北京大学信息科学技术学院考试试卷

考试科目： 编译技术 **姓名：** _____ **学号：** _____

考试时间： 2018 年 6 月 27 日 **任课教师：** 梁云

题号	一	二	三	四	五	六	七	八	**总分**
分数									
阅卷人									

装订线内

不要答题

北京大学考场纪律

1、考生进入考场后，按照监考老师安排隔位就座，将学生证放在桌面上。无学生证者不能参加考试；迟到超过 15 分钟不得入场。在考试开始 30 分钟后方可交卷出场。

2、除必要的文具和主考教师允许的工具书、参考书、计算器以外，其它所有物品（包括空白纸张、手机等）不得带入座位，已经带入考场的必须放在监考人员指定的位置，并关闭手机等一切电子设备。

3、考试使用的试题、答卷、草稿纸由监考人员统一发放，考试结束时收回，一律不准带出考场。若有试题印制问题请向监考教师提出，不得向其他考生询问。提前答完试卷，应举手示意请监考人员收卷后方可离开；交卷后不得在考场内逗留或在附近高声交谈。未交卷擅自离开考场，不得重新进入考场答卷。考试结束监考人员宣布收卷时，考生立即停止答卷，在座位上等待监考人员收卷清点后，方可离场。

4、考生要严格遵守考场规则，在规定时间内独立完成答卷。不准旁窥、交头接耳、打暗号，不准携带与考试内容相关的材料参加考试，不准抄袭或者有意让他人抄袭答题内容，不准接传答案或者试卷等。凡有严重违纪或作弊者，一经发现，当场取消其考试资格，并根据《北京大学本科考试工作与学习纪律管理规定》及其他相关规定严肃处理。

5、考生须确认自己填写的个人信息真实、准确，并承担信息填写错误带来的一切责任与后果。

学校倡议所有考生以北京大学学生的荣誉与诚信答卷，共同维护北京大学的学术声誉。

以下为试题和答题纸，共 21 页。

得分

一、选择题（每小题 2 分，共 20 分）

题号	1	2	3	4	5	6	7	8	9	10
答案										

1. 数据流分析通常分为前向分析和后向分析。那么下列哪一项优化技术的分析方向和其他的都不同？( )

(A) 常量传播　　(B) 可用表达式　　(C) 可达定义　　(D) 活跃变量

2. 下列说法正确的是？( )

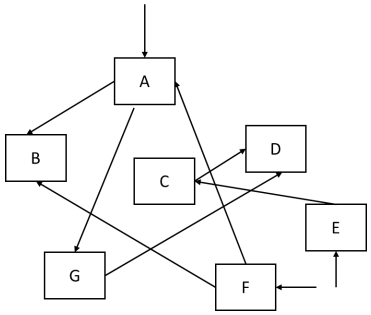
① 相比三元式，间接三元式在编译优化步骤中提供了便利

② 基于图染色的贪心算法总能为寄存器分配找到最优解

③ 在一个 DFA 中有且仅有唯一的一个终态

(A) ①②　　(B) ①③　　(C) ①　　(D) ②

3. 对于下面的引用关系图，删除哪些引用关系，会导致 D 被回收？( )



(A) F->A, C->D

(B) C->D, E->C

(C) A->G, G->D

(D) A->G, E->C

4. 考虑下面的程序：

```
…
procedure   p(x, y, z)：
begin
   y:=y+2;
   z:=z+x;
end
```

```
begin
   a:=5;
   p(a, a, a);
   print a；
end
```

试问，若参数传递的方式分别采用传地址和传值时，程序执行后输出 a 的值分别是：( )

(A) 5; 14
(B) 5; 12
(C) 14; 5
(D) 12; 5

5. 下面是一个用来求阶乘的 PASCAL 程序：

```
program main(a, b);
  var f, n:integer;
  function factor(n:integer):integer;
        begin
             if n=0 then factor:=1
             else factor:=n*factor(n-1)
        end;
  begin
        n:=6; f:=factor(n);write(f)
  end
```

