Вариант 1

1. Язык выражений со сложением, натуральными числами и двумя видами скобок: квадратными и круглыми. Причём скобки не обязательно сбалансированы в общем, но сбалансированы относительно своего типа, и внутри квадратных скобок могут быть только префиксы правильных скобочных последовательностей (включая пустой и сами ПСП) круглых скобок. Примеры слов из языка: $[1+(2]+3)$, $((1+[2+((30)])))$. Не из языка: $[1]+2]$, $(12+[11)]$.

2. Язык атрибутной грамматики:

$$\begin{array}{lcl}
S \to S' & ; & S'.a == S'.b \\
S' \to T\$S' & ; & S'_0.a := T.a + S'_1.a,\ S'_0.b := T.b \\
S' \to T & ; & S'.a := T.a,\ S'.b := T.b \\
T \to TBa & ; & T_0.a := T_1.a + 1,\ T_0.b := T_1.b + B.b \\
T \to \varepsilon & ; & T.a := 0,\ T.b := 0 \\
B \to bB & ; & B_0.b := B_1.b + 1 \\
B \to \varepsilon & ; & B.b := 0
\end{array}$$

3. Язык $\{wa^nb^*c^nw^R \mid w \in \{a,b\}^+ \ \&\ n > 0\}$.

Вариант 2

1. Язык $\{wh_1(w)h_2(w) \mid h_1(a) = \varepsilon, h_1(b) = bb, h_2(b) = \varepsilon, h_2(a) = aa\}$.

2. Язык $\{w\$ \mid |w|_{aba} = |w|_{ba} \ \& \ w = w^R \ \& \ w \in \{a, b\}^*\}$

3. Язык атрибутной грамматики:

$$
\begin{array}{lcl}
S \to TC & ; & T.free\_a == C.iter \vee (C.iter == 1 \ \& \ T.free\_a == T.a) \\
T \to aTb & ; & T_0.a := T_1.a + 1 \\
T \to K & ; & T.a := 0, T.free\_a := K.iter \\
K \to aK & ; & K_0.iter := K_1.iter + 1 \\
K \to \varepsilon & ; & K.iter := 0 \\
C \to cC & ; & C_0.iter := C_1.iter + 1 \\
C \to \varepsilon & ; & C.iter := 0
\end{array}
$$

Вариант 3

1. Язык $\{wv^R aaavccccw^R \mid w \in \{a,b\}^* \;\&\; v \in \{a,c\}^*\}$.

2. Язык тождественно истинных логических формул со связками &, $\neg$, скобками и единственной переменной P таких, что в них нет двойных отрицаний.

3. Язык атрибутной грамматики:

$$\begin{array}{lll}
S \to aS' \mid bS' & ; & S'.inh_attr := 1 \\
S' \to aS' \mid bS' & ; & S'_1.inh_attr := S'_0.inh_attr + 1 \\
S' \to T & ; & T.inh_attr := S'.inh_attr \\
T \to aTa \mid bTb \mid cTc & ; & T_1.inh_attr := T_0.inh_attr - 1 \\
T \to \varepsilon & ; & T.inh_attr == 0
\end{array}$$

Вариант 4

1. Язык логических формул только со связкой $\Rightarrow$, скобками и переменной P, не являющихся ни тождествами, ни тривиальными противоречиями, причём таких, что в них нет лишних скобок (навешенных на одно и то же выражение повторно).

2. Язык, описывающийся следующей атрибутной грамматикой:

$$
\begin{array}{lll}
S \to aS'a \,|\, bS'b & ; & S'.inh\_attr := 1 \\
S' \to aS'a \,|\, bS'b & ; & S'_1.inh\_attr := S'_0.inh\_attr + 1 \\
S' \to T & ; & T.inh\_attr := S'.inh\_attr \\
T \to aTa \,|\, bTb \,|\, cTc & ; & T_1.inh\_attr := T_0.inh\_attr - 1 \\
T \to \varepsilon & ; & T.inh\_attr == 0
\end{array}
$$

3. Язык $\{w_1 u u^R w_2 \mid |u| > 0 \ \& \ w_1 \neq u z_1 \ \& \ w_2 \neq z_2 u \ \& \ u, w_1, w_2 \in \{a,b\}^*\}$.

