

Лабораторная работа №9

«Форматтер исходных текстов»

Скоробогатов С. Ю., Коновалов А. В.

10 мая 2016

1 Цель работы

Целью данной работы является приобретение навыков использования генератора синтаксических анализаторов bison.

2 Исходные данные

Форматтер исходных текстов — это инструментальное средство, выполняющее вставку пробельных символов в исходные тексты программ, записанных на некотором языке, для улучшения удобочитаемости этих исходных текстов.

Форматтеры бывают *сильными* и *слабыми*. Слабый форматтер, в отличие от сильного, не может вставлять в исходный текст программы переводы строки.

Например, рассмотрим неотформатированную программу на языке Pascal:

```
var i:integer;
a,b,c:string
begin
a:='a';b:='b';
for i:=1 to 10 do begin
Println(a);
c:=a;a:=b;b:=c+b
end
end.
```

Слабый форматтер расставит в программе отступы и пробелы между лексемами:

```
var i: integer;
  a, b, c: string
begin
  a := 'a'; b := 'b';
  for i := 1 to 10 do begin
    Println(a);
    c := a; a := b; b := c+b
  end
end.
```

Сильный форматтер добавит переводы строки после ключевых слов **var** и **begin** и, кроме того, разобьёт строки, содержащие сразу несколько операторов:

```
var
  i: integer;
  a, b, c: string
begin
  a := 'a';
  b := 'b';
  for i := 1 to 10 do
    begin
      Println(a);
      c := a;
      a := b;
      b := c+b
    end
  end.
```

3 Задание

В данной лабораторной работе требуется разработать слабый или сильный форматтер исходных текстов для одного из следующих языков:

1. **var**-блок языка Pascal;
2. составной оператор языка Java (без вложенных анонимных классов, объявлений переменных обобщённых типов, **break**'ов на метку и оператора **switch**);
3. объявление класса в языке Java (со вложенными классами, тела методов — пустые);
4. объявление структуры в языке C (со вложенными структурами и объединениями);
5. XML;
6. JSON (язык создания объектов JavaScript);
7. определение функции в языке Scheme;
8. определение процедуры в языке Visual Basic;
9. Bison (в семантических действиях могут быть присваивания, вызовы функций и вложенные фигурные скобки);
10. РЕФАЛ-5;
11. Go (с опциональными точками с запятой);
12. операторы **if**, **for**, **while** в языке Rust;
13. объявление класса в языке C++98 (со вложенными классами, тела методов — пустые);

14. циклы **for**, **while** в языке C, оператор **switch** в языке C;
15. объявление класса в языке JavaScript (со вложенными классами, тела методов — выражения **return**);
16. объявление структуры в языке Rust (со вложенными структурами и объединениями);
17. РБНФ;
18. Flex (в семантических действиях могут быть присваивания, вызовы функций и вложенные фигурные скобки);
19. операторы **if**, **for**, **while** в языке Pascal;
20. язык Dot (только ориентированный граф со вложенными подграфами, из атрибутов доступны только **rankdir**, **label** и **color**).

Выполнение лабораторной работы состоит из следующих этапов:

1. Создание лексического анализатора с помощью flex.
2. Разработка синтаксического анализатора с помощью bison и соединение его с лексическим анализатором.
3. Вставка в синтаксический анализатор семантических действий, выполняющих порождение отформатированного исходного текста программы.
4. Тестирование работоспособности получившегося форматтера на наборе примеров.

Форматтер должен работать до обнаружения первой синтаксической ошибки в тексте программы.

Слабый форматтер должен сохранять комментарии в тексте программы.