中 国 科 学 技 术 大 学

2004－2005 学年第二学期考试试卷

考试科目：编译原理和技术　　得分：_____

学生所在系：_____ 姓名：_____ 学号：_____

1.（15 分）

(a) 字母表Σ = { (,) }上的语言{ ( ), (( )( )), ((( ))), ( )( )( )( )( )}是不是正规语言？为什么？

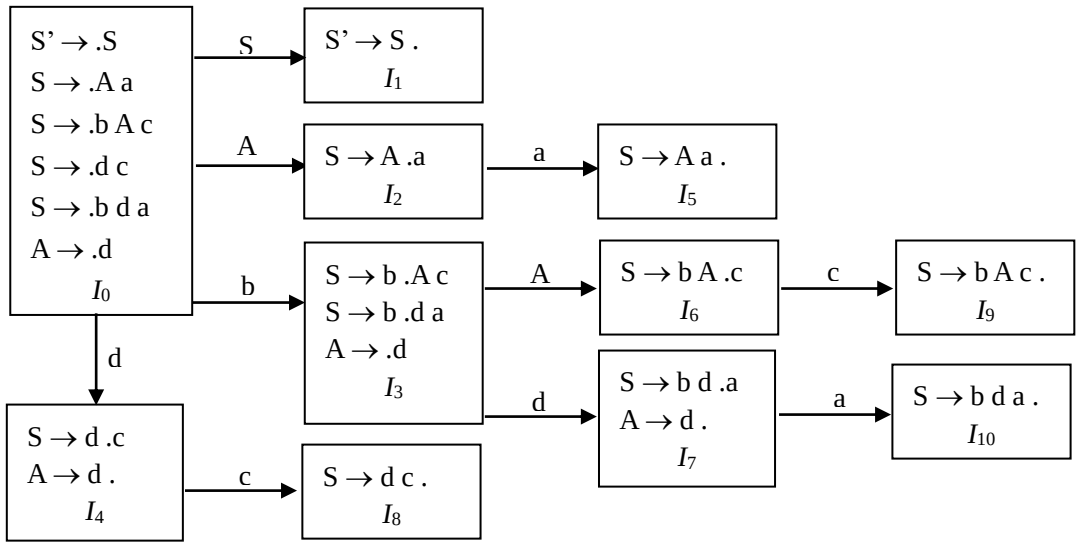
(b) 正规式 $(0 \mid 1)^*$ 和$((\varepsilon \mid 0)\, 1^*)^*$是否等价，说明理由。

2.（15 分）接受文法

$S \rightarrow A\ a \mid b\ A\ c \mid d\ c \mid b\ d\ a$

$A \rightarrow d$

活前缀的 DFA 见下图。请根据这个 DFA 来构造该文法的 SLR(1)分析表，并说明该文法为什么不是 SLR(1)文法。



3.（10 分）现有字母表Σ={ a }，写一个和正规式 a^*等价的上下文无关文法，要求所写的文法既不是 LR 文法，也不是二义文法。

4.（20 分）

(a) 下面的文法定义语言 $L = \{ a^n b^n c^m \mid m, n \geq 1\}$。写一个语法制导定义，其语义规则的作用是：对不属于语言 L 的子集 $L_1 = \{ a^n b^n c^n \mid n \geq 1\}$的句子，打印出错信息。

$S \rightarrow D\ C$　　$D \rightarrow a\ D\ b \mid a\ b$　　$C \rightarrow C\ c \mid c$

(b) 语句的文法如下：

S → **id** := E | **if** E **then** S | **while** E **do** S | **begin** S ; S **end** | **break**

写一个翻译方案，其语义动作的作用是：若发现 **break** 不是出现在循环语句中，及时报告错误。

5.（5 分）C 程序设计的教材上说，可以用两种形式表示字符串：其一是用字符数组存放一个字符串，另一种是用字符指针指向一个字符串。教材上同时介绍了这两种形式的很多共同点和不同点，但是有一种可能的区别没有介绍。下面是一个包含这两种形式的 C 程序：

```
char c1[]="good!";
char *c2="good!";

main()
{
    c1[0]= 'G';
    printf("c1=%s\n", c1);
    c2[0]= 'G';
    printf("c2=%s\n", c2);
}
```