请问程序调用 main(4, 4)之后，在第 3 次进入函数 factor 时的 factor(4)访问链指向( )

(A) main(4, 4)　　(B) main(1, 1)　　(C) factor(6)　　(D) factor(5)

6. 现有下列语法制导翻译（SDT）：

E->E1*T { E.val = E1.val *T.val } | T { E.val = T.val }
T->T1+i { T.val = T1.val +i.val } | i { T.val = i.val }

则句子 2*3+4*5+6 按照该 SDT 规约，其值为：( )

(A) 32
(B) 56
(C) 94
(D) 154

7. 使用自底向上的分析方法为下列 C 语言程序生成四元式目标代码，其中 A1、A2 和 A3 均是赋值语句。

```
if (a < b || c < d &&   e < f) {                    // E1
  A1;
}
else {
  A2;
}
while (a < b) {                                    // E2
  A3;
}
```

生成的中间代码片段如下，

```
100:   if a < b goto ___
101:   goto ___
102:   if c < d goto ___
103:   goto ___
104:   if e < f goto ___
105:   goto ___
106:   A1 对应的代码
...
116:   goto ___
117:   A2 对应的代码
...
127:   if a < b goto ___
128:   goto ___
129:   A3 对应的代码
...
139:   goto ___
140:   程序后续部分对应的代码
```

令条件语句中的布尔表达式为 E1；循环语句中的布尔表达式为 E2。请问 E1.truelist、E1.falselist、E2.truelist、E2.falselist 分别是？( )

(A) {100, 104}, {103, 105}, {127, 139}, {128}

(B) {100, 104}, {103, 105}, {127}, {128}

(C) {100, 104}, {105} , {127, 139} , {128}

(D) {100, 104}, {105}, {127}, {128}

8. 下列文法 G[S]

S→(T) | a+S | a

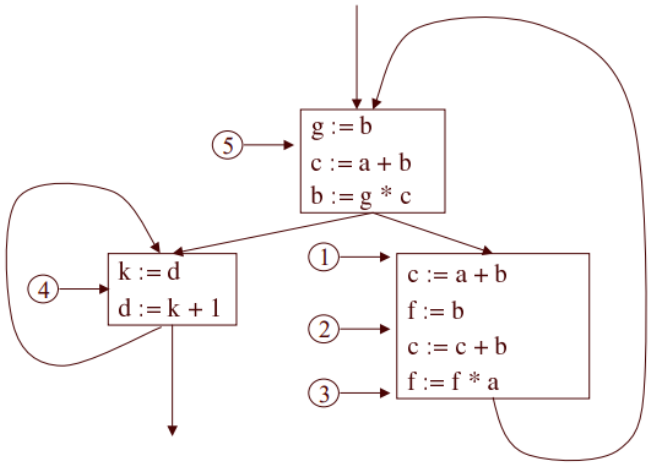
T→T,S | S

不可以推导出下列哪一个串？（ ）

(A) a+a+(a,a+a,a+a+(a,a))　　(B) (a,a)+a+(a+a+(a,a,a+a))

(C) (a,a,a+a)　　(D) a+((a),(a),(a),(a))

9. 在如下控制流图中，假设程序的出口处没有变量是活跃的，则图中标出的各处中，活跃变量数目最多的程序点是？( )



(A) ①　　(B)②　　(C) ④　　(D)⑤

10. 已知文法：

S→AB

S→bC

A→ ε

A→b

B→ ε

B→aD

C→AD

C→b

D→aS

D→c

不是 LL(1)的。请问在预测分析表中，哪一个选项对应的表项存在冲突？( )

