1、请将以下C语言程序片段：

（1）翻译为三地址中间代码；

（2）运用优化方法进行优化。

int a = 5;

int b = 3;

int c = a + b

if (c > 7) {

c = c - 2;

} else {

c = c + 2;

}

int d = a \* c;

2、给定以下三地址中间代码：

i = 0

a = b \* c

L1: if i < 10 goto L2 else goto L3

L2: d = a + b

e = b \* c

f = e + a

g = f + d

h = g - a

i = i + 1

goto L1

L3:

1. 请使用有向无环图（DAG）表示这段代码；
2. 通过DAG图进行公共子表达式消除和复写传播等优化。

3、考虑以下的上下文无关文法，它描述了一个简单的算术表达式语言：

E -> E + T | T

T -> T \* F | F

F -> ( E ) | id

请为这个文法写出一个语法制导定义，用于计算这些表达式的值。假设`id`的值可以从符号表中查找。

4、考虑以下的上下文无关文法，它描述了一个支持整数、实数、数组和类型转换的简单编程语言：

S -> VarDecl S | Stmt S | ε

VarDecl -> Type id ;

Type -> int | real | array [ num ] of Type

Stmt -> id = Expr ; | if ( Expr ) Stmt | for ( Expr ; Expr ; Expr ) Stmt | print ( Expr ) ;

Expr -> Expr + Term | Term

Term -> Term \* Factor | Factor

Factor -> ( Expr ) | id | num | id [ Expr ] | ( Type ) Expr

在这个文法中，`id`代表标识符，`num`代表数字字面量。请为这个文法写出一个语法制导定义，用于进行类型检查。如果类型不匹配或者存在其他类型错误，应产生错误。假设所有变量在使用前都已经声明，并且有一个符号表用于跟踪变量的类型和值。

5、考虑以下三地址代码序列，它是一个编译器后端生成的中间表示形式：

a = b + c

d = a + e

f = d - b

g = f + c

h = g + d

i = h - f

假设我们有一个包含三个寄存器（R1, R2, R3）的目标机器。请为上述代码序列进行寄存器分配，以最小化内存访问次数。假设所有变量初始时都存储在内存中，并且在每条指令执行后，除非另有说明，否则变量的值不会保留在寄存器中。