



第十二章 PL/0 代码生成

广东工业大学计算机学院



PL/O编译程序的目标代码结构

- ❖ PL/O编译程序所产生的目标代码是一个假想栈式计算机的汇编语言,可称为类PCODE指令代码,它不依赖任何实际计算机
- ❖ 此类栈式机没有累加器和通用寄存器,有一个栈式存储器,有四个控制寄存器(指令寄存器 I,指令地址寄存器 P,栈顶寄存器 T和基址寄存器 B),算逻运算都在栈顶进行。(这部分内容涉及到PL/O解释程序)





- ❖ 1. 栈顶寄存器(指针)T:由于每个过程当它被运行时, 给它分配的数据空间(下边称数据段)可分成两部分。
 - 静态部分:包括变量存放区和三个联系单元(SL、DL 和RA)。
 - 动态部分:作为临时工作单元和累加器用。需要时随时分配,用完后立即释放。
- ❖ 2. 基址寄存器(指针)B: 指向每个过程被调用时,在数据区S中给它分配的数据段起始地址,也称基地址。(第10章的SP)
- ❖ 3. 程序地址寄存器 P: 指向下一条要执行的指令
- ❖ 4. 指令寄存器 I: 存放正在执行的指令





PL/O编译程序的目标代码结构

❖ 指令格式

f 1 a

- f功能码
- Ⅰ 层次差(标识符引用层减去定义层)
- a 根据不同的指令有所区别 八条目标指令:





PL/O编译程序的目标代码结构和代码生成

- ❖ 指令 "LIT O A"
 - 立即数存入栈顶,即置T 所指存储单元的值为A
 - T加1
- ❖ LOD:将变量放到栈顶。a域为变量在所说明层中的相对位置,Ⅰ为调用层与说明层的层差值。
- ❖ STO:将栈顶的内容送入某变量单元中。a和I域的含意同LOD指令。
- ❖ CAL: 调用过程的指令。a为被调用过程的目标程序入口地址,Ⅰ为层差。
- ❖ INT: 为被调用的过程(或主程序)在运行栈中开辟数据区。a域为开辟的单元个数。
- ❖ JMP: 无条件转移指令, a为转向地址。
- ❖ JPC:条件转移指令,当栈顶的布尔值为非真时,转向a域的地址, 否则顺序执行。
- ❖ OPR: 关系运算和算术运算指令。将栈顶和次栈顶的内容进行运算, 结果存放在次栈顶。此外还可以是读写等特殊功能的指令,具体操 作由a域值给出。(P23)



- ❖ 编译程序的目标代码生成
 - 在分析程序体时生成的
 - 当分析程序体中的每个语句时,当语法正确则调用目标代码生成过程以生成与PL/O语句等价功能的目标代码,直到编译正常结束。除了过程说明部分外,变量和常量的说明都不产生目标代码。
 - 由过程GEN完成,GEN有三个参数,分别代表目标代码的功能码,层差和位移量 P417
 - 例如: gen(opr,0,16) 从命令行读入一个输入到栈 顶





gen(sto,lev-level,adr)

- 将栈顶内容送到变量单元中,
- ❖ lev: 当前处理的过程层次
- ❖ level:被引用变量或过程所在层次
- ❖ adr: 变量在过程中的相对位置(即第9章讲的DX)
- ❖ CX: 为目标代码code数组的下标指针(CX为指令的指针,由O开始顺序增加。)





* 相关定义

```
Code: array[0..cxmax] of instruction; p414
fct=(lit, opr, lod, sto, cal, int, jmp, jpc) P413
Instruction= packec record P413
```

f: fct;

I: 0..levmax

a: 0..amax;

end;

❖ CODE为一维数组,数组元素为记录型数据。每一个记录就是一条目标指令。





- ❖ 实际上目标代码的顺序是内层过程的排在前边,主程序 的目标代码在最后。
- ❖ 下面我们给出一个PL/O源程序和对应的目标程序的清单。



```
jmp 0 8
                              转向主程序入口
                       jmp 0 2
                              转向过程p入口
                               程p入口,为过程p开辟空间
                      int 0 3
       a=10;
const
                      1od 1 3
       b, c;
                       lit 0 10
                      opr 0 2
procedure p;
                       sto 1 4
  begin
                      opr 0 0
    c:=b+a;
                    8)
                      int 0 5 主程序入口开辟5个栈空间
                    9) opr 0 16 从命
  end;
                       sto 0 3
begin
                       lod 0 3
                              将变量b的值取至栈顶
                              将常数值0进栈
                      lit 0 0
                   (12)
  read(b);
                              次栈顶与栈顶是否不等
                       opr 0 9
  while b#0 do
                      jpc 0 24 等时转(24)(条件不满足转)
                   (15) cal 0 2 调用过程p
  begin
                   (16) lit 0 2 常数值2进栈
    call p;
                      lod 0 4 将变量c的值取至栈顶
    write (2*c);
                   (18) opr 0 4 次栈顶与栈顶相乘(2*c)
                       opr 0 14 栈顶值输出至屏幕
    read(b);
                   (20)
                       opr 0 15 换行
  end
                       opr 0 16 从命令行读取值到栈顶
                   (22)
                              栈顶值送变量b中
end.
                       sto 0 3
                       jmp 0 11 无条件转到循环入口(11)
                   (23)
                              结束退栈
                       opr 0 0
                                                 10
```



- ❖ 对于过程(分程序)的代码生成 在block入口处生成一条(jmp,0,0)指令,作为过程体入口指令, 该指令的第3区域的'0'需分析到过程体入口时才返填。
- ❖ 对分程序体入口的处理(见程序文本P424页block 的过程体) begin (*block*)

dx:=3; tx0:=tx;

(*保留当前table表指针值,实际为过程名在table表中的位置*)

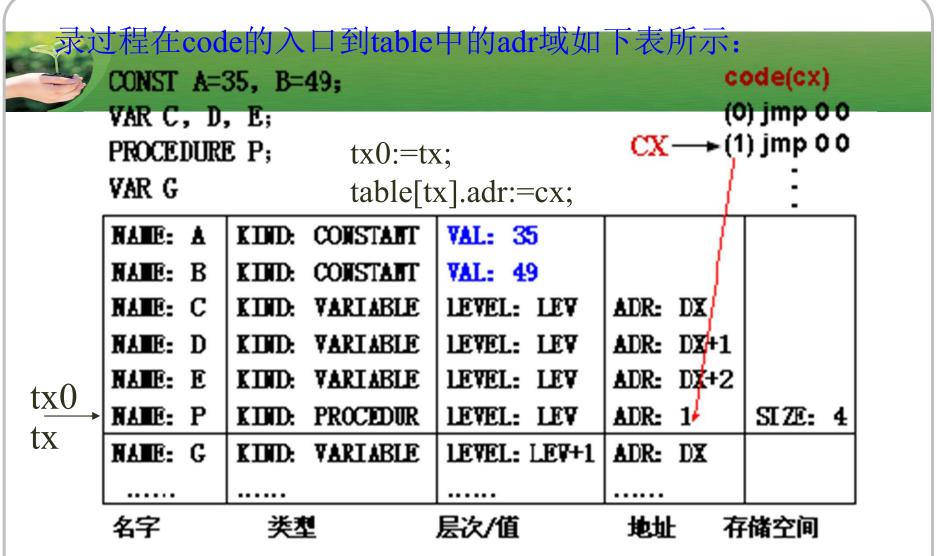
table[tx].adr:=cx;(*保留当前code指针值到过程名的adr域*)

gen(jmp,0,0);

(上例的前二行!)