Вариант 5

1. Язык, определяемый следующей атрибутной грамматикой:

$$
\begin{array}{lcl}
S \to ScS & ; & S_0.attr := S_1.flag \;\&\; S_2.flag, \\
 & & S_0.val = min(S_1.val, S_2.val),\ S_0.val \cdot S_0.flag == 0 \\
S \to T & ; & S.flag := T.flag,\ S.val := T.val \\
T \to aTT & ; & T_0.flag := T_1.flag \vee T_2.flag, T_0.val := |T_1.val - T_2.val| \\
T \to bb & ; & T_0.flag := 1,\ T_0.val := 1 \\
T \to \varepsilon & ; & T.flag := 0,\ T_0.val := 0
\end{array}
$$

2. Язык SRS $a \to ba$, $abb \to ab$, $ba \to ab$ на базисе $a^n b^n$.

3. Язык $\{a^n b^m w c w^R c^{n+m} \mid w \in \{a, c\}^*\}$.

Вариант 6

1. Язык, определяемый следующей атрибутной грамматикой:

$$\begin{array}{lcl} S \to SS & ; & S_0.attr := S_1.attr \mathbin{\&} S_2.attr,\ S_0.attr == 1 \\ S \to bTb & ; & S.attr := \neg T.attr \\ T \to aT & ; & T_0.attr := T_1.attr \\ T \to bTb & ; & T_0.attr := \neg T_1.attr \\ T \to \varepsilon & ; & T.attr := 0 \end{array}$$

2. Язык SRS $baa \to ba$, $aab \to ba$ над $a^n b^n a^n$.

3. Язык $\{a^n b^n w_1 a w_2 \mid w_i \in \{a, b\}^+ \mathbin{\&} |w_1| = |w_2|\}$.

Вариант 7

1. Язык выполнимых формул (т.е. истинных хотя бы в одной модели) без вложенных кванторов в моделях с единственным элементом. Предикаты $P(x)$, $Q(x)$, связка $\Rightarrow$, разрешённое имя переменной — только x, свободные вхождения переменных в формулу не допускаются. Подсказка: существует всего четыре различимые модели.

2. Язык $\{c^i a^n b^k c^j \mid k = n \vee (i + j > 1 \ \& \ i < j)\}$.

3. Язык, порождаемый следующей атрибутной грамматикой:

$$
\begin{array}{lll}
S \to Q & ; & \\
Q \to QaQc & ; & Q_1.attr == Q_2.attr,\ Q_0.attr := Q_1.attr \\
Q \to aAa & ; & Q.attr := A.attr + 2 \\
A \to BB & ; & A.attr := B_1.attr + B_2.attr \\
A \to AA & ; & A_0.attr := A_1.attr + A_2.attr \\
B \to bb & ; & B.attr := 2
\end{array}
$$

Вариант 8

1. Язык грамматик без ε-правил, содержащих только нетерминалы S, A и B и терминал a, которые порождают только слова из букв a чётной длины (возможно, не все). Считаем, что правила разделяются точкой с запятой. Алфавит языка: $\{A, B, a, \to, ;\}$.

2. Язык $\{a^k b^n c^m a^i \mid (k+n=m) \vee (n=0 \ \& \ k=i)\}$.

