山东大学
SHANDONG UNIVERSITY

编译原理

第五章 符号表管理

授 课 教 师 ：郑艳伟
手 机 ：18614002860（微信同号）
邮 箱 ：zhengyw@sdu.edu.cn

第五章 符号表管理

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

第五章 符号表管理

5.1 符号表的作用

符号表的作用

- 收集符号属性，如各种变量名及其数据类型、数据长度、内存偏移量等；
- 上下文语义的合法性检查的依据，如变量重复定义、标号检查等；
- 作为目标代码生成阶段地址分配和使用的依据。

对符号表的操作

- 对给定名字，查询此名是否已在表中；
- 往表中填入一个新的名字；
- 对给定名字，访问它的某些信息；
- 对给定名字，往表中填写或更新它的某些信息；
- 删除一个或一组无用的项。

第五章 符号表管理

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

变量

变量

- 名字：可以是变量、函数、过程、类的名字，一般不允许重名。
- 数据类型：如整型、实型、布尔型、字符型等等。
- 类别：如全局变量、局部变量、临时变量、形式参数等；有时候全局变量和局部变量通过所在符号表区分，而不是使用作用域信息区分，因此可以统称为普通变量。
- 宽度：指该具体数据类型占用的字节数，如整型、实型分别为4和8，而数组则是元素数量乘以每个元素宽度的值。
- 偏移量：即相对于某个基地址的偏移地址。

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

变量

```
1  int Fun(int x, int y) {
2    int i, j;
3    double u, v;
4    int arr[2, 3, 4];
5    ...
6  }
```

名字	数据类型	类别	宽度	偏移量
x	int	形参变量	4	0
y	int	形参变量	4	4
i	int	普通变量	4	0
j	int	普通变量	4	4
u	double	普通变量	8	8
v	double	普通变量	8	16
arr	array(2, 3, 4, int)	普通变量	96	24

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

数组

- 如果变量为数组，应建立子表额外记录如下信息
  - 数组维度。
  - 静态地址：指编译时确定的基准地址偏移，是由数组每个维度的下界及每个维度长度共同确定的。
  - 每个维度的下界：现在大部分语言下界是固定不变的，如C 语言数组下界为0，matlab 数组下界为1，因此可以省略该项。
  - 每个维度的上界。
  - 每个维度的元素数。

int arr[2,3,4]的内情向量表

维度	下界	上界	元素数
0	0	1	2
1	0	2	3
2	0	3	4

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

结构体

- 如果变量为结构体，应建立子表额外记录结构体各分量的信息
  - 一般来说，结构体子表结构等同于变量表结构。

```
1  struct STRU {
2    int a;
3    float b[20];
4    double c;
5  };
6  STRU s;
```

名字	数据类型	类别	宽度	偏移量
STRU	struct	结构体	92	–
s	struct(STRU)	普通变量	92	0
a	int	结构体成员	4	0
b	array(20, float)	结构体成员	80	4
c	double	结构体成员	8	84

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

过程和标号

- 过程
  - 名字
  - 是否为程序的外部过程?
  - 入口地址，对于能够通过名字使用call 指令调用的目标语言，可以省略该项；对于编译为可执行代码的编译器必须包含该项。
  - 返回值类型。
  - 返回值长度。
  - 形式参数列表，其结构等同于变量表。
- 标号
  - 名字
  - 标号地址
  - 是否已定义

