

## 第一、二题:

4.9

(a). 引入 L 的属性 b 记录 <sup>L 的指数</sup>

$S \rightarrow L_1 L_2 \quad \{ S.val = L_1.val + L_2.val / L_2.b; \}$   
 $S \rightarrow L \quad \{ S.val = L.val; \}$   
 $L \rightarrow L_1 B \quad \{ L.val = L_1.val \times 2 + B.val; L.b = L_1.b \times 2; \}$   
 $L \rightarrow B \quad \{ L.val = B.val; L.b = 2; \}$   
 $B \rightarrow 0 \quad \{ B.val = 0; \}$   
 $B \rightarrow 1 \quad \{ B.val = 1; \}$

(b)  $S \rightarrow L_1 L_2 \quad \{ L_1.i = 2^0; L_2.i = 2^1; S.val = L_1.val + L_2.val \}$

$S \rightarrow L \quad \{ L.i = 2^0; S.val = L.val \}$

$L \rightarrow L_1 B \quad \{ \text{if } L.i \geq 1: \}$

$B.i = L.i; L.i = L.i \times 2; \quad \# \text{ 考虑小数点前后}$

else:  $\# \text{ 两种模式}$

$L.i = L.i; L.S = \frac{1}{2} \times L.i; B.i = L_1.S;$

$L.val = L_1.val + B.C; \}$

$L \rightarrow B \quad \{ B.i = L.i; L.S = L.i \times \frac{1}{2}; L.val = B.C \}$

$B \rightarrow 0 \quad \{ B.C = 0 \}$

$B \rightarrow 1 \quad \{ B.C = B.i \}$

4.12

(a) 初始 语法制导的翻译

$S' \rightarrow S \quad \{ S.d = 0; \}$

$S \rightarrow (L) \quad \{ L.d = S.d + 1; \}$

$S \rightarrow a \quad \{ \text{print}(S.d); \}$

$L \rightarrow L_1 S \quad \{ L_1.d = L.d; S.d = L.d; \}$

$L \rightarrow S \quad \{ S.d = L.d; \}$

(b) d 属性标记该符号第一个符号所在位置 len 属性表示长度

$S' \rightarrow S \quad \{ S.d = 1; \}$

$S \rightarrow (L) \quad \{ L.d = S.d + 1; S.len = L.len + 2; \}$

$S \rightarrow a \quad \{ \text{print}(S.d); S.len = 1; \}$

$L \rightarrow L_1 S \quad \{ L_1.d = L.d; S.d = L.d + L_1.len + 1; L.len = L_1.len + 1 \}$

$L \rightarrow S \quad \{ S.d = L.d; L.len = S.len; \} \quad \# S.len + 1$

$S' \rightarrow M \quad \{ S.d = M.d; \}$

S

$S \rightarrow ( \quad \{ N.i = S.d; \}$

N  $\{ L.d = N.S \}$

L)

$S \rightarrow \quad \{ \text{print}(S.d); \} \quad \text{print}(\text{stack}[\text{top}].val);$

a

$L \rightarrow \quad \{ L.d = L.d; \}$

L,  $\{ P.i = L.d; \}$

P  $\{ S.d = P.S \}$

S

$L \rightarrow \quad \{ S.d = L.d; \}$

S

$M \rightarrow \varepsilon \quad \{ M.d = 0 \} \quad \text{stack}[\text{top}+1].val = 0$

$N \rightarrow \varepsilon \quad \{ N.S = N.i + 1 \} \quad \text{stack}[\text{top}+1].val =$   
 $\text{stack}[\text{top}-1].val + 1$

$P \rightarrow \varepsilon \quad \{ P.S = P.i \} \quad \text{stack}[\text{top}+1].val =$   
 $\text{stack}[\text{top}-2].val$

$s' \rightarrow M \quad \{s.d = M.s;\}$   
 $S$   
 $S \rightarrow ( \quad \{N.i = S.d;\}$   
 $N \quad \{L.d = N.s;\}$   
 $L) \quad \{S.len = L.len + 2;\} \quad \text{stack}[top-3].b = \text{stack}[top-1].b + 2$   
 $S \rightarrow \quad \{print(S.d);\} \quad \text{print}(\text{stack}[top].a);$   
 $a \quad \{S.len = 1;\} \quad \text{stack}[top].b = 1;$   
 $L \rightarrow \quad \{L.d = L.d;\}$   
 $L_1, \quad \{P.i = L.d + L.len + 1;\}$   
 $P \quad \{S.d = P.s;\}$   
 $S \quad \{L.len = L_1.len + S.len + 1;\} \quad \text{stack}[top-3].b = \text{stack}[top-3].b + \text{stack}[top].b + 1$   
 $L \rightarrow \quad \{S.d = L.d;\}$   
 $S \quad \{L.len = S.len;\}$   
 $\checkmark M \rightarrow \epsilon \quad \{M.s = 1;\} \quad \text{stack}[top+1].a = 1$   
 $\checkmark N \rightarrow \epsilon \quad \{N.s = N.i + 1;\} \quad \text{stack}[top+1].a$   
 $\quad \quad \quad \text{stack}[top-1].a + 1$   
 $P \rightarrow \epsilon \quad \{P.s = P.i;\} \quad \text{stack}[top+1].a =$   
 $\quad \quad \quad \text{stack}[top-2].a + \text{stack}[top-2].b + 1$   
 用 a 存 d    b 存 len

