**РУБЕЖНЫЙ КОНТРОЛЬ № 2**

**Примечания.**

1. **Выполняется в «бумажном» варианте.**
2. **При желании можно написать программу.**

**ВАРИАНТ 1 (1.1.1.)**

Показать, что грамматика G=({E, A}, {**id**, '(', ')', '[', ']', ';'}, P, E) с правилами P={E  **id,** E **id**(A), E  **id**[E],

A  E, A  E; A} не является LL(1)-грамматикой. Найти эквивалентную ей левофакторизованную LL(1)-грамматику G', построить LL(1)-анализатор для G' и выполнить разбор цепочки w = "**id**(**id**[**id**];**id**)".

**ВАРИАНТ 2 (1.1.2.)**

Показать, что грамматика G=({E}, {'.', '?', ':', '(', ')', **id**}, P, E) с правилами P={E  E.**id**(E) | (E) | E?E:E | **id**} не является LL(1)-грамматикой. Найти эквивалентную ей LL(1)-грамматику G', построить LL(1)-анализатор для G' и выполнить разбор цепочки w = "(**id**)?(**id**.**id**(**id**)):(**id**?**id**:**id**)".

**ВАРИАНТ 3 (1.1.3.)**

Показать, что грамматика G = ({S, L},{**a**, '(', ')', ','), P, S) с правилами P = {S → (L) | **a**, L → L, S | S} не является LL(1)-грамматикой. Найти эквивалентную ей LL(1)-грамматику G', построить LL(1)-анализатор для G' и выполнить разбор цепочки w = "(**a**, **a**)".

**ВАРИАНТ 4 (1.1.4.)**

Показать, что грамматика G=({E, T, X}, {**id**, '+', '[', ']', ','}, P, E) с правилами P={E  E+T | T, T  **id** | **id**[] | **id**[X], X  E, E | E } не является LL(1)-грамматикой. Найти эквивалентную ей LL(1)-грамматику G', построить LL(1)-анализатор для G' и выполнить разбор цепочки w="**id**+**id**[**id**+**id**, **id**[]]".

**ВАРИАНТ 5 (1.1.5.)**

Показать, что грамматика G=({E}, {**true, false, id, or, and, not**, '(', ')'}, P, E) с правилами P={E  E **or** E |

E **and** E | **not** E | (E) | **true** | **false** | **id**} является неоднозначной. Используя новые нетерминалы, устранить неоднозначность и явно определить приоритет булевых операций. Построить LL(1)-анализатор для преобразованной грамматики и выполнить разбор цепочки w="**id or id and not id**".

**ВАРИАНТ 6 (1.1.6.)**

Показать, что грамматика регулярных выражений G = ({R}, {'|', '\*', '(', ')', **a**, **b**}, P, R) с правилами

P = {R  R '|' R | R R | R '\*' | '(' R ')' | **a** | **b**} неоднозначна. Используя новые нетерминалы, устранить неоднозначность и явно определить приоритет операций. Построить LL(1)-анализатор для преобразованной грамматики и выполнить разбор цепочки w="(**a** | **b**)\***abb**".

**ВАРИАНТ 7 (1.1.7.)**

Показать, что грамматика G=({S, A}, {**a**, '(', ')', ','}, P, S) с правилами P={S  ( ) | **a** | ( A ), A  S | A, S} не является LL(1)-грамматикой. Найти эквивалентную ей не леворекурсивную и левофакторизованную LL(1)-грамматику G', построить LL(1)-анализатор для G' и выполнить разбор цепочки w="(( ),**a**,((**a**)))".

**ВАРИАНТ 8 (1.1.8.)**

Показать, что грамматика G = ({E}, {"!", "&&", "(", ")", **id**}, P, R) с правилами P = {E  "!" E, E E "&&" E, E  **id,** E "(" E ")"} неоднозначна. Используя новые нетерминалы, устранить неоднозначность и явно определить приоритет операций. Построить LL(1)-анализатор для преобразованной грамматики и выполнить разбор цепочки w="!(!**id** && **id**)".

**ВАРИАНТ 9 (1.1.9.)**

Показать, что грамматика G = ({Z, E, F, V, W}, {**i, m, o, c**}, P, Z) с правым концевым маркером $ и правилами P = {E  **m**E, E  **o**E**c**, E  VF, F  **m**E, F  , V  **i**W, W  **o**E**c**, W  , Z  E$} является LL(1)-грамматикой, построить для нее LL(1)-анализатор и выполнить разбор цепочки w = "**immiooicc**$".

**ВАРИАНТ 10 (1.1.10.)**

Показать, что грамматика G=({S, E, B, L}, {**id, num, print, while**, '(', ')', '{', '}', '>', ';'}, P, S) с правилами P = {S  **print**(E);|**while**(B)S|{L}, E  **id**|**num**, B  E>E, L  SL|} является LL(1)-грамматикой, построить для нее LL(1)-анализатор и выполнить разбор цепочки w="{**while**(**id**> **num**)**print**(**id**);**print**(**num**);}".

