Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ имени Н.Э.БАУМАНА
(национальный исследовательский университет)»

Факультет: Информатика и системы управления

Кафедра: Теоретическая информатика и компьютерные технологии

Лабораторная работа № 9

Раскрутка самоприменимого компилятора

Вариант 3

Работу выполнил

студент группы ИУ9-61

Бакланова А.Д.

Москва, 2020

Цель работы:

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

Исходные данные:

Форматтер исходных текстов — это инструментальное средство, выполняющее вставку пробельных символов в исходные тескты 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:

Задание:

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

Индивидуальный вариант:

3. объявление класса в языке Java (со вложенными классами, тела методов — пустые);

Реализация:

```
46  Start:      Object
47              { printf("%s\n",$1); };
48
49  Object:     prefix CLASS IDENT LBRACE add_indent body
50              { $$ =  MEM(strlen($1)+strlen($3)+strlen($6)+30); sprintf($$, "%s class %s { %s", $1, $3,
                $6);};
51              | CLASS IDENT LBRACE add_indent body
52              { $$ =  MEM(strlen($2)+strlen($5)+30); sprintf($$, "class %s { %s}", $2, $5);};
53
54
55  body:       class body
56              { $$ = MEM(strlen($1)+strlen($2)+30); sprintf($$, "\n%s \n%s", $1, $2); };
57              |method body
58              { $$ = MEM(strlen($1)+strlen($2)+30); sprintf($$, "\n%s \n%s", $1, $2); };
59              | RBRACE
60              { env[0] = env[0]-5; char *sp = make_tabs(env[0]);  $$ = MEM(10); sprintf($$, "\n\n%s}",
                sp);free(sp);};
61
62
63  class:      prefix CLASS IDENT LBRACE body
64              { char *sp = make_tabs(env[0]);  $$ =  MEM(strlen($1)+strlen($3)+strlen($5)+30); sprintf($$,
                "\n%s%s class %s { %s", sp, $1, $3, $5);free(sp);};
65              | CLASS IDENT LBRACE body
66              { char *sp = make_tabs(env[0]); $$ =  MEM(strlen($2)+strlen($4)+30); sprintf($$, "%sclass %s
                { %s",sp,  $2, $4); free(sp);};
67
68
69  method:     prefix IDENT LBRACKET RBRACKET LBRACE RBRACE
70              { char *sp = make_tabs(env[0]-10); $$ =  MEM(strlen($1)+strlen($2)+30); sprintf($$, "%s%s
                %s(){\n\n%s}",sp, $1, $2, sp);free(sp);};
71
72              | IDENT LBRACKET RBRACKET LBRACE RBRACE
73              { char *sp = make_tabs(env[0]-10); $$ =  MEM(strlen($1)+30); sprintf($$, "%s(){\n\n%s}", $1,
                sp); free(sp);};
74
75
76  prefix:     PUBLIC
77              { $$ =  MEM(30); sprintf($$, "public");};
78              | PRIVATE
79              { $$ =  MEM(30); sprintf($$, "private");};
80  add_indent:                             {env[0]=env[0]+5; }
81      ;
82  remove_indent:                          {   env[0]=env[0]-5;  }
83      ;
84
```

```
44  DIGIT    [0-9]
45  LETTER   [a-zA-Z]
46  IDENT    {LETTER}({LETTER}|{DIGIT})*
47  NUMBER   {DIGIT}+
48
49  %%
50
51  [\n\t ]+
52
53  class       return CLASS;
54  public      return PUBLIC;
55  private     return PRIVATE;
56
57
58  {IDENT}    {
59                  yylval->ident = strdup(yytext);
60                  return IDENT;
61              }
62
63
64  {NUMBER}    {
65                 yylval->number = atoi(yytext);
66                  return NUMBER;
67              }
68
69
70  \{      return LBRACE;
71  \}      return RBRACE;
72
73  \(      return LBRACKET;
74  \)      return RBRACKET;
75
76
77  ;       return SEMICOLON;
78  ,       return COMMA;
79
```

Тестирование:

```
*************** input ***************
public class A{

public metod(){}

public class B{class C{}}


}

*************** output ***************

public class A {
     public metod(){

     }


     public class B {
          class C {

          }


     }


}
```

Вывод:

В результате выполнения лабораторной работы были приобретены навыки использования генератора синтаксических анализаторов bison.