### 第三题:

(3)	2、临时数据
1、临时数据	局部数据
局部数据	保存的机器状态
保存的机器状态	访问链
访问链	控制链
控制链	参数 (包含 { 参数个数 参数 })
无参过程	返回值使用寄存器传递

```

1 case LEA: //指令格式(LEA,1,a) //其中“取地址”指令 LEA 用来获取名字变量在“运行时栈-
  stack”上“地址偏移”
2     stack[++top] = base(stack, b, i.1) + i.a;
3     break;
4 case LODA: //指令格式(LODA,0,0) //而“间接读”指令 LODA 则表示以当前栈顶单元的内容
  为“地址偏移”来读
5     //取相应单元的值,并将该值存储到原先的栈顶单元中。
6     stack[top] = stack[stack[top]];
7     break;
8 case STOA: //(STOA,0,0) //而“间接写”指令 STOA 则将位
9     //于栈顶单元的内容,存入到次栈顶单元内容所代表的栈单元里,然后弹出栈顶和次栈顶
10    stack[stack[top-1]] = stack[top];
11    top-=2;
12    break;
  
```

1. a: array(10,pointer(int))  
p: pointer(array(10,array(10,pointer(int))))
2. 分别占用1,1,10,10,1个int大小空间, 共23个, 偏移量为0, 1, 2, 12, 22
3.  $(*p)[1][0][1][0]$   
 $p[0][1][0][1][0]$
- 4.

无参过程	返回值得使用寄存器传递	参数	
			$p[0][1][0][1][0]$ $**(*(*(*p+1)+1))$
0 i	i=100: (LIT, 0,100)	stack[top]=100	LOD, 0,22    s[top] = p
2 a[0]	(STO, 0,0)	i = s[top]	LODA, 0, 0    s[top] = *p
...	a[i]=q: (LOD, 0,1)	s[top] = q	ADD, 0,1    s[top] = *p+1
11 a[1]	(STO, 0,3)	a[i] = s[top]	LODA, 0,0
12 b[0]	p = &b (LEA, 0,12)	s[top] = &b	LOD A, 0,0    s[top] = **(*p+1)
21 b[1]	(STO, 0,22)	p = s[top]	ADD, 0,1    s[top] = **(*p+1)+1
22 p			LODA, 0,0
			LODA, 0,0    s[top] = **(*(*p+1)+1)
			call cout

5. r: LOD,0,3  
r: LOD,0,3  
\*r: LOD,0,3  
LODA,0,0

## runtime练习:

1. 输出:36313032 2016
2. 汇编代码如下

```

1      .section .rodata
2      .LC0:
3          .ascii "%x %s\n"
4      .text
5      .globl main
6      .type main,@function
7      main:
8          pushl   %ebp
9          movl    %esp, %ebp
10         subl    $40, %esp
11         andl    $-16, %esp #此行在内下三行应该是负责对其用的
12         movl    $0, %eax
13         subl    %eax, %esp
14         movb    $50, -24(%ebp)
15
16         movb    $48, -23(%ebp)
17         movb    $49, -22(%ebp)
18         movb    $54, -21(%ebp)
19         movb    $0, -20(%ebp)
20         leal    -24(%ebp), %eax
21
22         movl    %eax, -28(%ebp)
23
24         subl    $-4,
25         pushl   -28(%ebp)

```

```

26     pushl    -24(%ebp)
27
28
29     pushl    $.LC0
30     call     printf
31     addq     $16, %esp
32     movl     $0, %eax
33     leave
34     ret

```

二

汇编代码:

1. N=2:

```

1      .file "test1.c"
2      .text
3      .globl f
4      .type f,@function
5  f:
6      pushl   %ebp
7      movl    %esp, %ebp
8      movl    $100, 8(%ebp)
9      movl    $16 , 12(%ebp)
10     movb    $65 , 17(%ebp)
11     movl    20(%ebp), %eax
12     pushl    12(%eax)
13     pushl    8(%eax)
14     pushl    4(%eax)
15     pushl    (%eax)
16     call     f
17     addl    $16, %esp
18     leave
19     ret
20 //当 N=2 时, 生成的汇编代码片段。