3. Язык атрибутной грамматики:

$$
\begin{array}{lcl}
[S] \to [Pred] & ; & \\
[Pred] \to [Poly]\_[Expr]\_[Expr] & ; & Expr_1.out == Expr_2.out \\
[Expr] \to [Op].[Expr] & ; & Op.in == Expr_1.out,\ Expr_0.out := Op.out \\
[Expr] \to [Op]\_[Val] & ; & Op.in == Val.type,\ Expr.out := Op.out \\
[Op] \to G & ; & Op.in := R,\ Op.out := A \\
[Op] \to A & ; & Op.in := A,\ Op.out := R \\
[Op] \to R & ; & Op.in := A,\ Op.out := A \\
[Poly] \to E & ; & \\
[Val] \to ([Val]) \mid [Val]* \mid [Val][Val] \mid a & ; & Val_0.type := R
\end{array}
$$

Вариант 9

1. Язык всех академических регулярных выражений, которые порождают конечные языки. Алфавит $\{a, b, \varepsilon, *, (,), |\}$ (т.е. пустое слово явно входит в регулярки).

2. Язык ложных формул логики предикатов в пустой модели без вложенных кванторов. Предикаты $Q(x)$, $P(x)$, связки $\Rightarrow$ и $\neg$, разрешённые имена переменных: x и y, свободные вхождения переменных в формулу не допускаются. Пример слов языка: $\forall x(Q(x)) \Rightarrow \exists x(P(x) \Rightarrow Q(x))$, $\neg\forall y P(y)$.

3. Язык атрибутной грамматики:

$[S] \to [Pred]$	;	
$[Pred] \to= \_[Expr]\_[Expr]$	;	$Expr_1.val == Expr_2.val$
$[Expr] \to [Op].[Expr]$	;	$Expr_0.val := (Op.fun)\ (Expr_1.val)$
$[Expr] \to [Op]\_[Val]$	;	$Expr.val := (Op.fun)\ (Val.val)$
$[Op] \to Double$	;	$Op.fun := (\lambda x \to x * 2)$
$[Op] \to Square$	;	$Op.fun := (\lambda x \to x * x)$
$[Val] \to 1$	;	$Val.val := 1$
$[Val] \to 1[Val]$	;	$Val_0.val := 1 + Val_1.val$

Вариант 10

1. Язык всех сентенциальных форм, порождаемых грамматикой $S \to aSbS$, $S \to a$ из начального нетерминала S, таких что в них встречается подслово bab.

2. Язык $\{a^n b^k w c^i w^R \,|\, (k = n \vee i > 1) \mathbin{\&} w \in \{a,b\}^*\}$.

3. Язык, определяемый следующей атрибутной грамматикой:

$$
\begin{array}{lcl}
S \to AaSaA & ; & S_0.attr := S_1.attr + A_1.attr,\ A_2.attr < S_1.attr \\
S \to b & ; & S.attr := 1 \\
A \to aAb & ; & A_0.attr := A_1.attr + 1 \\
A \to bAb & ; & A_0.attr := A_1.attr + 2 \\
T \to \varepsilon & ; & A.attr := 0
\end{array}
$$

Вариант 11

1. Язык всех сентенциальных форм, порождаемых грамматикой $S \to aSa$, $S \to bSb$, $S \to c$ из начального нетерминала S, таких что в них поровну букв a и b.

2. Язык $\{a^n c^m b^m c^i b^k \mid k = n \vee (i > 1 \,\&\, i = k)\}$.

3. Язык, определяемый следующей атрибутной грамматикой:

$$\begin{array}{lll} S \to AaaSaaA & ; & A_2.attr < A_1.attr \\ S \to b & ; & S.attr := 1 \\ A \to abA & ; & A_0.attr := A_1.attr + 1 \\ A \to bAb & ; & A_0.attr := A_1.attr + 2 \\ A \to Aab & ; & A_0.attr := A_1.attr + 1 \\ A \to \varepsilon & ; & A.attr := 0 \end{array}$$

Вариант 12

1. Язык всех сентенциальных форм, порождаемых грамматикой $S \to aaS$, $S \to Sbb$, $S \to \varepsilon$ из начального нетерминала S, таких что в них одинаково число букв a и b.

2. Язык $\{a^n c^m b^m c^i b^k \mid k = n \vee (i > 1 \,\&\, i = k)\}$.