❖ 其实上述内容最重要的是地址的回填(见下几页PPT)





(*生成转向过程体入口的指令,该指令的地址为cx已保留在过程名的adr域,真正的过程体入口地址,等生成过程体入口的指令时,将过程体入口返填(jmp,0,0)的第3区域,同时填到table[tx0].adr中*)

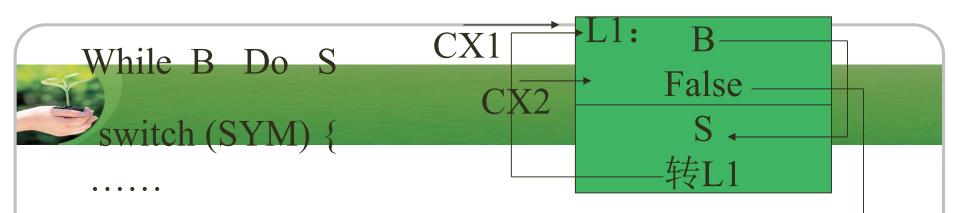


code[table[tx0].adr].a:=cx;

```
(0) jmp 0
     CONST A=35, B=49;
过程的入口地址填
                                                (1) jmp 0
     VAR C, D, E;
                       写在code和table中
     PROCEDURE P;
                             table[tx0].adr=cx (cx) int
     VAR G
                              VAL: 35
      NAME: A
               KIND: CONSTANT
      NAME: B
               KIND: CONSTANT
                              VAL: 49
                              LEVEL: LEV
                                           ADR: DX
      NAME: C
               KIND: VARIABLE
                              LEVEL: LEV
                                           ADR: DX+1
               KIND: VARIABLE
      NAME: D
                                           ADR: DX-2
      NAME: E
               KIND: VARIABLE
                              LEVEL: LEV
tx0
               KIND: PROCEDUR
                              LEVEL: LEV
                                           ADR: 1 cx
      NAME: P
                                                     SIZE: 4
      NAME: G
               KIND: VARIABLE
                              LEVEL: LEV+1
                                           ADR: DX
tx
               . . . . . .
                                  层次/值
                                                     存储空间
       名字
                    类型
                                             地址
```

过程体入口时的处理 table表格管理

```
code[table[tx0].adr].a:=cx;(cx为过程入口地址,填写在code中)
with table[tx0] do
begin
adr:=cx;(table[tx0].adr=cx)
size:=dx;(table[tx0].size=dx)
end;
请特别注意dx、tx、cx的作用和如何处理信息之间的连接关系。
```



CASE WHILESYM:

L2:

CX1=CX; GetSym(); //保留L1的地址

CONDITION(SymSetAdd(DOSYM,FSYS),LEV,TX);

//生成B的代码

CX2=CX; GEN(JPC,0,0);//条件跳转, L2待定,

if (SYM==DOSYM) GetSym();

else Error(18);

STATEMENT(FSYS,LEV,TX);//生成S的代码

GEN(JMP,0,CX1); //无条件跳转, 转到L1

CODE[CX2].A=CX; //回填L2

break;



- ❖ 附录
- ❖ 当源程序经过语法分析,如果未发现错误时,由编译程序调用解释程序,对目标代码开始进行解释执行。
- ❖ PL/0解释程序



PL/O解释程序

PL/0解释程序的存储分配

- ❖存储区:
 - ❖数组CODE存放目标程序
 - ❖运行时的数据区S
- ❖S是由解释程序定义的一维整型数组
- ❖由于PL/0语言的目标程序是一种假想的栈式计算机的汇编语言,现仍用Pascal语言解释执行





PL/O解释程序

- ❖ 解释程序定义了4个寄存器:
- ❖①I 指令寄存器:存放当前正在解释的一条目标指令
- ❖ ②P 程序地址寄存器:指向下一条要执行的目标程序的
- ❖ 地址(相当目标程序CODE数组的下标)



3T 栈顶寄存器。指向当前栈中最新分配的单元(T也是

数组S的下标)。

每个过程当它被调用时,给它分配的数据空间(下边称数据段)可分成两部分:

- ▶静态部分:包括变量存放区和三个联系单元
- ▶动态部分:作为临时工作单元和累加器用。需要时随时分配,用完后立即释放
- ④B 基址寄存器。指向当前执行过程的在数据区S中 给该过程分配的数据段起始地址,称基地址



当过程被调用时,在栈顶分配三个联系单元,这三个联

系单元存放的内容分别为:

- ①SL 静态链: 指向定义该过程的直接外过程(或主程序)) 运行时最新数据段的基地址
- ②DL 动态链: 指向调用该过程前正在运行过程的数据段 基地址
- ③RA 返回地址:记录调用该过程目标程序的断点,即当时的程序地址寄存器P的值。也就是调用过程指令的下一条指令的地址



❖PL/0编译程序给每个过程定义的变量在数据段内分配的相对位置是从3开始顺序增加。前面的三个单元为上面指出的联系单元

具体的过程调用和结束,对上述寄存器及3个联系单元的

填写和恢复由下列目标指令完成

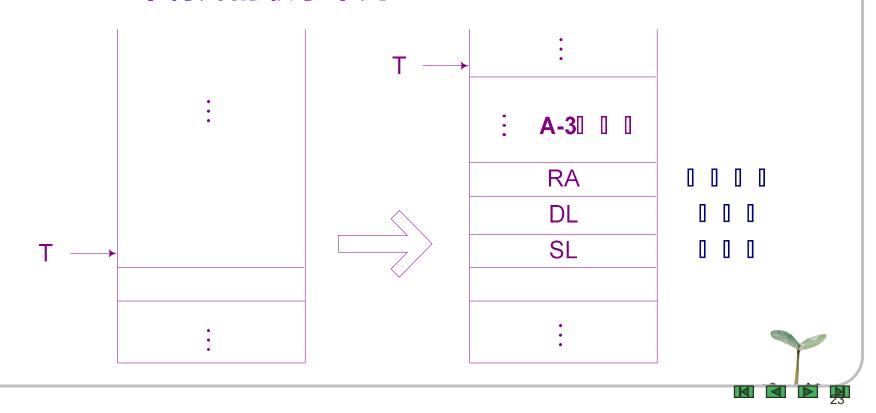
- (1) **INT 0 A**
- (2) OPR 00
- (3) CALLA





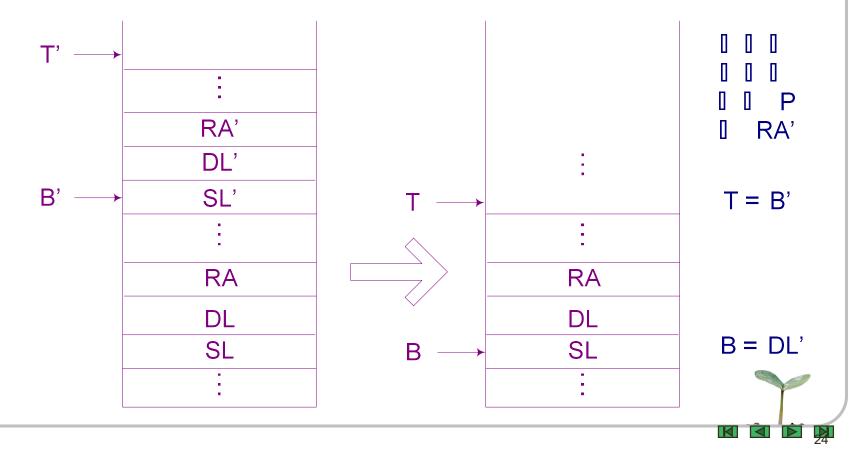
PL/O编译程序的目标代码结构

- - 在栈顶开辟 A 个存储单元, 服务于被调用的过程
 - A 等于该过程的局部变量数加 3
 - 3 **个特殊的联系单元**



令指令 "OPR 0 0"

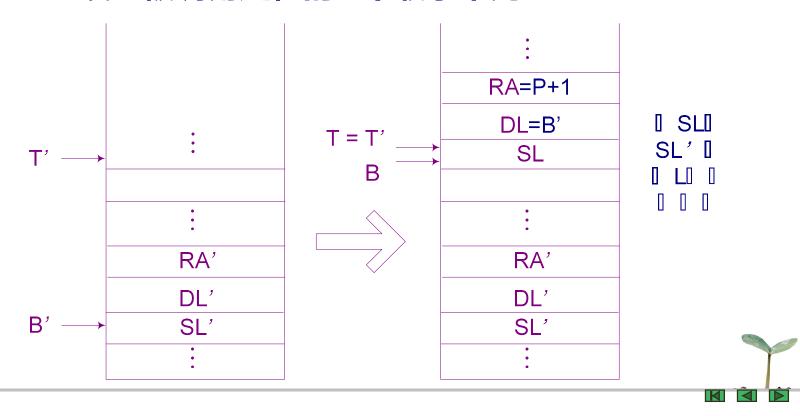
- 过程调用结束后,返回调用点并退栈
- 重置基址寄存器和栈顶寄存器



業P-code 虚拟机

◆指令 "CAL L A"

