<программа> → void main <блок>

<блок> → <начало блока>{<предложение>;}<конец блока>

<начало блока> → {

<конец блока> →}

<предложение> → <описание переменной>|<оператор присваивания>|<оператор ввода>|<оператор вывода>|<оператор if>|<оператор for>|<оператор while>

<описание переменной> → <тип данных><идентификатор>[=<выражение>]

<тип данных> → int | float | string

<идентификатор> → <символ>{<символ>|<цифра>}

<символ> → a-z|A-Z|\_

<цифра> → 0|1|2|3|4|5|6|7|8|9

<выражение> → <арифметическое выражение>|<константа>

<константа> → <строка> | <число>

<оператор присваивания> → <идентификатор> = <выражение>

<оператор ввода> → cin >> <идентификатор>

<оператор вывода> → cout << <выражение> {<< <выражение>}

<оператор if> → if (<условное выражение>) <блок> [else <блок>]

<оператор for> → for (<описание переменной>;<условное выражение>;<арифметическое выражение>) <блок>

<оператор while> → while (<условное выражение>) <блок>

<условное выражение> → <сложное условное выражение>{<операция сравнения> <сложное условное выражение>}

<сложное условное выражение> → (<условное выражение>)|<простое условное выражение>

<простое условное выражение> → <арифметическое выражение><знак сравнения> <арифметическое выражение>

<операция сравнения> → && | || | !

<знак сравнения> → > | < | = | >= | <= | !=

<арифметическое выражение> → <слагаемое>{<операция типа сложение><слагаемое>}

<слагаемое> → <множитель>{<операция типа умножение><множитель>}

<множитель> → (<арифметическое выражение>)|<число>|<идентификатор>

<операция типа сложение> → +|-

<операция типа умножение> → \*|/

<программа> → void main <блок>

<блок> → <начало блока> <предложение> <конец блока>

<начало блока> → {

<конец блока> →}

<предложение> → <описание переменной> <предложение>|

<оператор присваивания> <предложение>|

<оператор ввода> <предложение>|

<оператор вывода> <предложение>|

<оператор if> <предложение>|

<оператор for> <предложение>|

<оператор while> <предложение>

<описание переменной> → <тип данных><идентификатор> EX;

EX → =<выражение> | λ

<тип данных> → int | float | string

<выражение> → <арифметическое выражение>|<константа>

<константа> → <строка> | <число>

<оператор присваивания> → <идентификатор> = <выражение>;

<оператор ввода> → cin >> <идентификатор>

<оператор вывода> → cout <функция вывода> ;

<функция вывода> → << <выражение> | << <выражение> <функция вывода>

<оператор if> → if (<условное выражение>) <блок> EL

E → else <блок> | λ

<оператор for> → for (<описание переменной>;<условное выражение>;<арифметическое выражение>) <блок>

<оператор while> → while (<условное выражение>) <блок>

<условное выражение> → <сложное условное выражение> <второе сложное условное выражение>

<второе сложное условное выражение> → <операция сравнения> <сложное условное выражение><второе сложное условное выражение> | λ

<сложное условное выражение> → (<условное выражение>)|<простое условное выражение>

<простое условное выражение> → <арифметическое выражение><знак сравнения> <арифметическое выражение>

<арифметическое выражение> → <слагаемое> <второе слагаемое>

<второе слагаемое> → <операция типа сложение> <слагаемое><второе слагаемое> | λ

<слагаемое> → <множитель> <второй множитель>

<второй множитель> → <операция типа умножение> <множитель> <второй множитель> | λ

<множитель> → (<арифметическое выражение>)|<число>|<идентификатор>

1. Prog → main Block
2. Block → { Sent }
3. Sent → DescrVar Sent
4. Sent → OpEqu Sent
5. Sent → OpIn Sent
6. Sent → OpOut Sent
7. Sent → OpIf Sent
8. Sent → OpFor Sent
9. Sent → OpWhile Sent
10. Sent → λ
11. DescrVar → type id EquExpr ;
12. EquExpr → = Expr
13. EquExpr → λ
14. Expr → ArExpr
15. Expr → conststr
16. OpEqu → id = Expr;
17. OpIn → cin >> id ;
18. OpOut → cout << Expr FuncOut ;
19. FuncOut → λ
20. FuncOut → << Expr FuncOut
21. OpIf → if (CondExpr ) Block Else
22. Else → else Block
23. Else → λ
24. OpFor → for ( OpEqu ; CondExpr; ArExpr) Block
25. OpWhile → while (CondExpr) Block
26. CondExpr→ DiffCondExpr SecDCE
27. SecDCE → compop DiffCondExpr SecDCE
28. SecDCE → λ
29. DiffCondExpr→ (CondExpr)
30. DiffCondExpr→ SimpCondExpr
31. SimpCondExpr → id relop ArExpr
32. SimpCondExpr → constnum relop ArExpr
33. ArExpr→ Addendum SecAddendum
34. SecAddendum → addop Addendum SecAddendum
35. SecAddendum → λ
36. Addendum → Multiplier SecMultiplier
37. SecMultiplier → multop Multiplier SecMultiplier
38. SecMultiplier → λ
39. Multiplier → ( ArExpr)
40. Multiplier → constnum
41. Multiplier → id

<программа> → main <блок>

<блок> → { <предложение> }

<предложение> → <описание переменной> <предложение> |

<оператор присваивания> <предложение> |

<оператор ввода> <предложение> |

<оператор вывода> <предложение> |

<оператор if> <предложение> |

<оператор for> <предложение> |

<оператор while> <предложение>

<описание переменной> → type id EX ;

EX → = <выражение> | λ

<выражение> → <арифметическое выражение> | <константа>

<константа> → conststr | constnum

<оператор присваивания> → id = <выражение> ;

<оператор ввода> → cin >> id ;

<оператор вывода> → cout <функция вывода> ;

<функция вывода> → << <выражение> | << <выражение> <функция вывода>

<оператор if> → if ( <условное выражение> ) <блок> EL

EL → else <блок> | λ

<оператор for> → for ( <описание переменной> ; <условное выражение> ; <арифметическое выражение> ) <блок>

<оператор while> → while ( <условное выражение> ) <блок>

<условное выражение> → <сложное условное выражение> <второе сложное условное выражение>

<второе сложное условное выражение> → compop <сложное условное выражение><второе сложное условное выражение> | λ

<сложное условное выражение> → ( <условное выражение> ) | <простое условное выражение>

<простое условное выражение> → <арифметическое выражение> relop <арифметическое выражение>

<арифметическое выражение> → <слагаемое> <второе слагаемое>

<второе слагаемое> → addop <слагаемое> <второе слагаемое> | λ

<слагаемое> → <множитель> <второй множитель>

<второй множитель> → multop <множитель> <второй множитель> | λ

<множитель> → ( <арифметическое выражение> ) | constnum | id