3. Язык, определяемый следующей атрибутной грамматикой:

$$
\begin{array}{lcl}
S \to AaSaA & ; & A_2.attr < A_1.attr \\
S \to b & ; & S.attr := 1 \\
A \to abA & ; & A_0.attr := A_1.attr + 1 \\
A \to bAb & ; & A_0.attr := A_1.attr + 2 \\
A \to Aa & ; & A_0.attr := A_1.attr \\
A \to \varepsilon & ; & A.attr := 0
\end{array}
$$

Вариант 13

1. Язык SRS с правилами $abc \to cba$, $ac \to ca$, $ba \to ab$ над базисом $a^*b^*c^*$.

2. Язык $\{a^k b^n c^i a^{k+j} \mid j > k \vee (i > 1 \,\&\, i = n)\}$.

3. Язык, определяемый следующей атрибутной грамматикой:

$$
\begin{array}{lll}
S \to SbbA & ; & S_1.b < A.b,\ S_0.b := S_1.b + 1 + A.b \\
S \to b & ; & S.b := 1 \\
A \to aAa & ; & A_0.b := A_1.b \\
A \to bAb & ; & A_0.b := A_1.b + 1 \\
A \to \varepsilon & ; & A.b := 0
\end{array}
$$

Вариант 14

1. Язык $\{w \mid |w|_{ab} = |w|_{baa} \;\&\; w = w^R\}$. Алфавит $\{a, b\}$.

2. Язык $\{wz_1h(w)z_2h(w) \mid |w| > 0 \;\&\; h(a) = ba \;\&\; h(b) = \varepsilon\}$. Алфавит $\{a, b\}$.

3. Язык, определяемый следующей атрибутной грамматикой:

$$
\begin{array}{lcl}
S \to ASA & ; & A_1.b == A_2.b,\ S_1.b > A_1.b,\ S_0.b := S_1.b + 2 \cdot A.b \\
S \to b & ; & S.b := 1 \\
A \to aA & ; & A_0.b := A_1.b \\
A \to bbA & ; & A_0.b := A_1.b + 2 \\
A \to \varepsilon & ; & A.b := 0
\end{array}
$$

Вариант 15

1. Язык $\{w_0 u^n w_1 u w_2 \mid |w_0| < 3 \,\&\, |u| > 0 \,\&\, n > 1\}$. Алфавит $\{a, b\}$.

2. Язык $\{a^n b^m c^k \mid n < m \vee m < k \vee n < k\}$.

3. Язык, описывающийся следующей атрибутной грамматикой ($lookup$ — поиск по таблице значений $Table$, т.е. возвращает по $[Name].id$ такое $[Val].val$, что $([Name].id = [Val].val) \in Table$, и 0, если $[Name].id$ отсутствует в таблице):

$$
\begin{array}{lcl}
[S] \to \{[Decl]\}[Exp] & ; & [Exp].val == 1, \\
 & & [Exp].inh\_table := [Decl].table \\
[Decl] \to & ; & [Name].id \notin [Decl]_1.vars, \\
\quad ([Name] = [Val])[Decl] & & [Decl]_0.table := [Decl]_1.table \cup \{[Name].id = [Val].val\}, \\
 & & [Decl]_0.vars := [Decl]_1.vars \cup \{[Name].id\} \\
[Decl] \to \varepsilon & ; & [Decl].vars := \varnothing,\ [Decl].table := \varnothing \\
[Exp] \to [Name] & ; & [Exp].val := lookup([Name].id, [Exp].inh\_table) \\
[Exp] \to [Exp] \,\&\, [Exp] & ; & [Exp]_0.val := min([Exp]_1.val, [Exp]_2.val), \\
 & & [Exp]_1.inh\_table := [Exp]_0.inh\_table, \\
 & & [Exp]_2.inh\_table := [Exp]_0.inh\_table \\
[Name] \to a[Name] & ; & [Name]_0.id := [Name]_1.id +\!\!+ a \\
[Name] \to \varepsilon & ; & [Name].id := \varepsilon \\
[Val] \to 0 & ; & [Val].val := 0 \\
[Val] \to 1 & ; & [Val].val := 1
\end{array}
$$

Вариант 16

1. Язык 2-backref-регулярок в алфавите $\{[_1, [_2,]_1,]_2, x_1, x_2, a, b, (,), *\}$ (без альтернатив).