该程序在 X86/Linux 机器上运行时的信息如下：

```
c1=Good!
Segmentation fault (core dumped)
```

请问，出现 Segmentation fault 的原因是什么？

6.（15 分）下面是一个 C 语言程序：

```
long f1( i )
long i;
{
    return(i*10);
}

long f2(long i)
{
    return(i*10);
}

main()
{
    printf("f1 = %d, f2 = %d\n", f1(10.0), f2(10.0) );
}
```

其中函数 f1 和 f2 仅形式参数的描述方式不一样。该程序在 X86/Linux 机器上的运行结果如下：

f1 = 0, f2 = 100

请解释为什么用同样的实在参数调用这两个函数的结果不一样。

7.（10 分）下面是一个 C 语言程序和在 X86/Linux 机器上编译（未使用优化）该程序得到的汇编代码（为便于理解，略去了和讨论本问题无关的部分，并改动了一个地方）。

(a) 为什么会出现一条指令前有多个标号的情况，如.L2 和.L4，还有.L5、.L3 和.L1？从控制流语句的中间代码结构加以解释。

(b) 每个函数都有这样的标号.L1，它的作用是什么，为什么本函数没有引用该标号的地方？

```
main()
{
    long i,j;

    if ( j )
        i++;
    else
        while ( i ) j++;
}
```

```
main:
    pushl %ebp          ——将老的基地址指针压栈
    movl %esp,%ebp      ——将当前栈顶指针作为基地址指针
    subl $8,%esp        ——为局部变量分配空间
    cmpl $0,-8(%ebp)
    je .L2
    incl -4(%ebp)
    jmp .L3
.L2:
.L4:
    cmpl $0,-4(%ebp)
    jne .L6
    jmp .L5
.L6:
    incl -8(%ebp)
    jmp .L4
.L5:
.L3:
.L1:
    leave               ——和下一条指令一起完成恢复老的基地址指针，将栈顶
    ret                 ——指针恢复到调用前参数压栈后的位置，并返回调用者
```

8.（5 分）cc 是 UNIX 系统上 C 语言编译命令，–l 是连接库函数的选择项。两个程序员分别编写了函数库 libuser1.a 和 libuser2.a。当用命令

cc test.c –luser1.a –luser2.a

编译时，报告有重复定义的符号。（备注：库名中的 lib 在命令中省略。该命令和命令 cc test.c libuser1.a libuser2.a 的效果是一致的）。而改用命令

cc test.c –luser2.a –luser1.a

时，能得到可执行程序。试分析原因。

9.（5 分）根据教材上所介绍的方法，C++中的对象声明语句应如何翻译成 C 语句？如教材图 11.11（旧教材的图 10.11）程序中的

Point _center;

应怎样翻译？

2005 年编译原理和技术试题参考答案

1． (a) 语言{ ( ), (( ) ( )), ((( ))), ( ) ( ) ( ) ( ) ( )}是正规语言，因为该语言只包括有限个句子，它可以用正规式定义如下：

( ) | (( )( )) | ((( ))) | ( ) ( ) ( ) ( ) ( )

(b) 我们分析正规式((ε | 0) 1^*)*表示的语言。可以看出正规式(ε | 0) 1^*描述的语言是集合{ε, 1, 11, 111, …, 0, 01, 011, 0111, …}，它含长度为 1 的两个串 0 和 1。那么，$(0 | 1)^*$ 所描述的语言是((ε | 0) 1^*)* 所描述的语言的子集，因为$(0 | 1)^*$ 表示字母表{0, 1}上所有串的集合，因此((ε | 0) 1^*)* 所描述的语言不可能再有更多的句子，因而也是表示字母表{0, 1}上所有串的集合。

2．分析表如下。因为有两处有移进－归约冲突，所以该文法不是 SLR(1)文法。

注：Fallow(A) = {a, c}

状态	动作					转移	
	a	b	c	d	$	*S*	*A*
0		*s*3		*s*4		1	2
1					*acc*		
2	s5						
3				*s7*			6
4	*r*5		*s*8, *r*5				
5					*r*1		
6			*s*9				
7	*s*10, *r*5						
8					*r*3		
9					*r*2		
10					*r*4		

