1. Дайте формальное определение цепочки в алфавите V.
Любая последовательность символом конечной длины в алфавите V называется цепочкой символов.

2. Перечислите основные операции над цепочками символов.
Конкатенация, объединение или сложение цепочек
Реверсом цепочки называется операция записи в обратном порядке
α^n – возведение в степень n цепочки α – называется - кратная конкатенация цепочки с самой собой, т.е.
Длина цепочки обозначается так |α|.

3. Назовите способы формального задания языка.
Перечисление всех допустимых цепочек языка (способ формальный, на практике нереализуем, т.к. в общем случае множество цепочек языка бесконечно и перечислить их невозможно).
Указание способа порождения цепочек языка (задание грамматики языка) - применение т.н. генератора.
Задание метода распознавания цепочек языка - использование распознавателя.

4. Какими элементами задается формальная грамматика?
Порождающая грамматика есть четверка элементов $G =< T, N, P, S >$, где T – множество терминальных символов (иначе, терминалов); N – множество нетерминальных символов (иначе, нетерминалов); P – множество правил вывода, вида $\alpha \rightarrow \beta$. Здесь α называют левой частью правила, а β – правой. Символ «→» читается, как «может иметь вид». S – начальный символ (иначе, цель) грамматики.

5. Назовите типы формальных грамматик по классификации Хомского.

Тип 0 — неограниченные правила имеют следующий вид:
$\alpha \rightarrow \beta$, где $\alpha \in V+$ и $\beta \in V^*$.
Тип 1 — контекстно-зависимые Если для всех правил грамматики выполняется следующее соглашение: $\alpha \rightarrow \beta$, где $\alpha, \beta \in V+$ и $1 \leq |\alpha| \leq |\beta|$
Тип 2 — контекстно-свободные а все правила которой имеют следующий вид:
$A \rightarrow \alpha$, где $A \in VN$, $\alpha \in V+$. Если $\alpha \in V^*$, то такая грамматика является укорачивающейся КС грамматикой.
Тип 3 — регулярные может быть праволинейной или леволинейной. Грамматика называется праволинейной, если каждое правило грамматики имеет вид: $A \rightarrow t$, или $A \rightarrow tB$, где $A, B \in VN$; $t \in VT$.

Грамматика называется леволинейной, если каждое правило грамматики имеет вид: $A \rightarrow t$, или $A \rightarrow Bt$, где $A, B \in VN; t \in VT$.

6. Какое множество называется регулярным?
Множество L слов в конечном алфавите A называется регулярным множеством (языком), если оно определяется некоторым регуляным выражением. Заметим, что одно и то же регулярное множество может определяться разными регулярными выражениями.

7. Какое выражение называется регулярным?
Регулярное выражение – это способ определения регулярного языка,где используется только словарь терминальных символов. В этом случае словарь нетерминальных символов и множество правил порождающей грамматики не используются, как, собственно, и сама порождающая грамматика.

8. Дайте формальное определение конечного автомата.
Конечный автомат формально определяется как шестерка, или кортеж, следующим образом: $A = (X, Y, S, s0, \delta, \lambda)$, где: X – конечное непустое множество входных сигналов (входной алфавит); Y – конечное непустое множество выходных сигналов (выходной алфавит); S – конечное непустое множество состояний; s0 – начальное состояние автомата, $s0 \in S$; $\delta: S \times X \rightarrow S$ – функция переходов; $\lambda: S \times X \rightarrow Y$ – функция выходов.

9. Какой конечный автомат называется недетерминированным?
Недетерминизм конечного автомата состоит в том, что из текущего символа, находясь в некотором состоянии, автомат может перейти в несколько возможных состояний. Недетерминированный конечный автомат (НКА) – это пятерка $< K, T, D, H, S >$, в которой K – конечное множество состояний; T – конечное множество возможных символов на входе; D – функция разрешенных переходов между состояниями; H – начальное состояние; S – множество заключительных состояний.

10. Какой конечный автомат называется детерминированным?
Тот где из каждого состояния детерминированный конечный автомат (ДКА) может перейти только в одно допустимое состояние. ДКА является частным случаем НКА. Другими словами, класс языков, определяемых НКА, совпадает с классом языков, определяемых ДКА. Из этого следует важный

вывод о том, что НКА может быть преобразован в ДКА и наоборот.

11. Какой вывод цепочки называется правосторонним?
Вывод цепочки $\beta \in (VT)^*$ из $S \in VN$ в КС-грамматике $G = (VT, VN, P, S)$, называется правым (правосторонним), если в этом выводе каждая очередная сентенциальная форма получается из предыдущей заменой самого правого нетерминала.

12. Какой вывод цепочки называется левосторонним?
Левосторонним выводом слова называется такой вывод слова , в котором каждая последующая строка получена из предыдущей путем замены по одному из правил самого левого встречающегося в строке нетерминала.

13. Дайте определение дереву разбора строки.
Деревом вывода для заданной грамматики $G =< T, N, P, S >$ называется древовидная структура данных, которая удовлетворяет набору условий:
1. Каждая вершина дерева – это либо терминал T, либо нетерминал N, либо, если грамматика допускает пустую строку, ε.
2. Корень дерева есть стартовый символ – цель грамматики S.
3. Листьями дерева могут быть только терминальные символы.
4. Потомки $a1, a2, \dots , ai$ каждого узла A дерева соответствуют правилу вывода $A \rightarrow a1a2 \dots ai$.
5. Если грамматика содержит правило вывода $A \rightarrow \varepsilon$, то этому правилу соответствует узел дерева, единственным потомком которого является лист ε

14. Чем отличается стратегия нисходящего и восходящего построения дерева разбора строки?
При нисходящем синтаксическом анализе в основном ищутся левые порождения. При восходящем синтаксическом анализе в основном ищутся правые порождения. Нисходящий анализ исходит из символа предложения S (цели грамматики) и генерирует предложение. При восходящем анализе разбор начинается с предложения, которое необходимо свернуть в S.