(A) A 行 c 列

(B) A 行 a 列

(C) C 行 b 列

(D) C 行 c 列

得分

二、简答题 （30 分，每题 6 分）

1. 请画出一个 DFA，使其接收所有被 3 整除的正整数的二进制表示（不包括 0，可以有前缀 0），如 0011。

2. 将下面的表达式 (a[i] * b － c[i - j]) * 2 分别翻译成抽象语法树、四元式、间接三元式。

3. 现有如下的三地址中间代码，其中 t0 和 t6 分别在进入和离开本段程序时活跃。

```
t1 = t0 + 15
t2 = t0 * 3
t3 = 8
if t1 < t2
then
    t3 = t1 * t2
    t4 = t1 * 2
else
    t3 = t1 + t2
    t4 = t1 + t3
    t1 = 3
fi
t5 = t4 – t3
t6 = t5 + t1
```

请构造出变量(活跃范围)的干涉图（interference graph）。

4. 对于包含 4 字节宽整数 int、4 字节宽浮点数 float、数组类型、记录类型的一系列类型声明语句，可以使用如下的文法和 SDT 生成符号表。其中，BCDT 是非终结符，num 是一个数，'{'和'}'表示大括号这个字符本身。首先，请补全 SDT 中的动作（填写在<_____>内）。另外，type、t、width、w 这四个属性中，哪些是继承属性，哪些是综合属性？

1) D→T id; { top.push(id.lexeme, T.type, offset); // 添加符号
offset = <_____>; }
D1

2) D→ ε

3) T→B { C.t = <_____>;
C.w = <_____>; }
C { T.type = <_____>;
T.width = <_____>; }

4) T→record'{' { Stack.push(top); // 当前 top 入栈
Stack.push(offset); // 当前 offset 入栈
top = new Env(); // 创建新符号表
offset = <_____>; }
D'}' { T.type = record(top); // 创建记录类型
T.width = <_____>;
offset = Stack.pop(); // 栈顶赋给 offset
top = Stack.pop(); // 栈顶赋给 top }

5) B→int { B.type = integer;
B.width = <_____>; }

6) B→float { B.type = float;
B.width = <_____>; }

7) C→[num]C1 { C1.t = <_____>;
C1.w = <_____>;
C.type = array(num.value, C1.type); // 创建数组类型
C.width= <_____>; }

8) C→ ε { C.type = <_____>;
C.width = <_____>; }

5. 我们定义两个 for 循环 A、B 可以“完美合并”（假设 A 在 B 之前），记作 A~B，当且仅当下列条件都满足：1）A 和 B 之间没有其他语句；2）A 和 B 有相同的迭代次数； 3）B 循环体中所使用的变量的定义不会在 A 中。例如：

```
A: for i in (0, 10)：
        a = 5
    end for
B: for j in (2, 12)：
        c = j + 2
    end for
```

可以合并成

```
  for i in (0, 10)：
     a = 5
     c = i + 2 + 2
  end for
```

而在 AB 中插入任意一条语句，或者将 j 的范围改成（2，13），又或者将 B 的循环体改为“c = a + j” ，都会导致 A 和 B 不再能“完美合并”；但是，如果将 B 的循环体改为“a = 1　c = a + j”，那么 A 和 B 仍然可以完美合并，因为 B 内对变量 a 的使用的定义仍然在 B 的循环体内部。同时注意，“完美合并”不具有传递性，也就是说即使 (A~B)且(B~C)，也不一定能将 A、B、C 合并成一个循环，因为 A 中定义的变量可能在 C 中被使用。这时，按照先后顺序，将 A 和 B 合并成一个循环即可。现有下列用来识别简单程序的文法：

```
program    → program stat | ε
stat       → for_loop | assign
for_loop   → for idx_var in range : for_body end for
idx_var    → i | j
range      → (constant, constant)
for_body   → for_body assign | ε
assign     → other_var = cal_expr
cal_expr   → other_var expr
expr       → + constant | ε
other_var  → a | b | c | d | e
```

请在上述文法的基础上，设计一个语法制导定义（SDD），计算出被识别的程序在按照先后顺序（即循环 ABC 中，若 A~B，则优先合并 AB）合并完所有可以“完美合并”的 for 循环后，还剩下的 for 循环的数量。并请说明自定义的属性表示的含义。【提示：符号 constant 的数值可通过其 val 属性获得；SDD 中允许定义所需要的数据结构，如数组、集合、堆、栈、队列等，及其相应的操作】