3．满足条件的一个文法如下：

S → a S a | a | ε

4． (a) 语法制导的定义如下：

S → D C	if D.length ≠ C.length then print (“error”)
D → a b	D.length := 1
D → a D_1 b	D.length := D_1.length + 1
C → c	C.length := 1
C → C_1 c	C.length := C_1.length + 1

(b) 翻译方案如下：

S′ → {S.loop := false}S

S → **id** := E

S → **if** E **then** {S_1.loop := S.loop}S_1

S → **while** E **do** {S_1.loop := true}S_1

S → **begin** {S_1.loop := S.loop}S_1 ; {S_2.loop := S.loop}S_2 **end**

S → **break**{if not S.loop then print("error") }

5．c1 是字符数组，c1[]="good!"看成是对这个数组的元素逐个地赋值。c2 是字符指针，它所指向的"good!"看成是一个字符串常量，常量的值不允许修改，因此编译器把这个字符串常量分配在只读数据区。

当执行赋值 c2[0]= 'G'时，试图对只读数据区赋值，因此报告错误。

6．历史上，C 语言中有参函数定义的一般形式是：

类型标识符 函数名（形式参数列表）

形式参数声明

对于这种形式的声明，C 语言编译器是不做实在参数和形式参数的个数和类型是否一致的检查的。如本题中函数 f1 的声明是这种形式。

现在，ANSI 新标准允许使用另一种方法声明形式参数，即在函数名后的括号中列出形式参数的同时，声明形式参数的类型。本题中函数 f2 的声明是这种形式。C 语言编译器对于这种形式的函数声明是要进行实在参数和形式参数的个数和类型是否一致的检查的。

因此，在编译函数调用 f2(10.0)时，会把浮点数 10.0 转换成整数 10，并产生将整数 10 压栈的指令。因此函数调用 f2(10.0)的结果是 100。

而在编译函数调用 f1(10.0)时，会把浮点数 10.0 转换成双精度型（参见《编译原理习题精选》第 5.8 题），并产生将该双精度型数压栈的指令。双精度型数占 8 个字节。它低地址的 4 个字节看成整数时正好是 0。因此函数调用 f1(10.0)的结果是 0（参见 http://staff.ustc.edu.cn/~yiyun 上 2003 年编译原理试题第 7 题）。

7．(a) 条件语句和循环语句的中间代码结构如下：

if (E) then S_1 else S_2	while (E) do S
E 的代码	L4: E 的代码
假转 L2	真转 L6
S_1 的代码	转 L5
转 L3	L6: S 的代码
L2: S_2 的代码	转 L4
L3:	L5:

用这种代码结构，当 while 语句作为条件语句的 S_2 时，就会出现题目所给的这种情况。

(b) .L1 标号定义的入口是返回调用者时该执行的指令，在函数内部有 return 语句时就会跳转到.L1。

8．这是基于下面几点原因。

1．两个函数库 libuser1.a 和 libuser2.a 都定义了某个函数或某个置初值的外部变量，把它简称为 a。

2．test.c 引用 a。

3．test.c 还引用 libuser2.a 的其它某个函数或外部变量，把它简称为 b。在 libuser2.a 中，a 和 b 在同一个目标文件中。

4．在 libuser1.a 中，含 a 的那个目标文件不含 b。

进一步的解释如下。

由于 test.c 引用 a 和 b，用第二种次序连接时，a 和 b 的定义在 libuser2.a 都能找到，所以不会再把 libuser1.a 中含 a 定义的目标文件连接进来。而用第一种次序连接时，先把 libuser1.a 中含 a 定义的目标文件连接进来，然后还需要把 libuser2.a 中含 b 定义的目标文件连接进来，引起把 a 的定义也要带进来，因为在 libuser2.a 中 a 和 b 的定义在同一个目标文件中。由此造成要连接 a 的两个定义。

9．C++语言中类的对象声明不加翻译就成了 C 语言中相应结构类型的变量声明，不管对象声明出现在程序中的什么地方。

备注： 由于 C++语言中一个类的所有非静态属性构成一个 C 语言的结构类型，并且类的名字就作为结构类型的名字，因此上面的语句不做任何修改，_center 自然就成了 Point 结构类型的变量。所要注意的是，还应该根据 Point 类中构造函数的参数置初值的情况，来给_center 适当地静态置初值。