- 调用地址为 A 的过程 (置指令地址寄存器为A)
- L为调用过程与被调用过程的层差
- 设置被调用过程的3个联系单元



- 指令 "LIT 0 A"
 - 立即数存入栈顶,即置T所指存储单元的值为A
 - _ T加1
- ◆ 指令 "LOD L A"
 - 将层差为L、偏移量为A的存储单元的值取到栈顶
 - _ T加1
- ◆ 指令 "STO L A"
 - T減1
 - 将栈顶的值存入层差为L、偏移量为A的存储单元
- → 注:层差为L、偏移量为A的存储单元,即沿当前层静态链SL开始向前第L层的SL作为基址,加上A,即为该单元的地址



- ◆指令 "OPR 0 1"
 - 求栈顶元素的相反数,结果值留在栈顶
- ◆指令 "OPR 0 6"
 - 栈顶元素的奇偶判断,若为奇数,结果为1;若为偶数,结果为0;结果值留在栈顶



- 指令 "OPR 0 2"
 - 次栈顶与栈顶的值相加,结果存入次栈顶
 - T減1
- ◆ 指令 "OPR 0 3"
 - 次栈顶的值减去栈顶的值,结果存入次栈顶
 - T減1
- ◆指令 "OPR 0 4"
 - 次栈顶的值乘以栈顶的值,结果存入次栈顶
 - T減1
- ◆指令 "OPR 0 5"
 - 次栈顶的值除以栈顶的值,结果存入次栈顶
 - T減1



类P-code虚拟机

- ↑ 指令 "OPR 0 8" 比较次栈顶与栈顶是否相等
 - 若相等, 结果为0; 存结果至次栈顶; T 减1
- ◆ 指令 "OPR 0 9" 比较次栈顶与栈顶是否不相等
 - 若不相等,结果为0;存结果至次栈顶; T减1
- ◆ 指令 "OPR 0 10" 比较次栈顶是否小于栈顶
 - 若小于, 结果为0; 存结果至次栈顶; T 减1
- 令 指令 "OPR 0 11" 比较次栈顶是否大于等于栈顶
 - 若大于等于,结果为0;存结果至次栈顶; □减1
- ◆ 指令 "OPR 0 12" 比较次栈顶是否大于栈顶
 - 若大于,结果为0;存结果至次栈顶; T减1
- ◆ 指令 "OPR 0 13" 比较次栈顶是否小于等于栈顶
 - 若小于等于,结果为0;存结果至次栈顶; \ 减1



- ◆指令 "JMP 0 A"
 - 无条件转移至地址 A, 即置指令地址寄存器为A
- ◆指令 "JPC 0 A"
 - 条件转移指令
 - 若栈顶为 ○,则转移至地址 A,即置指令地址寄存器为A



- ◆ 指令 "OPR 0 14"
 - 栈顶的值输出至控制台屏幕
 - T 减 1
- ◆ 指令 "OPR 0 15"
 - 控制台屏幕输出一个换行
- ◆ 指令 "OPR 0 16"
 - 从控制台读入一行输入,置入栈顶
 - _ T加1





- ◆ 类P-code 解释程序
 - 数据结构

```
运行栈 int s[stacksize]
指令寄存器 struct instruction
{
    enum fct f; /*操作码*/
    int l; /*引用层与声明层的层差*/
    int a; /*因不同的f各异*/
} i
指令地址寄存器 int p;
基址寄存器 int b;
栈顶寄存器 int t;
虚拟机代码段 struct instruction code[cxmax];
```



♦ 类P-code 解释程序

- 处理流程

- (1) 初始化 p=b=t=0; s[0]=s[1]=s[2]=0;
 - (2) **取指令到指令寄存器** i=code[p]; p++;
 - (3) 分析并解释执行指令 :
 - (4) 若程序未结束 (p != 0) , 转 (2)
 - (5) 返回

(0) jmp 0 8	转向主程序入口	, two-t
O (1) jmp 0 2	转向过程p入口	
(2) int 0 3	过程p入口,为过程p开辟空间	
(3) lod 1 3	取变量b的值到栈顶	const a=10;
(4) lit 0 10	取常数10到栈顶	
(5) opr 0 2	次栈顶与栈顶相加	var b,c;
(6) sto 1 4	栈顶值送变量c中	procedure p;
(7) opr 0 0	退栈并返回调用点(16)	begin
(8) int 0 5	主程序入口开辟5个栈空间	
(9) opr 0 16	从命令行读入值置于栈顶	c:=b+a;
(10) sto 0 3	将栈顶值存入变量b中	end;
(11) lod 0 3	将变量b的值取至栈顶	begin
(12) lit 0 0	将常数值0进栈	
(13) opr 0 9	次栈顶与栈顶是否不等	read(b);
(14) jpc 0 24	相等时转(24)(条件不满足转)	while b#0 do
(15) cal 0 2	调用过程p	begin
(16) lit 0 2	常数值2进栈	
(17) lod 0 4	将变量c的值取至栈顶	call p;
(18) opr 0 4	次栈顶与栈顶相乘(2*c)	write(2*c);
(19) opr 0 14	栈顶值输出至屏幕	read(b);
(20) opr 0 15	————————————————————————————————————	
(21) opr 0 16	ンンカークーフライン	
(22) sto 0 3	栈顶值送变量b中	end.
(23) jmp 0 11		
(24) opr 0 0	结束退栈	34

	(0) jmp 0 8	转向主程序入口	, total
	(1) jmp 0 2 (2) int 0 3	转向过程p入口 过程p入口,为过程p开辟空间	
	(3) lod 1 3 (4) lit 0 10	取变量b的值到栈顶 取常数10到栈顶	const a=10; var b,c;
	(5) opr 0 2 (6) sto 1 4 (7) opr 0 0	次栈顶与栈顶相加 栈顶值送变量c中 退栈并返回调用点(16)	procedure p;
\longrightarrow	(8) int 0 5 (9) opr 0 16	主程序入口开辟5个栈空间 从命令行读入值置于栈顶	begin c:=b+a;
	(10) sto 0 3 (11) lod 0 3 (12) lit 0 0	将栈顶值存入变量b中 将变量b的值取至栈顶 将常数值0进栈	end; begin
	(13) opr 0 9 (14) jpc 0 24	次栈顶与栈顶是否不等 相等时转(24)(条件不满足转)	read(b); while b#0 do
	(15) cal 0 2 (16) lit 0 2 (17) lod 0 4	调用过程p 常数值2进栈 将变量c的值取至栈顶	begin call p;
	(18) opr 0 4 (19) opr 0 14	次栈顶与栈顶相乘(2*c) 栈顶值输出至屏幕	write(2*c); read(b);
	(20) opr 0 15 (21) opr 0 16	换行 从命令行读取值到栈顶 b	t end
	(22) sto 0 3 (23) jmp 0 11 (24) opr 0 0	栈顶值送变量b中 _无条件转到循环入口(11) 结束退栈	end.