15. Поясните сущность метода рекурсивного спуска.
Метод рекурсивного спуска (РС) реализует нисходящий синтаксический разбор (сверху-вниз) с помощью рекурсивных функций. Для каждого нетерминала создается отдельная функция. Функция называется так же, как и нетерминал. Функция должна найти во входной строке подцепочку, выводимую из этого нетерминала (или сообщить об ошибке). Для написания исходного кода каждой такой функции

используются правила вывода соответствующей грамматики. При этом нетерминал в правой части правила вывода эквивалентен вызову процедуры, соответствующей данному нетерминалу. Работа алгоритма может быть описана следующим образом. Нисходящий разбор (сверху-вниз) всегда начинается с символа предложения S. В точке входа в программу (функция main) вызывается функция S. Задача функции определить, выводится ли входная цепочка из символа S. В теле функции S могут вызваться другие функции, поскольку в процессе вывода могут встречаться нетерминалы. Предполагается, что все цепочки обязательно заканчиваются маркером конца ввода $\perp$. При этом концом ввода может служить как конец исходной программы (конец файла), так и конец текущей строки.

16. Запишите достаточные условия применимости метода рекурсивного спуска.

Первым требованием является время работы алгоритма разбора, оно должно быть пропорциональным длине входной цепочки. Кроме того, алгоритм должен быть корректным, т.е.: 1. Должен быть способен распознать любую цепочку, принадлежащую языку. 2. Должен отвергать цепочки, не принадлежащие языку. 3. Не должен зацикливаться при любом вводе.

17. Дайте определения множествам first(α) и follow(A).

$first(\alpha)$ – множество терминалов, с которых начинаются цепочки $\alpha \in (T \cup N) *$ заданной грамматики $G = \{T, N, P, S\}$. Утверждение 7. Если в грамматике присутствуют правила вывода вида $X \rightarrow \alpha|\beta$, где α и β начинаются с одних и тех же символов (т.е. $first(\alpha) \cap first(\beta) \neq \varnothing$), то метод рекурсивного спуска для данной грамматике неприменим.

Множество $follow(X)$ включает все терминалы, которые могут появляться при разборе цепочки непосредственно справа от X, т.е. $follow(X) = \{a \in T | S \Rightarrow \alpha X \beta, \beta = a\gamma\}$ Метод рекурсивного спуска применим для КС-грамматик, таких что для любой пары альтернатив $A \rightarrow \alpha|\beta$, справедливо: $first(\alpha) \cap first(\beta) = \varnothing$ Только одно из правил $\alpha \Rightarrow \varepsilon$ или $\beta \Rightarrow \varepsilon$ верно Если $\alpha \Rightarrow \varepsilon$, $first(\beta) \cap follow(A) = \varnothing$ Сделаем важное замечание. В общем случае не существует алгоритма, способного по данной грамматике построить эквивалентную грамматику, для которой метод рекурсивного спуска применим.

18. Изобразите общую схему работы транслятора и поясните ее элементы.

Транслятор обычно выполняет свою работу в два этапа: 1. Этап анализа исходного текста. 2. Этап синтеза, в результате

которого генерируется машинноориентированное представление.
Перечислим основные этапы трансляции: 1. Лексический анализ – оперирует с исходным текстом программы. 2. Синтаксический анализ – соединен с выходом лексического анализатора и входом семантического анализатора. 3. Семантический анализ – генерирует абстрактное синтаксическое дерево разбора. 4. Генерация машинно-независимого кода – переводит абстрактное дерево разбора в последовательность инструкция, не привязанную к конкретной архитектуре. 5. Оптимизация машинно-независимого кода – служит для исключения из кода бесполезных инструкций. 6. Распределение памяти – выделяет для каждой переменной в программе область памяти и закрепляет за ней определенный адрес. 7. Генерация машинного кода – выдает последовательность инструкций, пригодную для исполнения на ЭВМ.

19. Какой тип грамматик лежит в основе лексического анализатора?
Лексический анализ текста проводится по регулярной грамматике. Известно, что регулярная грамматика эквивалентна конечному автомату, следовательно, для написания лексического анализатора необходимо построить диаграмму состояний, соответствующего конечного автомата.

20. Дайте формальное определение задачи синтаксического анализатора.
Основная задача синтаксического анализа состоит в нахождении порождения для заданного выражения (или принятия решения о том, что порождения не существует). В основе анализа лежит формальная грамматика используемого языка. Таким образом, задача синтаксического анализа разбивается на две подзадачи: 1. Необходимо определить соответствует ли последовательность лексем на входе анализатора синтаксису языка, задаваемому формальной грамматикой. 2. Необходимо построить для данной последовательности дерево разбора. В основе синтаксического анализа большинства языков программирования лежат КС-грамматики.

21. Приведите примеры стандартных семантических проверок при разборе языка программирования.
Семантический анализ необходим для устранения особенностей языка, не поддающихся описанию средствами формальной грамматики. В частности, проверка типов и область видимости переменных происходит на семантической фазе анализа. Семантической проверки на повторное объявление одного и того же идентификатора.