2003 年编译原理试题

1.（20 分）写出字母表Σ = {a, b}上语言 L = {*w* | *w* 中 a 的个数是偶数}的正规式，并画出接受该语言的最简 DFA。

2.（15 分）考虑下面的表达式文法，它包括数组访问、加和赋值：

E → E[E] | E + E | E = E | (E) | **id**

该文法是二义的。请写一个接受同样语言的 LR(1)文法，其优先级从高到低依次是数组访问、加和赋值，并且加运算是左结合，赋值是右结合。

3.（10 分）下面是产生字母表Σ = { 0, 1, 2}上数字串的一个文法：

S → D S D | 2

D → 0 | 1

写一个语法制导定义，它打印一个句子是否为回文数（一个数字串，从左向右读和从右向左读都一样时，称它为回文数）。

4.（10 分）教材上 7.2.1 节的翻译方案

P → *D*	{*offset* := 0}
D → *D* ; *D*	
D → **id** : *T*	{ *enter*(**id**.*name*, *T.type*, *offset*); *offset* := *offset* + *T.width* }
T → **integer**	{*T.type* := *integer*; *T.width* := 4 }
T→ **real**	{*T.type* := *real*; *T.width* := 8 }

使用了变量 *offset*。请重写该翻译方案，它完成同样的事情，但只使用文法符号的属性，而不使用变量。

5.（5 分）一个 C 语言程序如下：

```
void fun(struct {int x; double r;} val) {     }

main()
{
    struct {int x; double r;} val;

    fun(val);
}
```

该程序在 X86/Linux 机器上的用 cc 命令编译时，报告的错误信息如下：

```
1: warning: structure defined inside parms
1: warning: anonymous struct declared inside parameter list
1: warning: its scope is only this definition or declaration,
1: warning: which is probably not what you want.
7: incompatible type for argument 1 of ‘fun’
```

请问，报告最后一行的错误的原因是什么？如何修改程序，使得编译时不再出现这个错误信息。

6.（10 分）一个 C 语言程序如下：

```
typedef struct _a{
        short   i;
        short   j;
        short   k;
}a;

typedef struct _b{
        long    i;
        short   k;
}b;

main()
{
        printf("Size of short, long, a and b = %d,%d,%d,%d\n",
                        sizeof(short), sizeof(long),sizeof(a),sizeof(b));
}
```

该程序在 X86/Linux 机器上的运行结果如下：

```
Size of short, long, a and b = 2, 4, 6, 8
```

已知 short 类型和 long 类型分别对齐到 2 的倍数和 4 的倍数。试问，为什么类型 b 的 size 会等于 8？

7.（15 分）一个 C 语言程序如下：

```
int fact(i)
int i;
{
    if(i==0)
        return 1;
    else
        return i*fact(i-1);
}

main()
{
    printf("%d\n", fact(5));
    printf("%d\n", fact(5,10,15));
    printf("%d\n", fact(5.0));
    printf("%d\n", fact());
}
```

该程序在 X86/Linux 机器上的运行结果如下：

```
120
```

120
1
Segmentation fault (core dumped)

请解释下面问题：

- 第二个 fact 调用：结果为什么没有受参数过多的影响？
- 第三个 fact 调用：为什么用浮点数 5.0 作为参数时结果变成 1?
- 第四个 fact 调用：为什么没有提供参数时会出现 Segmentation fault？

8.（5 分）C 语言的赋值操作并非仅对简单类型而言，例如若有类型声明 long a[100], b[100];，则赋值 a=b 是允许的。同样，若 a 和 b 是同一类型的两个结构，则赋值 a=b 也是允许的。

用教材上第七章所给出的三地址语句，我们能否为这种赋值产生中间代码？若你持肯定态度，请你给出对应这种赋值的中间代码序列；否则请你为这种赋值设计一种三地址语句。你所选用或设计的三地址语句要便于目标代码的生成。