得分

三、LR 分析 （10 分）

有增广文法如下：

S'→S

S→Ab

S→ABc

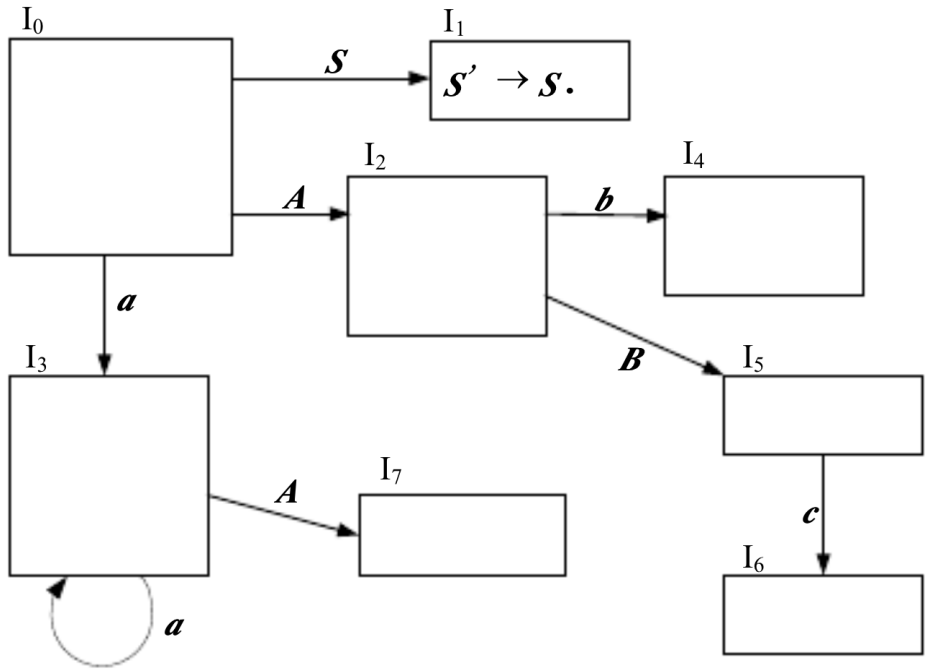
A→aA

A→a

B→b

对于这个文法，有 LR(0)自动机如下（部分内容被挖去）。

1） 补全 LR(0)自动机。



2） 指出 LR(0)自动机中，哪些状态分别出现了何种冲突。

3） 证明该文法是 SLR 的。

得分

四、SDD（10 分）

有如下 SDD：

产生式	语义规则
T->FT’	T’.inh = F.val T.val = T’.syn
T’->*FT_1’	T_1’.inh = T’.inh * F.val T’.syn = T_1’.syn
T’->ε	T’.syn = T’.inh
F->digit	F.val = digit.lexval

请画出 3*4 的注释语法分析树，并用虚线标注出求值顺序。

得分

五、中间代码分析 （10 分）

现有下列的三地址代码：

1) i = 1
2) j = 20
3) if j < 0 goto (11)
4) t1 = 10 / i
5) t2 = t1 + j
6) t3 = 8 * t2
7) t4 = 24 – t3
8) a[t4] = 10.2
9) j = j – 1
10) goto (4)
11) i = i + 1
12) if i <5 goto (2)
13) i = 2
14) t5 = i * 2
15) t6 = 12 / t5
16) a[t6] = 2.6
17) i = i + 4
18) if i < 20 goto (13)

(1) 请首先为这段代码构造出控制流图（CFG）。

(2) 请用基本块表示出控制流图中存在的所有循环。

(3) 已知控制流图的一个节点 d 支配节点 n，当且仅当从开始节点（可以理解为源）到节点 n 的每一条路径均要经过节点 d。据此，可以定义支配树：在支配树中，节点 d 的父节点是最临近的 d 且支配 d 的节点。以开始节点即为树根，画出该控制流图对应的支配树。