```

2. N=11

```

1      file "test1.c"
2      .text
3      .globl f
4      .type f,@function
5  f:
6      pushl   %ebp
7      movl    %esp, %ebp
8      pushl   %edi
9      pushl   %esi
10     movl    $100, 8(%ebp)
11     movl    $24, 12(%ebp)
12     movb    $65, 17(%ebp)
13     subl    $8, %esp
14     movl    28(%ebp), %eax
15     subl    $24, %esp
16     movl    %esp, %edi
17     movl    %eax, %esi
18     cld
19     movl    $24, %eax #确认大小
20     movl    %eax, %ecx
21     rep

```

```

22     movl
23     call f
24     addl $32, %esp
25     leal -8(%ebp), %esp
26     popl %esi
27     popl %edi
28     leave
29     ret

```

3. addl 作用是清除给函数传参的空间，先前这16个空间正好对应给函数传递的参数，函数调用时产生的局部空间会自己删除，所以只需考虑删除传参分配的空间

leal的作用是移动esp到合适的位置，恢复函数调用时借用的edi和esi寄存器，恢复现场。

4. 编译器根据结构体大小，将结构体中的元素依次pushl到栈中，然后调用函数

### 三

1. line元素值为1,2,3,4,5,6,7,8,9,10

```

1     .file "p.c"
2     .text
3     .globl main
4     .type main,@function
5 main:
6     pushl %ebp
7     movl %esp, %ebp
8     subl $72, %esp
9     andl $-16, %esp
10    movl $0, %eax
11    subl %eax, %esp
12    leal -56(%ebp), %eax #留给数组的
13    movl %eax, -64(%ebp) #指针p
14    movl $0, -60(%ebp) #i
15 .L2:
16    cmpl $9, -56(%ebp)
17    jle .L5 #继续循环
18    jmp .L3
19 .L5:
20    movl -64(%ebp), %edx
21    movl -60(%ebp), %eax
22    movl %eax, (%edx)
23    subl $12, %esp
24    leal -64(%ebp), %eax
25    pushl %eax
26    call g
27    add $16, %esp
28    leal -60(%ebp), %eax
29    incl (%eax)
30    jmp .L2
31 .L3:
32    movl $0, %eax
33    leave
34    ret
35 .globl g
36 .type g,@function
37 g:
38    pushl %ebp
39    movl %esp, %ebp
40    movl 8(%ebp), %eax #传参的位置
41    movl (%eax), %eax

```

```

42     addl $1, (%eax) #(**p)++
43     movl 8(%ebp), %eax
44     addl $1, (%eax)
45     leave
46     ret

```

#### 四

```

1  main:
2      pushl %ebp
3      movl %esp, %ebp
4      subl $24, %esp
5      andl $-16, %esp
6      movl $0, %eax
7      subl %eax, %esp
8      movl $0, -20(%ebp)
9      movl $0, -16(%ebp)
10     movl $1, -12(%ebp)
11     movl $2, -12(%ebp)
12     movl $3, -8(%ebp)
13     movl $0, %eax
14     leave
15     ret

```

分配变量时考虑作用域，离开作用域后该变量空间即被废弃，可以被分配新值

#### 五

```

1  .LC0:
2      .long 0
3      .long 1
4      .long 2
5      .long 3
6      .long 4
7      .long 5
8  .LC1:
9      .string "%d\n"
10     .text
11     .globl main
12     .type main,@function
13  main:
14     pushl %ebp
15     movl %esp, %ebp
16     pushl %edi
17     pushl %esi
18     subl $48, %esp
19     andl $-16, %esp
20     movl $0, %eax
21     subl %eax, %esp
22     leal -40(%ebp), %edi
23     movl $.LC0, %esi
24     cld
25     movl $6, %eax
26     movl %eax, %ecx
27     rep
28     movsl
29     movl $6, -44(%ebp)
30     movl $7, -48(%ebp)
31     #数据初始化完成

```

```

32     leal -40(%ebp), %eax
33     addl $24, %eax
34     movl %eax, -52(%ebp)
35     subl $8, %esp
36     movl -52(%ebp), %eax
37     subl $4, %eax
38     pushl %eax
39     pushl $.LC1
40     call printf
41     addl $16, %esp # -8 还有两个pushl
42     movl $0, %eax
43     leal -8(%ebp), %esp
44     popl %esi
45     popl %edi
46     leave
47     ret

```

波浪线处：分配空间，初始化数组，使用的是题目二中提到的rep和movsl 数据传输指令。数组元素的初值一开始在.LC0 大概率是静态数据区，需要拷贝到程序运行的地方。

## 六:

结果为11。local嵌套深度最大的定义是在f(11,local)函数里，所以之后的local会返回11。  
level=10为止，arg（）的访问链均指向上一层，之后都指向level=10的一层