запустить с помощью JDK не позднее 17 версии.

3. IDE

IDE - инструмент для редактирования кода, компиляции и отладки.

Реализация представляет собой класс IDE, использующий библиотеку Swing для создания приложений с графическим интерфейсом. Поставляется в виде .exe файла.

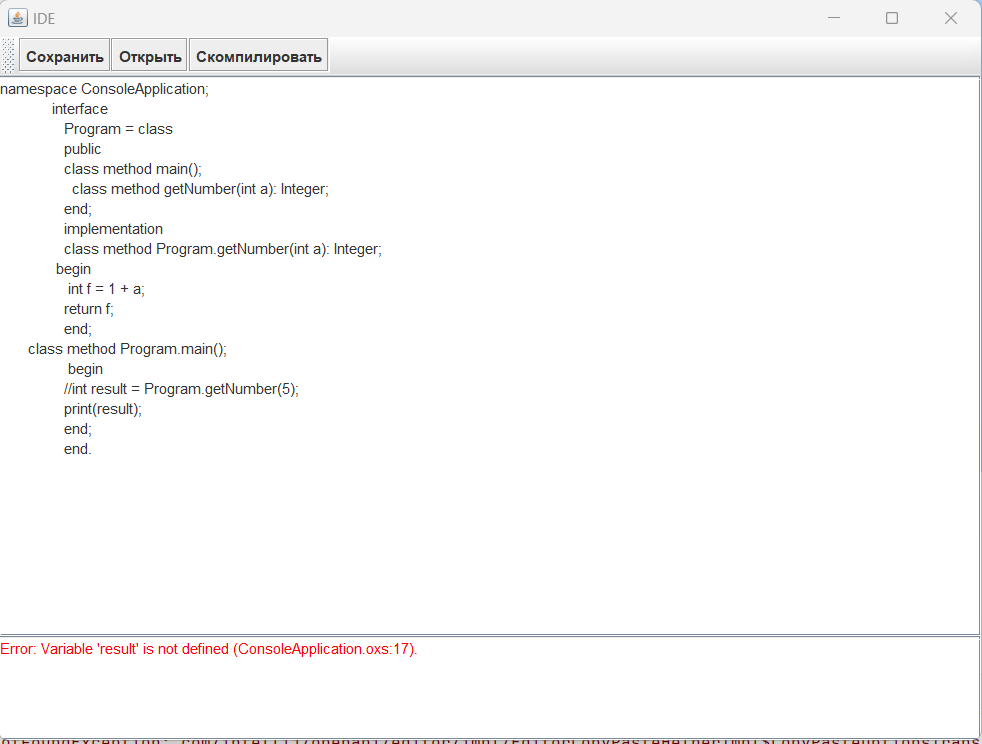


Рис. 2. IDE

Функции:

1. Сохранения текущего кода в файл .oxs;

2. Открытие существующих файлов;

3. Компиляция кода;

4. Вывод сообщений об успехе компиляции или ошибок в процессе компиляции.