2. Язык атрибутной грамматики:

$$
\begin{array}{lcl}
S \to SS & ; & S_1.a1 == S_1.a2 + S_2.a1,\ S_0.a1 := S_1.a1 + S_2.a1, \\
 & & S_0.a2 = S_1.a1 \\
S \to a & ; & S.a1 := 1,\ S.a2 := 0
\end{array}
$$

3. Язык $\{wa^n c^n b^* w^R \mid n > 0 \;\&\; w \in \{a, b\}^+\}$.

Вариант 17

1. Язык всех академических регулярных выражений без альтернативы, вложенных скобок и звёздной высоты $= 1$ (т.е. без вложенных итераций), которые порождают языки, в которых есть хотя бы одно слово с подстрокой ab. Алфавит $\{a, b, *, (,)\}$.

2. Язык $\{w_1 w_2 \mid w_1 = v_1 a v_2 \;\&\; w_2 = u_1 b u_2 \;\&\; |v_1| = |v_2| \;\&\; |u_1| > |u_2|\}$.

3. Язык атрибутной грамматики:

$$
\begin{array}{lll}
[S] \to [Pred] & ; & \\
[Pred] \to= \_[Expr]\_[Expr] & ; & Expr_1.val == Expr_2.val \\
[Expr] \to [Op].[Expr] & ; & Expr_0.val := (Op.fun)\ (Expr_1.val) \\
[Expr] \to [Op]\_[Val] & ; & Expr.val := (Op.fun)\ (Val.val) \\
[Op] \to Mod & ; & Op.fun := (\lambda x \to x\ mod\ 2) \\
[Op] \to Double & ; & Op.fun := (\lambda x \to x * 2) \\
[Val] \to 1 & ; & Val.val := 1 \\
[Val] \to 1[Val] & ; & Val_0.val := (Val_1.val)^2 + Val_1.val + 1
\end{array}
$$

Вариант 18

1. Язык всех академических регулярных выражений без итерации, которые порождают языки, в которых есть слово ab. Алфавит $\{a, b, |, (,)\}$.

2. Язык SRS $ac \to ca$, $c \to bcb$, $bbc \to cb$ над базисом $a^n cda^n c$.

3. Язык атрибутной грамматики:

$$
\begin{array}{lcl}
[S] \to [Pred] & ; & \\
[Pred] \to= \_[Expr]\_[Expr] & ; & Expr_1.val == Expr_2.val,\ Expr_1.val < 3 \\
[Expr] \to [Op].[Expr] & ; & Expr_0.val := (Op.fun)\ (Expr_1.val) \\
[Expr] \to [Op]\_[Val] & ; & Expr.val := (Op.fun)\ (Val.val) \\
[Op] \to Mod & ; & Op.fun := (\lambda x \to x\ mod\ 2) \\
[Op] \to Double & ; & Op.fun := (\lambda x \to x * 2) \\
[Val] \to 1 & ; & Val.val := 1 \\
[Val] \to 1[Val] & ; & Val_0.val := (Val_1.val)^2 + Val_1.val + 1
\end{array}
$$

Вариант 19

1. Язык $\{c^i a^n b^k c^j \mid k = n \vee i + j > 1\}$.
2. Язык SRS $a \to bab$, $aa \to aaa$ над множеством базисных слов $b^n a^n$.
3. Язык $\{wv^R uvuw^R \mid |u| = 2 \;\&\; u \in \{a,c\}^+ \;\&\; w \in \{a,b\}^+ \;\&\; v \in \{b,c\}^+\}$.