	(0) jmp 0 8	转向主程序入口		, Prof
	(1) jmp 0 2	转向过程p入口		♦ 19 1
	(2) int 0 3	过程p入口,为过程p开辟空间		., ,
	(3) lod 1 3	取变量b的值到栈顶		const a=10;
	(4) lit 0 10	取常数10到栈顶		var b,c;
	(5) opr 0 2	次栈顶与栈顶相加		
	(6) sto 14	栈顶值送变量c中	←	procedure p;
	(7) opr 0 0	退栈并返回调用点(16)		t begin
	(8) int 0 5	主程序入口开辟5个栈空间	0	c:=b+a;
	(9) opr 0 16	从命令行读入值置于栈顶	_	·
	(10) sto 0 3	将栈顶值存入变量b中	5	end;
	(11) lod 0 3	将变量b的值取至栈顶		begin
	(12) lit 0 0	将常数值0进栈		read(b);
1	(13) opr 0 9	次栈顶与栈顶是否不等	5	while b#0 do
	(14) jpc 0 24 (15) cal 0 2	相等时转(24)(条件不满足转) 调用过程p		
	(16) lit 0 2	桐用は性P 常数值2进栈	0	begin
	(17) lod 0 4	将变量c的值取至栈顶		call p;
	(17) lod 0 4 (18) opr 0 4	次栈顶与栈顶相乘(2*c)	0	write(2*c);
	(19) opr 0 14	栈顶值输出至屏幕		
	(20) opr 0 15	掐 行	→ 0	read(b);
	(21) opr 0 16	从命令行读取值到栈顶	b	end
	(22) sto 0 3			end.
	(23) jmp 0 11			Silai
	(24) opr 0 0	结束退栈		40



	(0) jmp 0 8	转向主程序入口			. /Til
	(1) jmp 0 2	转向过程p入口			♦ 19]
	(2) int 0 3	过程p入口,为过程p开辟空间			
	(3) lod 1 3	取变量b的值到栈顶			const a=10;
	(4) lit 0 10	取常数10到栈顶			
	(5) opr 0 2	次栈顶与栈顶相加			var b,c;
	(6) sto 1 4	栈顶值送变量c中			procedure p;
	(7) opr 0 0	退栈并返回调用点(16)			begin
	(8) int 0 5	主程序入口开辟5个栈空间			c:=b+a;
	(9) opr 0 16	从命令行读入值置于栈顶			· · · · · · · · · · · · · · · · · · ·
	(10) sto 0 3	将栈顶值存入变量b中		\leftarrow	end;
	(11) lod 0 3	将变量b的值取至栈顶		t	begin
	(12) lit 0 0	将常数值0进栈			read(b);
	(13) opr 0 9	次栈顶与栈顶是否不等	5		` '
	(14) jpc 0 24	相等时转(24)(条件不满足转)	3		while b#0 do
	(15) cal 0 2	调用过程p	0		begin
р	(16) lit 0 2	常数值2进栈			call p;
	(17) lod 0 4	将变量c的值取至栈顶	0		•
	(18) opr 0 4	次栈顶与栈顶相乘(2*c)			write(2*c);
	(19) opr 0 14	栈顶值输出至屏幕 ***	0		read(b);
	(20) opr 0 15 (21) opr 0 16	换行 以会会污渍取信到线顶	b		end
	(21) opr 0 16 (22) sto 0 3	从命令行读取值到栈顶 栈顶值送变量b中		,	end.
	(22) \$10 0 3 (23) jmp 0 11				ciiu.
	(24) opr 0 0				
	(24) opi 0 0	治术 些化			12





(0) jmp 0 8	转向主程序入口		, June 1
(1) jmp 0 2 (2) int 0 3	转向过程p入口 过程p入口,为过程p开辟空间		♦ 19 1
(3) lod 1 3	取变量b的值到栈顶		const a=10;
→ (4) lit 0 10 p (5) opr 0 2	取常数10到栈顶 次栈顶与栈顶相加	5	var b,c;
(6) sto 14	栈顶值送变量c中	16	procedure p;
(7) opr 0 0 (8) int 0 5	退栈并返回调用点(16) 主程序入口开辟5个栈空间	0	begin c:=b+a;
(9) opr 0 16 (10) sto 0 3	从命令行读入值置于栈顶 将栈顶值存入变量b中	0	end;
(11) lod 0 3 (12) lit 0 0	将变量b的值取至栈顶 将常数值0进栈	b	begin
(13) opr 0 9 (14) jpc 0 24	次栈顶与栈顶是否不等 相等时转(24)(条件不满足转)	5	read(b); while b#0 do
(15) cal 0 2 (16) lit 0 2	调用过程p 常数值2进栈	0	begin
(17) lod 0 4 (18) opr 0 4	将变量c的值取至栈顶 次栈顶与栈顶相乘(2*c)	0	call p; write(2*c);
(19) opr 0 14	栈顶值输出至屏幕	0	read(b);
(20) opr 0 15 (21) opr 0 16	换行 从命令行读取值到栈顶		end
(22) sto 0 3	栈顶值送变量b中		end.
(23) jmp 0 11 (24) opr 0 0			

(0) jmp 0 8	转向主程序入口				A Tril
(1) jmp 0 2	转向过程p入口				♦ [5]
(2) int 0 3	过程p入口,为过程p开辟空间				
(3) lod 1 3	取变量b的值到栈顶 取常数10到栈顶			t	const a=10;
(4) lit 0 10 → (5) opr 0 2	双吊致 10到 185页 次栈顶与栈顶相加		10	τ	var b,c;
(6) sto 14	栈顶值送变量c中				procedure p;
(7) opr 0 0	退栈并返回调用点(16)		5		
(8) int 0 5	主程序入口开辟5个栈空间		16		begin
(9) opr 0 16	从命令行读入值置于栈顶		10		c:=b+a;
(10) sto 0 3	将栈顶值存入变量b中		0		end;
(11) lod 0 3	将变量b的值取至栈顶				begin
(12) lit 0 0	将常数值0进栈	→	0		read(b);
(13) opr 0 9	次栈顶与栈顶是否不等	b			
(14) jpc 0 24 (15) cal 0 2	相等时转(24)(条件不满足转)				while b#0 do
(16) lit 0 2	调用过程p 常数值2进栈		5		begin
(17) lod 0 4	将变量c的值取至栈顶				call p;
(18) opr 0 4	次栈顶与栈顶相乘(2*c)		0		write(2*c);
(19) opr 0 14	栈顶值输出至屏幕				read(b);
(20) opr 0 15	换行		0		
(21) opr 0 16	从命令行读取值到栈顶		0		end
(22) sto 0 3	栈顶值送变量b中		U		end.
(23) jmp 0 11					
(24) opr 0 0	结束退栈				40

(0) jmp 0 8	转向主程序入口			, January
(1) jmp 0 2	转向过程p入口			♦ 191
(2) int 0 3 (3) lod 1 3	过程p入口,为过程p开辟空间 取变量b的值到栈顶			
(4) lit 0 10	取常数10到栈顶			const a=10;
(5) opr 0 2	次栈顶与栈顶相加			var b,c;
→ (6) sto 1 4 p (7) opr 0 0	栈顶值送变量c中 退栈并返回调用点(16)		15	t procedure p; begin
(8) int 0 5 (9) opr 0 16	主程序入口开辟5个栈空间 从命令行读入值置于栈顶		16	c:=b+a;
(10) sto 0 3	将栈顶值存入变量b中		0	end;
(11) lod 0 3 (12) lit 0 0	将变量b的值取至栈顶 将常数值0进栈		0	begin read(b);
(13) opr 0 9 (14) jpc 0 24	次栈顶与栈顶是否不等 相等时转(24)(条件不满足转)	b		while b#0 do
(15) cal 0 2 (16) lit 0 2	调用过程p 常数值2进栈		5	begin
(17) lod 0 4	将变量c的值取至栈顶		<u> </u>	call p;
(18) opr 0 4	次栈顶与栈顶相乘(2*c)		0	write(2*c);
(19) opr 0 14 (20) opr 0 15	栈顶值输出至屏幕 换行		0	read(b);
(21) opr 0 16	从命令行读取值到栈顶		0	end
(22) sto 0 3 (23) jmp 0 11				end.
(24) opr 0 0	结束退栈			