得分

六、SDT （10 分）

JSON 是一种通用的数据格式，在 JavaScript 等语言中得到了广泛的使用。假设 JSON 的一种简化格式如下：

- 每一个 JSON 对象的基本表示形式为：{ 基本项, 基本项, ……}；
- 每一个基本项的格式是：**”key” : “value”** ；
- 每个对象中可以包含零个或多个基本项，用逗号（,）隔开，每个基本项的后面都跟着一个逗号；
- 每个基本项中的第二个字符串（”value”）可以替换为一个嵌套的 JSON 对象。

一个简单的 JSON 对象的例子如下：

```
{
"programmer": { "firstName": "Brett", "lastName":"James", "Company": "Google" ,},
"author": { "firstName": "Isaac", "lastName": "Asimov", "genre": "fiction", },
"musician":   { "firstName": "Eric", "lastName": "Clapton", "instrument": "guitar", },
}
```

在实际使用中，经常需要在 JSON 和 XML 中进行互相转换。设基本的 XML 格式如下：

- 起始标签和结束标签必须成对出现（例如 <key> value </key> ）。
- 支持标签的嵌套，例如：

  <tag> <tag1> value1 </tag1> <tag2> value2 </tag2> </tag>
- 在每一层（包括最外层）都允许出现多个标签项；
- 可以忽略空格和换行的影响。

举例来说，上面例子中的 JSON 对象可以转换为如下的 XML 代码：

```
<programmer>      <firstName> Brett </firstName>
   <lastName> James </lastName>       <company> Google </company>
</programmer>
<author>       <firstName> Isaac </firstName>
   <lastName> Asimov </lastName>     <genre> fiction </genre>
```

```
</author>
<musician>    <firstName> Eric </firstName>
      <lastName> Clapton </lastName>    <instrument> guitar </instrument>
</musician>
```

利用编译技术构造一个转换程序，把 JSON 对象转换为更易读的 XML 格式。

1) 假设词法分析器会把包含引号的字符串识别为同一类 token。以词法分析识别出的 token 作为终结符号，给出表示上述 JSON 数据格式的适合 LR 分析的上下文无关文法，并简要说明其正确性。

2) 在上述文法的基础上，给出把上述 JSON 数据转换为相应的 XML 代码的语法制导的翻译方案（SDT）。【提示：需要为 token 定义相应的词法属性；在生成的 XML 代码中不用考虑缩进和空格。】

得分

七、代码优化 （10 分）

如果一条语句使用的变量总是未被初始化的，那么我们将这这条语句称为“僵尸语句”。移除一部分“僵尸语句”后，程序的其他部分也可能因此变成“僵尸语句”（比如这些语句使用的变量只在“僵尸语句”中被定义过）。

1） 请设计一个数据流分析算法来找到程序中的所有“僵尸语句”。被分析的程序只涉及赋值语句、判断语句和控制流语句（if_else 语句、while 语句、for 语句）。只需说明这个数据流算法的方向（前向、后向），状态的表示（用什么数据结构标识和判断是否为僵尸语句，如何初始化），状态转移函数（经过一条语句后，状态应如何改变），交汇操作（两个状态如何合并）。【注意：需要将 for 循环拆分为多个独立语句。 以 for (i = 0; i < 10; i++) 为例， 将 i = 0、 i < 10、 i++ 当作三个单独的语句。】

2） 请用你定义的数据流算法找出下列代码中的“僵尸语句”。对于每条“僵尸语句”，写出数据流算法每次处理该条语句时，你在第（1）问中定义的状态是如何改变的。

```
a = x;   // x 已经被定义过
b [10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
if ( a > 0 ) {
        c = c + 1;
        d = d +1;
} else {
        c = c + 2;
        d = 5;
}
b[5] = a + c;
b[d] = a – 2;
for ( ; i < 10; i++){
        e += b[i];
}
```