Вариант 20

1. Язык $\{c^i a^n b^k a^j \mid (k > n) \vee (i = j \;\&\; n > 2)\}$.
2. Язык SRS $a \to ab$, $aa \to baab$ над множеством базисных слов $a^n b a^n$.
3. Язык слов, первая половина которых не начинается с палиндрома.

Вариант ε

1. Язык синтаксически корректных вызовов функций в языке Рефал. Вызов функции заключается в угловые скобки, аргумент-выражение от имени функции отделяется пробелом, выражение может быть вызовом функции, конкатенацией двух выражений, выражением в скобках, строкой (в одинарных кавычках) или переменная. Внутри строки могут быть экранированные обратным слешем одинарные кавычки. Также обратным слешем внутри кавычек экранируется сам обратный слеш.

2. Язык образцов в языке Рефал, которые распознают множества слов, обладающие префикс-свойством. Алфавит образцов: $\{e_1, e_2, s_1, s_2, a, b\}$. Здесь e_i — переменные типа выражение, s_i — переменные типа буква, a, b — буквы.

3. Язык атрибутной грамматики для Рефал-предложений:

$$
\begin{array}{lll}
[S] \to [Pattern] = [Expr]; & ; & Expr.vars \subseteq Pattern.vars \\
[Pattern] \to [Evar][Pattern] & ; & Pattern_0.vars := \\
 & & Pattern_1.vars \cup \{Evar.name\} \\
[Pattern] \to [Const] & ; & Pattern.vars := \varnothing \\
[Const] \to (a|b|c)^* & ; & \\
[Expr] \to < [Function]\_[Expr] > [Expr] & ; & Expr_0.vars := Expr_1.vars \cup Expr_2.vars \\
[Expr] \to [Expr][Expr] & ; & Expr_0.vars := Expr_1.vars \cup Expr_2.vars \\
[Expr] \to [Const] & ; & Expr.vars := \varnothing \\
[Expr] \to [Evar] & ; & Expr.vars := \{Evar.name\} \\
[Evar] \to e.[Num] & ; & Evar.name := e.(Num.str) \\
[Num] \to 1[Num] & ; & Num_0.str := 1Num_1.str \\
[Num] \to 0[Num] & ; & Num_0.str := 0Num_1.str \\
[Num] \to \varepsilon & ; & Num.str := \varepsilon
\end{array}
$$

Вариант $*$

1. Язык множества переходов в конечном автомате, определяющих автомат-ловушку. Финальное состояние — символ q_f, стартовое — символ q_0; алфавит: $\{q_0, q_1, q_2, q_f, <, >, a, \to, ;\}$. Правила записываются стандартно: $<q_i,\ s>\to q_j$; (и разделяются точкой с запятой).

2. Язык регулярных выражений, для которых автомат Глушкова детерминирован. Алфавит $\{a, b, *, |, (,)\}$.

3. Язык атрибутной грамматики для регулярок:

$[S] \to [Regexp]$	;	
$[Regexp] \to [Regexp][Regexp]$	;	$Regexp_1.val \neq \varepsilon$, $Regexp_2.val \neq \varepsilon$ $Regexp_0.val := Regexp_1.val +\!\!+ Regexp_2.val$
$[Regexp] \to ([Regexp] \mid [Regexp])$	;	$Regexp_1.val \neq \varepsilon \vee Regexp_2.val \neq \varepsilon$, $Regexp_1.val \neq \mid$, $Regexp_0.val := \mid$
$[Regexp] \to ([Regexp])*$	;	$Regexp_1.val \neq \varepsilon$, $Regexp_1.val \neq *$, $Regexp_0.val := *$
$[Regexp] \to \varepsilon$	;	$Regexp.val := \varepsilon$
$[Regexp] \to a$	;	$Regexp.val := a$
$[Regexp] \to b$	;	$Regexp.val := b$