(0) jmp 0 8 转向主程序入口 ,		, Prof
(1) jmp 0 2 转向过程p入口		♦ [5]
(2) int 0 3 过程p入口,为过程p开辟空间		
(3) lod 1 3 取变量b的值到栈顶		const a=10;
(4) lit 0 10 取常数10到栈顶		·
(5) opr 0 2 次栈顶与栈顶相加		var b,c;
(6) sto 1 4 栈顶值送变量c中		procedure p;
→ (7) opr 0 0 退栈并返回调用点(16)		t begin
P (8) int 0 5 主程序入口开辟5个栈空间	16	c:=b+a;
(9) opr 0 16 从命令行读入值置于栈顶		·
(10) sto 0 3 将栈顶值存入变量b中	0	end;
(11) lod 0 3 将变量b的值取至栈顶		begin
(12) lit 0 0 将常数值0进栈 ————————————————————————————————————	0	read(b);
(13) opr 0 9 次栈顶与栈顶是否不等 b	4 =	
(14) jpc 0 24 相等时转(24) (条件不满足转)	15	while b#0 do
(15) cal 0 2	5	begin
(16) lit 0 2 常数值2进栈	3	call p;
(17) lod 0 4 将变量c的值取至栈顶 (48) opr 0 4 为 为技术 医 医 技术	0	-
(18) opr 0 4 次栈顶与栈顶相乘(2*c)		write(2*c);
(19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行	0	read(b);
(20) opr 0 15		end
(21) Opi 0 10	0	end.
(22) \$10 0 3		GIIG.
(24) opr 0 0 结束退栈		M A DI
(2-7) OPI 0 0 5日		

(0) jmp 0 8	转向主程序入口		, Prof
(1) jmp 0 2 (2) int 0 3 (3) lod 1 3 (4) lit 0 10 (5) opr 0 2 (6) sto 1 4 (7) opr 0 0 (8) int 0 5 (9) opr 0 16 (10) sto 0 3 (11) lod 0 3 (12) lit 0 0 (13) opr 0 9	转向过程p入口 过程p入口,为过程p开辟空间 取变量b的值到栈顶 取常数10到栈顶 次栈顶与栈顶相加 栈顶值送变量c中 退栈并返回调用点(16) 主程序入口开辟5个栈空间 从命令行读入值置于栈顶 将栈顶值存入变量b中 将变量b的值取至栈顶 将常数值0进栈 次栈顶与栈顶是否不等	t	const a=10; var b,c; procedure p; begin c:=b+a; end; begin read(b);
(14) jpc 0 24 (15) cal 0 2 → (16) lit 0 2	相等时转(24)(条件不满足转) 调用过程p 常数值2进栈	15 5	while b#0 do begin call p;
(17) lod 0 4 (18) opr 0 4	将变量c的值取至栈顶 次栈顶与栈顶相乘(2*c)	0	write(2*c);
(19) opr 0 14 (20) opr 0 15 (21) opr 0 16 (22) sto 0 3 (23) jmp 0 11	栈顶值输出至屏幕 换行 从命令行读取值到栈顶 栈顶值送变量b中 无条件转到循环入口(11)	0 0	read(b); end end.
(24) opr 0 0	结束退栈		

(0) jmp 0 8	转向主程序入口		, Prof
(1) jmp 0 2	转向过程p入口		♦ 19 1
(2) int 0 3	过程p入口,为过程p开辟空间		
(3) lod 1 3	取变量b的值到栈顶		const a=10;
(4) lit 0 10	取常数10到栈顶		·
(5) opr 0 2	次栈顶与栈顶相加		var b,c;
(6) sto 1 4	栈顶值送变量c中		procedure p;
(7) opr 0 0	退栈并返回调用点(16)		begin
(8) int 0 5	主程序入口开辟5个栈空间		c:=b+a;
(9) opr 0 16	从命令行读入值置于栈顶		· · · · · · · · · · · · · · · · · · ·
(10) sto 0 3	将栈顶值存入变量b中		end;
(11) lod 0 3	将变量b的值取至栈顶		→ begin
(12) lit 0 0	将常数值0进栈	2	read(b);
(13) opr 0 9	次栈顶与栈顶是否不等	4 -	` '
(14) jpc 0 24	相等时转(24)(条件不满足转)	15	while b#0 do
(15) cal 0 2	调用过程p	_	begin
(16) lit 0 2	常数值2进栈	5	call p;
$\longrightarrow (17) \log 0 4$	将变量c的值取至栈顶	0	•
(18) opr 0 4	次栈顶与栈顶相乘(2*c)	U	write(2*c);
(19) opr 0 14	栈顶值输出至屏幕 ************************************	0	read(b);
(20) opr 0 15	换行 以 企 么怎法即使到投票	0	end
(21) opr 0 16	从命令行读取值到栈顶	\rightarrow 0	
(22) sto 0 3		· / •	end.
(23) jmp 0 11	プレスポートマネッル日ントノントロ(・ ・ /		
(24) opr 0 0	结束退栈		50

			No. of Concession, Name of Street, or other Persons, Name of Street, or ot
(0) jmp 0 8	转向主程序入口		4 /Til
(1) jmp 0 2	转向过程p入口		♦ 19 1
(2) int 0 3	过程p入口,为过程p开辟空间		
(3) lod 1 3	取变量b的值到栈顶		const a=10;
(4) lit 0 10	取常数10到栈顶		var b,c;
(5) opr 0 2 (6) sto 1 4	次栈顶与栈顶相加 栈顶值送变量c中		
(6) St0 1 4 (7) opr 0 0	退栈并返回调用点(16)		procedure p;
(7) opi 0 0 (8) int 0 5	主程序入口开辟5个栈空间		begin
(9) opr 0 16	从命令行读入值置于栈顶		← c:=b+a;
(10) sto 0 3	将栈顶值存入变量b中	15	t end;
(11) lod 0 3	将变量b的值取至栈顶	15	begin
(12) lit 0 0	将常数值0进栈	2	
(13) opr 0 9	次栈顶与栈顶是否不等		read(b);
(14) jpc 0 24	相等时转(24)(条件不满足转)	15	while b#0 do
(15) cal 0 2	调用过程p		begin
(16) lit 0 2	常数值2进栈	5	call p;
(17) lod 0 4 → (18) opr 0 4	将变量c的值取至栈顶 次栈顶与栈顶相乘(2*c)	0	write(2*c);
(18) opr 0 4 (19) opr 0 14	枝顶値輸出至屏幕		
(20) opr 0 15	换行	0	read(b);
(21) opr 0 16	从命令行读取值到栈顶		end
(22) sto 0 3	 	→ 0	end.
(23) jmp 0 11	无条件转到循环入口(11)	כ ב	
(24) opr 0 0	结束退栈		

(0) jmp 0 8	转向主程序入口		, Prof
(1) jmp 0 2	转向过程p入口		♦ 19 1
(2) int 0 3	过程p入口,为过程p开辟空间		
(3) lod 1 3	取变量b的值到栈顶		const a=10;
(4) lit 0 10	取常数10到栈顶		·
(5) opr 0 2	次栈顶与栈顶相加		var b,c;
(6) sto 1 4	栈顶值送变量c中		procedure p;
(7) opr 0 0	退栈并返回调用点(16)		begin
(8) int 0 5	主程序入口开辟5个栈空间		c:=b+a;
(9) opr 0 16	从命令行读入值置于栈顶		·
(10) sto 0 3	将栈顶值存入变量b中		end;
(11) lod 0 3	将变量b的值取至栈顶		`† begin
(12) lit 0 0	将常数值0进栈	30	read(b);
(13) opr 0 9	次栈顶与栈顶是否不等	4.5	` '
(14) jpc 0 24	相等时转(24)(条件不满足转)	15	while b#0 do
(15) cal 0 2	调用过程p	5	begin
(16) lit 0 2	常数值2进栈	5	call p;
(17) lod 0 4	将变量c的值取至栈顶	0	• 1
(18) opr 0 4	次栈顶与栈顶相乘(2*c) 栈顶值输出至屏幕		write(2*c);
(19) opr 0 14 (20) opr 0 15	投资通制山主併养 换行	0	read(b);
(21) opr 0 16	从命令行读取值到栈顶		end
(21) opi 0 10 (22) sto 0 3	战 (古)	\rightarrow 0	end.
(22) sto 0 3 (23) jmp 0 11		o	Cita.
(24) opr 0 0			
(-1) op: 00	-H/NA214)		