**ВАРИАНТ 11 (1.1.11.)**

Показать, что грамматика G = ({Exp, Op, Num}, {0,1,2,3,4,5,6,7,8,9, '+', '-'. '\*', '/', '(', ')'} с правилами P = {Exp Num | Exp Op Exp | (Exp), Op '+' | '-' | '\*' | '/', Num 0|1|2|3|4|5|6|7|8|9} не является LL(1)-грамматикой. Найти эквивалентную ей LL(1)-грамматику, построить LL(1)-анализатор для преобразованной грамматики и выполнить разбор цепочки w = "(9-1)/(3+1)".

**ВАРИАНТ 12 (1.1.12.)**

Показать, что грамматика G=({S,S', E}, {**a, b, i, t, e**}, P, S) с правилами P={S  **i**E**t**SS'|**а,** S'  **e**S|E  **b**} является неоднозначной. Построить LL(1)-анализатор для G и выполнить разбор цепочки w="**ibtibtaea**". Неоднозначность разрешать путем выбора правила с самой длинной правой частью.

**ВАРИАНТ 13 (1.1.13.)**

Показать, что грамматика G=({T, R}, {**num,** '(', ')', '–', '+', '\*'}, P, T) с правилами P = {T  –TT | +TT | (T) | R,

R \*RR | **num**} является LL(1)-грамматикой, построить для нее LL(1)-анализатор и выполнить разбор цепочки w="+ – 5 \* 2 3 4". Здесь **num –** это любая цифра**.**

**ВАРИАНТ 14 (1.1.14.)**

Показать, что грамматика G=({S, A, B, C}, {**a**, **b**, **c**}, P, S) с правилами P={S  AB, A  C**a**|B  B**a**AC|**c**,

C  **b**|} не является LL(1)-грамматикой. Найти эквивалентную ей не леворекурсивную LL(1)-грамматику, построить LL(1)-анализатор для преобразованной грамматики и выполнить разбор цепочки w="**baca**".

**ВАРИАНТ 15 (1.1.15.)**

Показать, что грамматика G = ({S, A, B, C}, {**a**, **b**, **c**, **x**}, P, S) с правилами P = {S  A**s**, A  BCA, A  BC**a**,

B  **b**, C  **c**} является LL(3)-грамматикой, но не является LL(1)- и LL(2)-грамматиками, построить LL(3)-таблицу для грамматики G, оставив только полезные столбцы, а не все комбинации входных символов.

**ВАРИАНТ 16 (1.1.16.)**

Показать, что грамматика G = ({S}, {true, false, ∨,∧,￢,(,)}, P, S) с правилами P = {S →(S) | S ∨ S | S ∧ S |

￢S | true | false} неоднозначна. Найти G′ ∈ LL(1) такую, что L (G) = L (G′). Построить LL(1)-анализатор для G′ и выполнить разбор цепочки w = “”. Приоритет операций должен быть определен явно.

––

**ВАРИАНТ 17 (1.1.11.)**

Показать, что грамматика G = ({Exp, Op, Num}, {0,1,2,3,4,5,6,7,8,9, '+', '-'. '\*', '/', '(', ')'} с правилами

P = {Exp Num | Exp Op Exp | (Exp), Op '+' | '-' | '\*' | '/', Num 0|1|2|3|4|5|6|7|8|9} не является LL(1)-грамматикой. Найти эквивалентную ей LL(1)-грамматику, построить LL(1)-анализатор для преобразованной грамматики и выполнить разбор цепочки w = "(9-1)/(3+1)".

**ВАРИАНТ 18 (1.1.18.)**

Для грамматики G = ({**id**, "=", "\*"} , {S, E, V}, P, S), с правилами P = {S  V = E, S  E, E  V, V  **id,** V  \*E} построить LR(1)-анализатор и выполнить разбор цепочки w = "\*\*\***id** = \*\*\***id**".

**ВАРИАНТ 19 (1.1.19.)**

Показать, что грамматика G = ({"**(**", " **)**", **a**}, {List, Sequence, Cell , Atom}, P, List) с правилами P = {List  **(**Sequence**)**, Sequence  Sequence Cell, Sequence  ε, Cell  List, Cell  Atom, Atom  **a**} не является LL(1)-грамматикой. Найти эквивалентную ей LL(1)-грамматику G', построить LL(1)-анализатор для G' и выполнить разбор цепочки w = "**((a)(aaa(a)))**".

**ВАРИАНТ 20 (1.1.12.)**

Показать, что грамматика G=({S,S', E}, {**a, b, i, t, e**}, P, S) с правилами P={S  **i**E**t**SS'|**а,** S'  **e**S|E  **b**} является неоднозначной. Построить LL(1)-анализатор для G и выполнить разбор цепочки w="**ibtibtaea**". Неоднозначность разрешать путем выбора правила с самой длинной правой частью.