Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

Кафедра информатики и прикладной математики

Основы разработки компиляторов

Лабораторная работа 2

"Разработка синтаксического анализатора"

Вариант 7

Выполнил: Шкаруба Н.Е.

Группа: P3318

2016 г

Цель работы

Разработать синтаксический анализатор.

Программа

```
public class SyntaxAnalyzer {
  private static List<List<Symbol>> grammar = new ArrayList<>();
  private static List<Map<Integer, Integer>> table = new ArrayList<>();
  static {grammar.addAll(Arrays.asList(
          new ArrayList<Symbol>(),
          new ArrayList<Symbol>(Arrays.asList(
                new Symbol(false, 2, 0),
                new Symbol(false, 3, 0))),
          new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 103, 0),
                new Symbol(false, 5, 0))),
          new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 101, 0),
                new Symbol(false, 4, 0),
                new Symbol(true, 102, 0))),
          new ArrayList<Symbol>(Arrays.asList(
                new Symbol(false, 6, 0),
                new Symbol(false, 20, 0))),
          new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 104, 0),
                new Symbol(false, 21, 0))),
          new ArrayList<Symbol>(Collections.singletonList(
                new Symbol(false, 7, 0))),
          new ArrayList<Symbol>(Collections.singletonList(
                new Symbol(false, 8, 0))),
          new ArrayList<Symbol>(Collections.singletonList(
                new Symbol(false, 9, 0))),
          new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 104, 0),
                new Symbol(true, 108, 0),
                new Symbol(false, 10, 0),
                new Symbol(true, 105, 0))),
          new ArrayList<Symbol>(Collections.singletonList(
                new Symbol(false, 16, 0))),
          new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 101, 0),
                new Symbol(false, 4, 0),
                new Symbol(true, 107, 0))),
          new ArrayList<Symbol>(Arrays.asList(
                new Symbol(false, 11, 0),
                new Symbol(false, 12, 0))),
          new ArrayList<Symbol>(Collections.singletonList(
                new Symbol(false, 12, 0))),
          new ArrayList<Symbol>(Collections.singletonList(
                new Symbol(true, 110, 0))),
          new ArrayList<Symbol>(Arrays.asList(
                new Symbol(false, 13, 0),
                new Symbol(false, 17, 0))),
          new ArrayList<Symbol>(Arrays.asList(
                new Symbol(false, 14, 0),
                new Symbol(false, 18, 0))),
          new ArrayList<Symbol>(Collections.singletonList(
                new Symbol(false, 15, 0))),
          new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 118, 0),
                new Symbol(false, 10, 0),
                new Symbol(true, 119, 0))),
          new ArrayList<Symbol>(Collections.singletonList(
                new Symbol(true, 104, 0))),
```

```
        new ArrayList<Symbol>(Collections.singletonList(
                new Symbol(true, 120, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 121, 0),
                new Symbol(false, 10, 0),
                new Symbol(true, 122, 0),
                new Symbol(false, 6, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 109, 0),
                new Symbol(false, 13, 0),
                new Symbol(false, 17, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 110, 0),
                new Symbol(false, 13, 0),
                new Symbol(false, 17, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 124, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 111, 0),
                new Symbol(false, 14, 0),
                new Symbol(false, 18, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 112, 0),
                new Symbol(false, 14, 0),
                new Symbol(false, 18, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 113, 0),
                new Symbol(false, 14, 0),
                new Symbol(false, 18, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 114, 0),
                new Symbol(false, 14, 0),
                new Symbol(false, 18, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 115, 0),
                new Symbol(false, 14, 0),
                new Symbol(false, 18, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 116, 0),
                new Symbol(false, 14, 0),
                new Symbol(false, 18, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 117, 0),
                new Symbol(false, 14, 0),
                new Symbol(false, 18, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 124, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 124, 0))),