(0) jmp 0 8	转向主程序入口		, June 1
(1) jmp 0 2	转向过程p入口		♦ 191
(2) int 0 3	过程p入口,为过程p开辟空间		
(3) lod 1 3	取变量b的值到栈顶		const a=10;
(4) lit 0 10	取常数10到栈顶		· I
(5) opr 0 2	次栈顶与栈顶相加		var b,c;
(6) sto 1 4	栈顶值送变量c中		procedure p;
(7) opr 0 0	退栈并返回调用点(16)		begin
(8) int 0 5	主程序入口开辟5个栈空间		c:=b+a;
(9) opr 0 16			
(10) sto 0 3	将栈顶值存入变量b中		end;
(11) lod 0 3	将变量b的值取至栈顶		begin
(12) lit 0 0	将常数值0进栈		read(b);
(13) opr 0 9		4.5	\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \
(14) jpc 0 24		15	while b#0 do
(15) cal 0 2	调用过程p	E	begin
(16) lit 0 2	常数值2进栈	5	call p;
(17) lod 0 4	将变量c的值取至栈顶	0	
(18) opr 0 4		0	write(2*c);
(19) opr 0 14 → (20) opr 0 15		0	read(b);
			end
(21) opr 0 10 (22) sto 0 3		\rightarrow 0	end.
(22) sto 0 3 (23) jmp 0 1		b	ena.
(24) opr 0 0			
(2 4) opi 0 0			

(1) jmp 0 8 转向主程序入口 (1) jmp 0 2 转向过程p入口,为过程p开辟空间 (2) int 0 3 过程p入口,为过程p开辟空间 (3) lod 1 3 取变量b的值到栈顶 (4) lit 0 10 取常数10到栈顶 (5) opr 0 2 次栈顶与栈顶相加 (5) opr 0 0 退栈并返回调用点(16) (8) int 0 5 主程序入口开辟5个栈空间 (9) opr 0 16 从命令行读入值置于栈顶 (10) sto 0 3 将找顶值存入变量b中 (11) lod 0 3 将变量b的值取至栈顶 (12) lit 0 0 将常数值0进栈 (13) opr 0 9 次栈顶与栈顶是否不等 (14) jpc 0 24 相等时转(24) (条件不满足转) (15) cal 0 2 调用过程p (16) lit 0 2 常数值2进栈 (17) lod 0 4 将变量c的值取至栈顶 (18) opr 0 4 次栈顶与栈顶相乘(2*c) (19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行 const a=10; var b,c; procedure p; begin c:=b+a; end; begin read(b); while b#0 do begin call p; write(2*c); read(b);				
(2) int 0 3 过程p入口,为过程p开辟空间 (3) lod 1 3 取变量b的值到栈顶 (4) lit 0 10 取常数10到栈顶 (5) opr 0 2 次栈顶与栈顶相加 (6) sto 1 4 栈顶值送变量c中 (7) opr 0 0 退栈并返回调用点(16) (8) int 0 5 主程序入口开辟5个栈空间 (9) opr 0 16 从命令行读入值置于栈顶 (10) sto 0 3 将线顶值存入变量b中 (11) lod 0 3 将变量b的值取至栈顶 (12) lit 0 0 将常数值0进栈 (13) opr 0 9 次栈顶与栈顶是否不等 (14) jpc 0 24 相等时转(24) (条件不满足转) (15) cal 0 2 调用过程p (16) lit 0 2 常数值2进栈 (17) lod 0 4 将变量c的值取至栈顶 (18) opr 0 4 次栈顶与栈顶相乘(2*c) (19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行	(0) jmp 0 8	转向主程序入口		, /mil
(3) lod 1 3 取变量b的值到栈顶 (4) lit 0 10 取常数10到栈顶 (5) opr 0 2 次栈顶与栈顶相加 (6) sto 1 4 栈顶值送变量c中 (7) opr 0 0 退栈并返回调用点(16) (8) int 0 5 主程序入口开辟5个栈空间 (9) opr 0 16 从命令行读入值置于栈顶 (10) sto 0 3 将栈顶值存入变量b中 (11) lod 0 3 将变量b的值取至栈顶 (12) lit 0 0 将常数值0进栈 (13) opr 0 9 次栈顶与栈顶是否不等 (14) jpc 0 24 相等时转(24) (条件不满足转) (15) cal 0 2 调用过程p (16) lit 0 2 常数值2进栈 (17) lod 0 4 将变量c的值取至栈顶 (18) opr 0 4 次栈顶与栈顶相乘(2*c) (19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行	(1) jmp 0 2	转向过程p入口		♦ 19 1
(4) lit 0 10 取常数10到栈顶 (5) opr 0 2 次栈顶与栈顶相加 (6) sto 1 4 栈顶值送变量c中 (7) opr 0 0 退栈并返回调用点(16) (8) int 0 5 主程序入口开辟5个栈空间 (9) opr 0 16 从命令行读入值置于栈顶 (10) sto 0 3 将栈顶值存入变量b中 (11) lod 0 3 将变量b的值取至栈顶 (12) lit 0 0 将常数值0进栈 (13) opr 0 9 次栈顶与栈顶是否不等 (14) jpc 0 24 相等时转(24) (条件不满足转) (15) cal 0 2 调用过程p (16) lit 0 2 常数值2进栈 (17) lod 0 4 将变量c的值取至栈顶 (18) opr 0 4 次栈顶与栈顶相乘(2*c) (19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行	* *			
(4) lit 0 10 取常数10到栈顶 (5) opr 0 2 次栈顶与栈顶相加 (6) sto 1 4 栈顶值送变量c中 (7) opr 0 0 退栈并返回调用点(16) (8) int 0 5 主程序入口开辟5个栈空间 (9) opr 0 16 从命令行读入值置于栈顶 (10) sto 0 3 将栈顶值存入变量b中 (11) lod 0 3 将变量b的值取至栈顶 (12) lit 0 0 将常数值0进栈 (13) opr 0 9 次栈顶与栈顶是否不等 (14) jpc 0 24 相等时转(24) (条件不满足转) (15) cal 0 2 调用过程p (16) lit 0 2 常数值2进栈 (17) lod 0 4 将变量c的值取至栈顶 (18) opr 0 4 次栈顶与栈顶相乘(2*c) (19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行				const a=10:
(6) sto 1 4	* *			
(7) opr 0 0		* *** *** * * * * * * * * * * * * * * *		
(8) int 0 5 主程序入口开辟5个栈空间 (9) opr 0 16 从命令行读入值置于栈顶 (10) sto 0 3 将栈顶值存入变量b中 (11) lod 0 3 将变量b的值取至栈顶 (12) lit 0 0 将常数值0进栈 (13) opr 0 9 次栈顶与栈顶是否不等 (14) jpc 0 24 相等时转(24) (条件不满足转) (15) cal 0 2 调用过程p (16) lit 0 2 常数值2进栈 (17) lod 0 4 将变量c的值取至栈顶 (18) opr 0 4 次栈顶与栈顶相乘(2*c) (19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行				procedure p;
(8) int 0 5 (9) opr 0 16 从命令行读入值置于栈顶 (10) sto 0 3 将栈顶值存入变量b中 (11) lod 0 3 将变量b的值取至栈顶 (12) lit 0 0 将常数值0进栈 (13) opr 0 9 次栈顶与栈顶是否不等 (14) jpc 0 24 相等时转(24) (条件不满足转) (15) cal 0 2 调用过程p (16) lit 0 2 常数值2进栈 (17) lod 0 4 将变量c的值取至栈顶 (18) opr 0 4 次栈顶与栈顶相乘(2*c) (19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行	` ' -			begin
(10) sto 0 3 将栈顶值存入变量b中 (11) lod 0 3 将变量b的值取至栈顶 (12) lit 0 0 将常数值0进栈 (13) opr 0 9 次栈顶与栈顶是否不等 (14) jpc 0 24 相等时转(24) (条件不满足转) (15) cal 0 2 调用过程p (16) lit 0 2 常数值2进栈 (17) lod 0 4 将变量c的值取至栈顶 (18) opr 0 4 次栈顶与栈顶相乘(2*c) (19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行	* *			
(11) lod 0 3 将变量b的值取至栈顶 (12) lit 0 0 将常数值0进栈 (13) opr 0 9 次栈顶与栈顶是否不等 (14) jpc 0 24 相等时转(24) (条件不满足转) (15) cal 0 2 调用过程p (16) lit 0 2 常数值2进栈 (17) lod 0 4 将变量c的值取至栈顶 (18) opr 0 4 次栈顶与栈顶相乘(2*c) (19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行	. , .			·
(12) lit 0 0 将常数值0进栈 (13) opr 0 9 次栈顶与栈顶是否不等 (14) jpc 0 24 相等时转(24) (条件不满足转) (15) cal 0 2 调用过程p (16) lit 0 2 常数值2进栈 (17) lod 0 4 将变量c的值取至栈顶 (18) opr 0 4 次栈顶与栈顶相乘(2*c) (19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行	` '			end;
(12) lit 0 0	* *			begin
(14) jpc 0 24 相等时转(24) (条件不满足转) (15) cal 0 2 调用过程p (16) lit 0 2 常数值2进栈 (17) lod 0 4 将变量c的值取至栈顶 (18) opr 0 4 次栈顶与栈顶相乘(2*c) (19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行				
(15) cal 0 2 调用过程p (16) lit 0 2 常数值2进栈 (17) lod 0 4 将变量c的值取至栈顶 (18) opr 0 4 次栈顶与栈顶相乘(2*c) (19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行			4.5	\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \
(16) lit 0 2 常数值2进栈 5 (17) lod 0 4 将变量c的值取至栈顶 0 (18) opr 0 4 次栈顶与栈顶相乘(2*c) 0 (19) opr 0 14 栈顶值输出至屏幕 0 (20) opr 0 15 换行	` ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' '		15	while b#0 do
(17) lod 0 4 将变量c的值取至栈顶 (18) opr 0 4 次栈顶与栈顶相乘(2*c) (19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行 Call p; write(2*c); read(b);			5	begin
(18) opr 0 4 次栈顶与栈顶相乘(2*c) (19) opr 0 14 栈顶值输出至屏幕 (20) opr 0 15 换行	` '		5	call p:
(19) opr 0 14			0	
(20) opr 0 15 换行	` ' -	· · · · · · · · · · · · · · · · · · ·		
			0	read(b);
→ (21) opr 0 16 从命令行读取值到栈顶 end end	` ' '			end
n (22) sto 0.3 株顶値送恋量b由 — Ond And		以		end
p (22) sto 0 3	, ,)	J Clid.
(24) opr 0 0 结束退栈	, , ,			

