判断题

第一章

1. 源语言是编写被编译的程序使用的语言。
2. 解释执行与编译执行的根本区别在于解释程序对源程序没有真正进行翻译。
3. “遍”是指对源程序从头到尾扫描一遍，并做相应的加工处理。
4. 编译程序是将源程序翻译成等价的目标程序的程序。
5. 宿主语言是目标机的目标语言。
6. 编译程序是应用软件。

第二章

1. NFA 和 DFA 的区别之一是映射函数是否唯一。
2. NFA 的初始状态可以是多个。 DFA 只有一个初始状态。
3. 若一个文法是递归的，则它所产生语言的句子个数必定是无穷的。
4. 存在这样的语言，它能被确定的有穷自动机识别，但不能用正规表达式表示。
5. 设有字母表 V，则 $V_T \cap V_N = \Phi$。
6. 有文法 G1=G2，则 L(G1)=L(G2)。
7. 文法等价是指所描述的语言是完全相同的。
8. 一个确定有限状态自动机中，有且仅有一个唯一的终态。
9. 设 R 和 S 分别是字母表Σ上的正规式，则有 L(R|S)=L(R)∪L(S)。
10. 自动机 M1 和 M2 的状态数不同，则二者必不等价。
11. 确定有限自动机以及非确定有限自动机都能正确地识别正规集。
12. 对任意一个右线性正规文法 G，都存在一个 NFA M，满足 L(G)=L(M)。
13. 对任何正规式 e，都存在一个 NFA M，满足 L(M)=L(e)。
14. 从一个句型到另一个句型的推导过程是唯一的。
15. 最右推导是最右规约的逆过程，最左推导是最左规约的逆过程。
16. 一张转换图只包含有限个状态，其中有一个被认为是初态，最多只有一个终态。
17. 二义文法不是上下文无关文法。
18. 对能用有限自动机描述的一个语言，该语言的一子集所构成的语言也一定能用有限自动机来描述。
19. 对任意文法 G，都存在相应的正规式与之等价。
20. 对文法 G 中的一个句子，如果能够找到两种以上的推导，则该句子是二义性的。
存在两个不同语法分析树
21. 文法是描述语言的语法结构的形式规则。

第三章

1. 词法分析器输出结果中的单词属性名是进行语法分析的文法的终结符。
2. Lex 是编译程序自动生成工具。 自动生成词法分析器。
3. 词法分析识别出来的是具有独立意义的最小语法单位。
4. 词法分析作为单独的一遍来处理较好。
5. 词法分析依据的是语言的文法规则。 词法规则
6. 编译的预处理程序的处理对象是源程序。

第四章

1. 语法分析识别出来的是具有独立意义的最小语法范畴。
2. 语法分析方法中的递归下降分析法属于自顶向下分析方法。
3. 一棵语法树反映了其叶结点从左向右连接形成的句型(句子)的任意推导情况。

4. 若文法规则存在 P→P'a ，则 first(P') =first(P)。
5. 对文法中的某个句子，如果存在多种（多于一种）不同的最右推导，则也会存在多种不同的最左推导。
6. 包含左递归的文法肯定不能直接用 LL 分析法来分析。
7. 要构造行之有效的自上而下的分析器，则必须消除左递归。
8. 产生式是定义语法范畴的一种书写规则。
9. 语法分析的任务试分析语句是如何构成程序的。
10. 自上而下分析及自下而上分析中的“下”指的是被分析的源程序串。

第五章

1. 不同的语法分析方法仅仅是分析识别的模式不同，而扫描模式都是一样的。
2. 算符优先分析法每次规约的都为当前句型的素短语，而 LR 分析法规约的为当前句型的句柄。
3. 在 LR 文法的分析过程中，分析栈中的内容对应某合法句型的前缀。
4. 自下而上的分析法是一种“移进—归约”法。
5. LR 法是自顶向下语法分析方法。
6. LR 分析法在自左至右扫描输入串时就能发现错误，但不能准确地指出出错地点。
7. LR 分析技术无法适用二义文法。
8. 简单优先文法允许任意两个产生式具有相同右部。
9. 一个句型的句柄一定是文法某产生式的右部。
10. 一个句型的直接短语是唯一的。

第六章

1. 符号表在编译阶段非常重要，但符号表不会带入运行阶段。
2. 综合属性是用于“自上而下”传递信息。
3. 只含有综合属性的属性文法称为 S-属性文法，它是 L-属性文法的特例。
4. 对于任何一个编译程序来说，产生中间代码是必不可少的。
5. 逆波兰表示法表示表达式时无须使用括号。
6. 使用语法制导翻译方法的编译程序能同时进行语法分析和语义分析。
7. 返填就是稍后填写转移指令的地址。
8. 三元式同间接三元式是等价的。
9. 符号表中的信息栏中登记了每个名字的属性和特征等有关信息，如类型、种属、所占单元大小、地址等等。
10. 逆波兰法表示的表达式亦称前缀式。
11. 在编译中进行语法检查的目的是为了发现程序中所有错误。

第七章

1. 动态存储分配是指编译程序运行时才能确定其全部数据空间的大小。
2. C 语言既可以采取静态存储分配又可以采取动态存储分配。
3. 对于数据空间的存贮分配，FORTRAN 采用动态贮存分配策略。
4. 数组元素的地址计算与数组的存储方式有关。
5. 静态数组的存储空间可以在编译时确定。
6. 每个过程的活动记录的体积在编译时可静态确定。
7. 在程序中标识符的出现仅为使用性的。

第八章

1. 局部优化中使用的 DAG 图反映了基本块之间的关系。
2. 含有优化功能的编译程序执行效率高。

3. 进行代码优化时应着重考虑循环的代码优化，这对提高目标代码的效率将起更大作用。
4. 每个基本块可用一个 DAG 表示。
5. 每个基本块只有一个入口和一个出口。
6. 仅考虑一个基本块，不能确定一个赋值是否真是无用的。

第九章

1. 目标代码生成时，应考虑如何充分利用计算机的寄存器的问题。
2. 目标代码指令越丰富，代码生成的工作越复杂。
3. 窥孔优化只能在目标代码上进行，且只需扫描一遍目标代码即可。