9.（5 分）一个 C 程序的三个文件的内容如下：

head.h:

```
short int a = 10;
```

file1.c:

```
#include "head.h"

main()
{
}
```

file2.c:

```
#include "head.h"
```

在 X86/Linux 机器上的编译命令如下：

```
cc file1.c file2.c
```

编译结果报错的主要信息如下：

```
multiple definition of ‘a’
```

试分析为什么会报这样的错误。

10.（5 分）按照教材上介绍的方法，把下面 C++语言的函数翻译成 C 的函数。

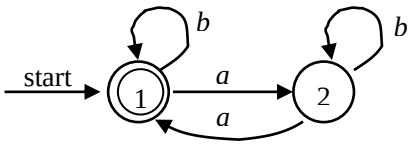
```
void zoom (GraphicalObj &obj, double zoom_factor, Point &center) {
    obj.translate (–center.x, –center.y);      // 将中心点移至原点(0, 0)
    obj.scale (zoom_factor);                   // 缩放
}
```

2003 年编译原理试题参考答案

1．语言 L 的正规式是：

$(a\ b^*a \mid b)^*$ 或 $b^*(a\ b^*a\ b^*)^*$

接受该语言的最简 DFA 是：



2. $E \rightarrow T = E \mid T$

$T \rightarrow T + F \mid F$

$F \rightarrow F[E] \mid (E) \mid \mathbf{id}$

3.

$S' \rightarrow S$	print(S.val)
$S \rightarrow D_1\ S_1\ D_2$	S.val = $(D_1.val = D_2.val)$ and $S_1.val$
$S \rightarrow 2$	S.val = true
$D \rightarrow 0$	D.val = 0
$D \rightarrow 1$	D.val = 1

4. 文法符号 *D* 的属性 *offset*1 是继承属性，代表在分析 *D* 前原来使用的变量 *offset* 的大小；属性 *offset*2 是综合属性，代表在分析 *D* 后原来使用的变量 *offset* 的大小。*P* 的属性 *offset* 是综合属性，记录该过程所分配的空间。

P → {*D.offset*1 := 0} *D* {*P.offset* := *D.offset*2 }

D → {D_1.*offset*1 := *D.offset*1 }D_1 ;
{D_2.*offset*1 := D_1.*offset*2 }D_2 {*D.offset*2 := D_2.*offset*2 }

D → **id** : *T* {*enter* (**id**.*name*, *T.type*, *D.offset*1);
*D.offset*2 := *D.offset*1 + *T.width* }

T → **integer** {*T.type* := *integer*; *T.width* := 4 }

T→ **real** {*T.type* := *real*; *T.width* := 8 }

5．C 语言对所有的类型都采用结构等价，唯有结构类型例外，采用名字等价。这里的类型不相容是因为两个 val 不是名字等价的。

要消除这个错误，包括所有的警告，程序修改如下：

```
struct s{
    int x;
    double r;
};
```

```
void fun(struct s val) {      }

main()
{
    struct s val;

    fun(val);
}
```

6．一个数组的 size 等于数组元素的 size 乘以数组元素的个数，这是一个原则。

对于结构类型 b 来说，它的一个变量只需 6 个字节就够了。如果声明结构类型 b 的一个数组，有两种可能：

（a）结构类型 b 的 size 是 6，数组元素之间空两个字节，以保证每个数组元素的地址都是 4 的倍数（因为第一个域 i 的类型是 long，要求对齐到 4 的倍数）。

（b）结构类型 b 的 size 是 8，以保证它作为数组元素的类型时，数组元素之间不用再考虑对齐问题。

方法（a）违反了我们一开始提到的原则，因此只能取方法（b）。

7．（1）参数表达式逆序计算并进栈，fact 能够取到第一个参数。

（2）参数 5.0 转换成双精度数进栈，占 8 个字节。它低地址的 4 个字节看成整数时正好是 0。

（3）由于没有提供参数，fact 把老 ebp（控制链）（main 的活动记录中保存的 ebp）当成参数，它一定是一个很大的整数，使得活动记录栈溢出。

8．教材上的中间代码有形如 x := y 的复写语句，用它就可以了。但是生成目标代码时，必须考虑 x 和 y 的类型。若 x 和 y 不是简单类型，则应该根据它们类型的 size，产生一连串的值传送指令。

9．由于 file1.c 和 file2.c 两个文件都包含文件 head.h，而 head.h 中有 short int a = 10，因此该程序有两个 a 的强符号定义。

10．按照教材上介绍的方法，函数的翻译结果如下：

```
void zoom (GraphicalObj &obj, double zoom_factor, Point &center) {
    obj.vptr[0] (obj, –center.x, –center.y);    // 将中心点移至原点(0, 0)
    obj.vptr[1] (obj, zoom_factor);             // 缩放
}
```