			Name of Street, or other Designation of the Owner, where the Parket of the Owner, where the Owner, which is the Owner, where the Owner, which is the Owner, where the Owner, which is the Owner, whic
(0) jmp 0 8	转向主程序入口		, /Til
(1) jmp 0 2	转向过程p入口		♦ 19]
(2) int 0 3	过程p入口,为过程p开辟空间		
(3) lod 1 3	取变量b的值到栈顶		const a=10;
(4) lit 0 10	取常数10到栈顶		var b,c;
(5) opr 0 2	次栈顶与栈顶相加		
(6) sto 14	栈顶值送变量c中		procedure p;
(7) opr 0 0 (8) int 0 5	退栈并返回调用点(16) 主程序入口开辟5个栈空间		begin
(9) opr 0 16	从命令行读入值置于栈顶		c:=b+a;
(10) sto 0 3	将栈顶值存入变量b中		end;
(11) lod 0 3	将变量b的值取至栈顶		
(12) lit 0 0	将常数值0进栈	0	
(13) opr 0 9	次栈顶与栈顶是否不等		read(b);
(14) jpc 0 24	相等时转(24)(条件不满足转)	15	while b#0 do
(15) cal 0 2	调用过程p		begin
(16) lit 0 2	常数值2进栈	5	call p;
(17) lod 0 4	将变量c的值取至栈顶	0	
(18) opr 0 4	次栈顶与栈顶相乘(2*c)	0	write(2*c);
(19) opr 0 14 (20) opr 0 15	栈顶值输出至屏幕 换行	0	read(b);
(20) opr 0 13 (21) opr 0 16	从命令行读取值到栈顶		end
$\longrightarrow (22) \text{ sto } 0.3$	 お 「 は は は は は は は は は は は は は は は は は は	→ 0	end.
(23) jmp 0 11			
(24) opr 0 0	结束退栈		

(0) jmp 0 8 \$	传向主程序入口		A Arti
	传向过程p入口		♦ 191
	过程p入口,为过程p开辟空间		
·	双变量b的值到栈顶		const a=10;
	仅常数10到栈顶		var b,c;
` ' -	次栈顶与栈顶相加		
· ,	浅顶值送变量c中		procedure p;
	3栈并返回调用点(16) - 积点 》 D.T. 195 A. 比容词		begin
. ,	E程序入口开辟5个栈空间 从命令行读入值置于栈顶		c:=b+a;
` ' •	A叩マ11 英八恒直丁13.IV B栈顶值存入变量b中		end;
	的 於 於 於 於 於 於 於 的 的 的 的 的 的 的 的 的 的 的 的		· · · · · · · · · · · · · · · · · · ·
	3文量 0 13 1 1 1 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1		begin
	次栈顶与栈顶是否不等		read(b);
	等时转(24)(条件不满足转)	15	t while b#0 do
	周用过程p		begin
	常数值2进栈	0	
	的 整量c的值取至栈顶		call p;
·	欠栈顶与栈顶相乘(2*c)	0	write(2*c);
·	线顶值输出至屏幕 7.7	0	read(b);
			end
` ' -	人命令行读取值到栈顶 とで使送水量6点	0	
	送顶值送变量b中 大条件转到循环入口(11) b		end.
	5余件转到循环人口(II <i>)</i> 6束退栈		
P (24) Opi 0 0 \$	日本巡找		00

(0) jmp 0 8	转向主程序入口		, Prof
(1) jmp 0 2	转向过程p入口		♦ 9
(2) int 0 3	过程p入口,为过程p开辟空间		
(3) lod 13	取变量b的值到栈顶		const a=10;
(4) lit 0 10	取常数10到栈顶		var b,c;
(5) opr 0 2	次栈顶与栈顶相加		
(6) sto 1 4	栈顶值送变量c中		procedure p;
(7) opr 0 0	退栈并返回调用点(16)		begin
(8) int 0 5	主程序入口开辟5个栈空间		c:=b+a;
(9) opr 0 16	从命令行读入值置于栈顶		'
(10) sto 0 3	将栈顶值存入变量b中		end;
$\longrightarrow (11) \log 0.3$	将变量b的值取至栈顶		begin
p (12) lit 0 0	将常数值0进栈		read(b);
(13) opr 0 9	次栈顶与栈顶是否不等	4.5	• • • • • • • • • • • • • • • • • • • •
(14) jpc 0 24	相等时转(24)(条件不满足转)	15	while b#0 do
(15) cal 0 2	调用过程p 常数值2进栈	0	begin
(16) lit 0 2 (17) lod 0 4	市致恒2近伐 将变量c的值取至栈顶	0	call p;
(17) lod 0 4 (18) opr 0 4	次栈顶与栈顶相乘(2*c)	0	write(2*c);
(19) opr 0 14	栈顶值输出至屏幕		
(20) opr 0 15	换行	0	read(b);
(21) opr 0 16	从命令行读取值到栈顶		end
(22) sto 0 3	 は 「 は に は に に に に に に に に に に	\rightarrow 0	end.
(23) jmp 0 11		o	
(24) opr 0 0	结束退栈		M D 57