        new ArrayList<Symbol>(Collections.singletonList(
                new Symbol(false, 5, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 124, 0))),
        new ArrayList<Symbol>(Collections.singletonList(
                new Symbol(false, 4, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 106, 0),
                new Symbol(false, 5, 0))),
        new ArrayList<Symbol>(Arrays.asList(
                new Symbol(true, 105, 0),
                new Symbol(false, 19, 0)))));
    table.addAll(
        Arrays.asList(
                new HashMap<Integer, Integer>(),
                new HashMap<Integer, Integer>() {{
                  put(103, 1); }},
                new HashMap<Integer, Integer>() {{
                  put(103, 2); }},
                new HashMap<Integer, Integer>() {{
                  put(101, 3); }},
                new HashMap<Integer, Integer>() {{
                  put(101, 4);
                  put(104, 4);
```

```
                put(121, 4); }},
            new HashMap<Integer, Integer>() {{
                put(104, 5); }},
            new HashMap<Integer, Integer>() {{
                put(101, 8);
                put(104, 6);
                put(121, 7); }},
            new HashMap<Integer, Integer>() {{
                put(104, 9); }},
            new HashMap<Integer, Integer>() {{
                put(121, 10); }},
            new HashMap<Integer, Integer>() {{
                put(101, 11); }},
            new HashMap<Integer, Integer>() {{
                put(104, 13);
                put(110, 12);
                put(118, 13);
                put(120, 13); }},
            new HashMap<Integer, Integer>() {{
                put(110, 14); }},
            new HashMap<Integer, Integer>() {{
                put(104, 15);
                put(118, 15);
                put(120, 15); }},
            new HashMap<Integer, Integer>() {{
                put(104, 16);
                put(118, 16);
                put(120, 16); }},
            new HashMap<Integer, Integer>() {{
                put(104, 17);
                put(118, 18);
                put(120, 17); }},
            new HashMap<Integer, Integer>() {{
                put(104, 19);
                put(120, 20); }},
            new HashMap<Integer, Integer>() {{
                put(121, 21); }},
            new HashMap<Integer, Integer>() {{
                put(105, 24);
                put(109, 22);
                put(110, 23);
                put(119, 24);
                put(122, 24); }},
            new HashMap<Integer, Integer>() {{
                put(105, 32);
                put(109, 32);
                put(110, 32);
                put(111, 25);
                put(112, 26);
                put(113, 27);
                put(114, 28);
                put(115, 29);
                put(116, 30);
                put(117, 31);
                put(119, 32);
                put(122, 32); }},
            new HashMap<Integer, Integer>() {{
                put(101, 33);
                put(104, 34); }},
            new HashMap<Integer, Integer>() {{
                put(101, 36);
                put(102, 35);
                put(104, 36);
                put(107, 35);
                put(121, 36); }},
            new HashMap<Integer, Integer>() {{
                put(105, 38);
                put(106, 37); }})); }
public static String analyze(List<Symbol> symbols) {
    Symbol program = new Symbol(false, 1, 0);
    Deque<Symbol> stack = new ArrayDeque<>();
    stack.push(program);
    int head;
```

```
    int i = 0;
    while (i < symbols.size()) {
       Symbol popped = stack.pop();
       Symbol terminal = symbols.get(i);
       head = popped.getId();
       System.out.println(head);
       System.out.println(popped);
       System.out.println(terminal);
       Integer ruleID = table.get(head).get(terminal.getId());
       if (ruleID == null) {
          return "Program is incorrect, error at line: " + terminal.getLine();}
       head = ruleID;
       List<Symbol> rule = grammar.get(head);
       for (int j = rule.size() - 1; j >= 0; j--) {
          if (rule.get(j).getId() != 124) {
             stack.push(rule.get(j)); }}
       while (stack.peek() != null && stack.peek().isTerminal()) {
/             stack.pop();
          System.out.println("SKIPPED! " + symbols.get(i));
          System.out.println("POP! " + stack.pop());
          i++;}
       System.out.println("End: " + stack.peek());
       System.out.println(stack); }
    if (stack.size() == 0) {
       return "Program is correct";}
    return "Program is correct";}}
```