			Name of Street, or other Designation of the Owner, where the Parket of the Owner, where the Owner, which is the Owner, where the Owner, which is the Owner, where the Owner, which is the Owner, whic
(0) jmp 0 8	转向主程序入口		A Pril
(1) jmp 0 2	转向过程p入口		♦ 19 1
(2) int 0 3	过程p入口,为过程p开辟空间		
(3) lod 1 3	取变量b的值到栈顶		const a=10;
(4) lit 0 10	取常数10到栈顶		var b,c;
(5) opr 0 2	次栈顶与栈顶相加		
(6) sto 1 4 (7) opr 0 0	栈顶值送变量c中 温埃共源回海田点(46)		procedure p;
(7) opr 0 0 (8) int 0 5	退栈并返回调用点(16) 主程序入口开辟5个栈空间		begin
(9) opr 0 16	从命令行读入值置于栈顶		c:=b+a;
(10) sto 0 3	将栈顶值存入变量b中		end;
(11) lod 0 3	将变量b的值取至栈顶		
\longrightarrow (12) lit 0 0	将常数值0进栈	0	t begin
(13) opr 0 9	次栈顶与栈顶是否不等		read(b);
(14) jpc 0 24	相等时转(24)(条件不满足转)	15	while b#0 do
(15) cal 0 2	调用过程p		begin
(16) lit 0 2	常数值2进栈	0	call p;
(17) lod 0 4	将变量c的值取至栈顶	0	· · ·
(18) opr 0 4	次栈顶与栈顶相乘(2*c)	0	write(2*c);
(19) opr 0 14 (20) opr 0 15	栈顶值输出至屏幕 换行	0	read(b);
(20) opr 0 15 (21) opr 0 16	从命令行读取值到栈顶		end
(22) sto 0 3	 お 「 は は は は は は は は は は は は は は は は は は	→ 0	end.
(23) jmp 0 11			
(24) opr 0 0	结束退栈		

(0) jmp 0 8	转向主程序入口		. / -1
(1) jmp 0 2 (2) int 0 3 (3) lod 1 3 (4) lit 0 10	转向过程p入口 过程p入口,为过程p开辟空间 取变量b的值到栈顶 取常数10到栈顶 次栈顶与栈顶相加		const a=10;
(5) opr 0 2 (6) sto 1 4 (7) opr 0 0 (8) int 0 5 (9) opr 0 16	人民员与党员相加 栈顶值送变量c中 退栈并返回调用点(16) 主程序入口开辟5个栈空间 从命令行读入值置于栈顶		procedure p; begin c:=b+a;
(10) sto 0 3 (11) lod 0 3	将栈顶值存入变量b中 将变量b的值取至栈顶	0	t end; begin
(12) lit 0 0 → (13) opr 0 9	将常数值0进栈 次栈顶与栈顶是否不等	0	read(b);
(14) jpc 0 24 (15) cal 0 2	相等时转(24)(条件不满足转) 调用过程p	15	while b#0 do begin
(16) lit 0 2 (17) lod 0 4	常数值2进栈 将变量c的值取至栈顶	0	call p;
(18) opr 0 4 (19) opr 0 14	次栈顶与栈顶相乘(2*c) 栈顶值输出至屏幕	0	write(2*c); read(b);
(20) opr 0 15 (21) opr 0 16 (22) sto 0 3	换行 从命令行读取值到栈顶 栈顶值送变量b中	→ 0	end end.
(23) jmp 0 11 (24) opr 0 0			

(0) jmp	08 转向	主程序入口			. /mil
(1) jmp		过程p入口			♦ 19 11
(2) int (p入口,为过程p开辟空间			
(3) lod		量b的值到栈顶			const a=10;
(4) lit 0		数10到栈顶			var b,c;
(5) opr		页与栈顶相加 法举章			
(6) sto		值送变量c中			procedure p;
(7) opr		并返回调用点(16) 第2000年 (16)			begin
(8) int (京入口开辟5个栈空间 冬年法》位置工程语			c:=b+a;
(9) opr		令行读入值置于栈顶 表传表入恋号\: 中			·
(10) sto (11) lod		页值存入变量b中 量b的值取至栈顶			end;
(11) lod (12) lit 0		型的追取主 成 项 效值0进栈		\dashv t	pegin
(12) iit (对自心还说 页与栈顶是否不等	0		read(b);
→ (14) jpc		对转(24)(条件不满足转)	15		while b#0 do
(14) jpo		过程p	10		
(16) lit 0	, -,,	三元 三2进栈	0		begin
(17) lod		=-2.13 量c的值取至栈顶			call p;
(18) opr		页与栈顶相乘(2*c)	0		write(2*c);
(19) opr	· 0 14 栈顶(直输出至屏幕			read(b);
(20) opr			0		
(21) opr		令行读取值到栈顶	\rightarrow 0		end
(22) sto		直送变量b中	b	∈	end.
(23) jmp		牛转到循环入口(11)	D		
(24) opr	·00 结束i	艮栈			60

(0) jmp 0 8	转向主程序入口		, /T-1
(1) jmp 0 2	转向过程p入口		♦ 19 1
(2) int 0 3	过程p入口,为过程p开辟空间		
(3) lod 1 3	取变量b的值到栈顶		const a=10;
(4) lit 0 10	取常数10到栈顶		·
(5) opr 0 2	次栈顶与栈顶相加		var b,c;
(6) sto 1 4	栈顶值送变量c中		procedure p;
(7) opr 0 0	退栈并返回调用点(16)		begin
(8) int 0 5	主程序入口开辟5个栈空间		c:=b+a;
(9) opr 0 16	从命令行读入值置于栈顶		·
(10) sto 0 3	将栈顶值存入变量b中		end;
(11) lod 0 3	将变量b的值取至栈顶		begin
(12) lit 0 0	将常数值0进栈		read(b);
(13) opr 0 9	次栈顶与栈顶是否不等	4 =	●
(14) jpc 0 24	相等时转(24)(条件不满足转)	15	while b#0 do
(15) cal 0 2	调用过程p		begin
(16) lit 0 2	常数值2进栈	0	call p;
(17) lod 0 4	将变量c的值取至栈顶	0	-
(18) opr 0 4	次栈顶与栈顶相乘(2*c)	0	write(2*c);
(19) opr 0 14	栈顶值输出至屏幕 ***	0	read(b);
(20) opr 0 15	换行 以 <i>会</i> 会经验现债率以		end
(21) opr 0 16	从命令行读取值到栈顶 **还使送恋量b中	\rightarrow 0	
(22) sto 0 3		b	end.
p (23) jmp 0 11			
$\stackrel{\bullet}{\longrightarrow}$ (24) opr 0 0	结束退栈		01

(0) jmp 0 8	转向主程序入口		A Prof
(1) jmp 0 2	转向过程p入口		♦ 49]
(2) int 0 3	过程p入口,为过程p开辟空间		
(3) lod 1 3	取变量b的值到栈顶		const a=10;
(4) lit 0 10	取常数10到栈顶		· · · · · · · · · · · · · · · · · · ·
(5) opr 0 2	次栈顶与栈顶相加		var b,c;
(6) sto 1 4	栈顶值送变量c中		procedure p;
(7) opr 0 0	退栈并返回调用点(16)		begin
(8) int 0 5	主程序入口开辟5个栈空间		c:=b+a;
(9) opr 0 16	从命令行读入值置于栈顶		
(10) sto 0 3	将栈顶值存入变量b中		end;
(11) lod 0 3	将变量b的值取至栈顶		begin
(12) lit 0 0	将常数值0进栈		read(b);
(13) opr 0 9	次栈顶与栈顶是否不等		` '
(14) jpc 0 24	相等时转(24)(条件不满足转)		while b#0 do
(15) cal 0 2	调用过程p		begin
(16) lit 0 2	常数值2进栈		call p;
(17) lod 0 4	将变量c的值取至栈顶		• •
(18) opr 0 4	次栈顶与栈顶相乘(2*c)		write(2*c);
(19) opr 0 14	栈顶值输出至屏幕 ************************************		_ read(b);
(20) opr 0 15	投行	b t	end
(21) opr 0 16	从命令行读取值到栈顶		
(22) sto 0 3	栈顶值送变量b中	P=0	end.
(23) jmp 0 11		-	
(24) opr 0 0	结束退栈		<u> </u>