第五章 符号表管理

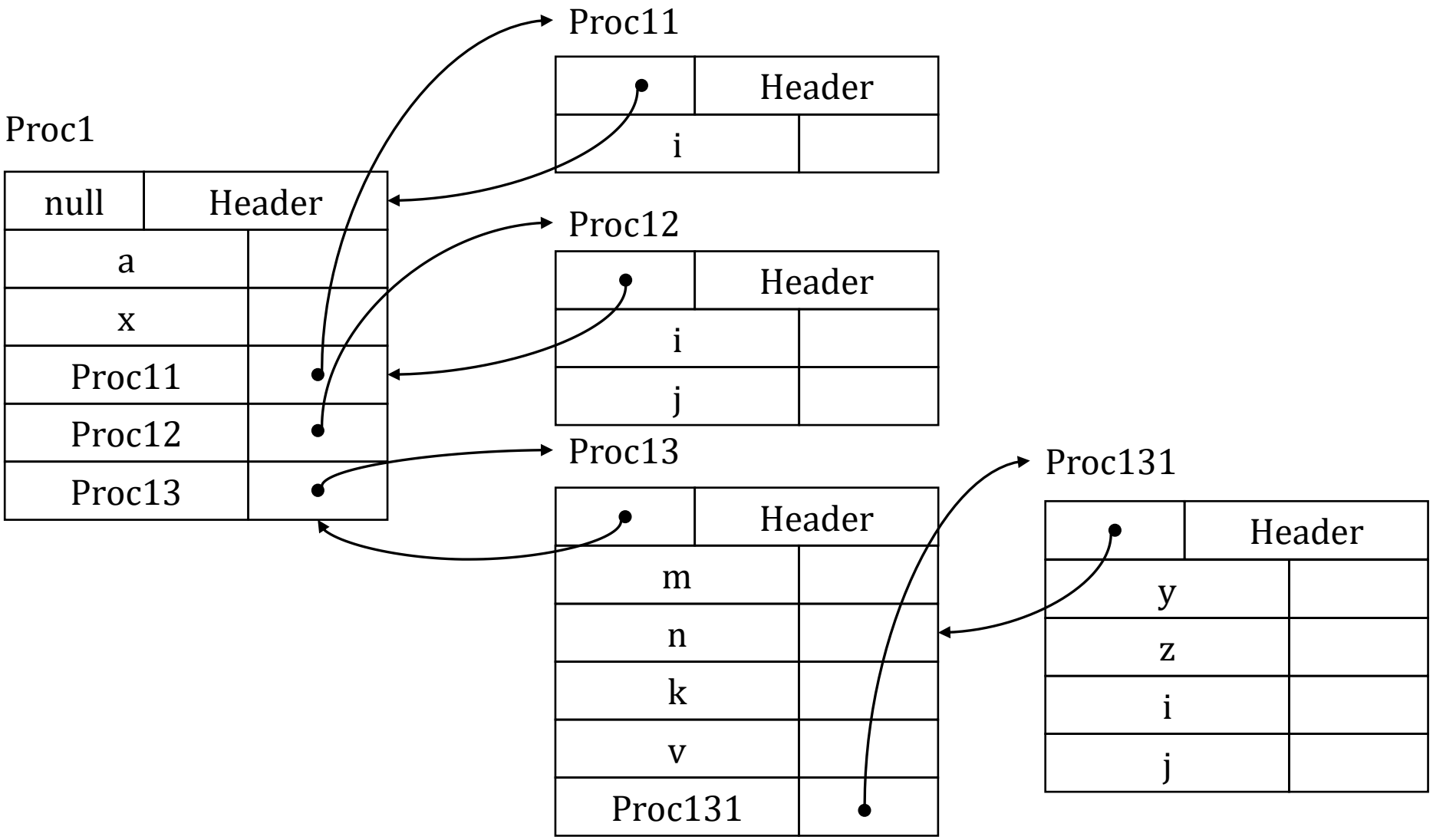
山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

```
1  program Proc1;
2    var a: array [0..10] of integer;
3      x: integer;
4      procedure Proc11;
5        var i: integer;
6        begin
7          ... a ...
8        end {Proc11};
9      procedure Proc12 (i, j: integer);
10       begin
11         x = a[i];
12         a[i] = a[j];
13         a[j] = x;
14       end {Proc12};
15     procedure Proc13 (m, n: integer);
16       var k, v: integer;
17       function Proc131 (y, z: integer) : integer;
18         var i, j: integer;
19         begin
20           ... a ... v ...
21           Proc12(i, j);
22           ...
23         end {Proc131};
24       begin
25         ...
26       end {Proc13};
27   begin
28     ...
29   end {Proc1}.
```

- 一个典型的嵌套过程定义的变量作用域规则
  - 访问变量如果在本过程声明或为本过程形参，则使用本过程的局部变量或形参
  - 访问变量如果不是本过程局部变量或形参，则在上层父过程局部变量或形参中查找，找到则使用。
  - 如果父过程仍未找到访问变量，则继续向上级父过程查找，直到找到，或者不再有父过程（调用的变量不存在）。

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

嵌套定义过程符号表结构组织

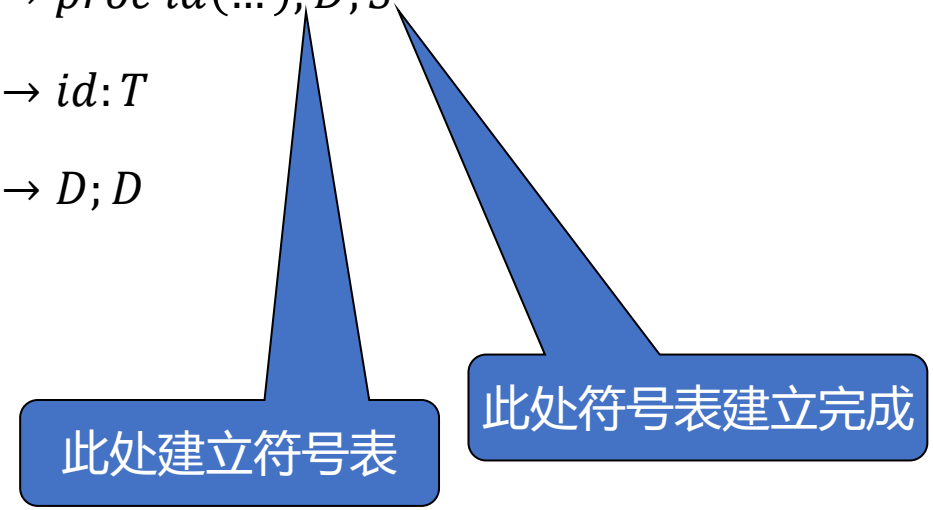


第五章 符号表管理

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

6.3.1 嵌套定义过程符号表结构组织

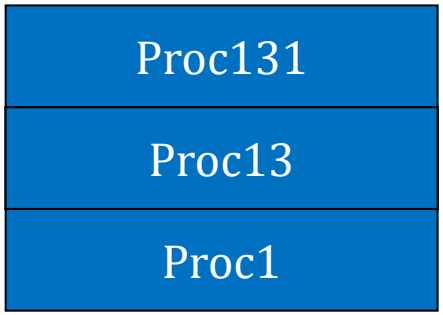
- 一个不太严谨的过程嵌套定义文法
  - $D \rightarrow proc\ id(\ldots); D; S$
  - $D \rightarrow id: T$
  - $D \rightarrow D; D$



山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

```
1  program Proc1;
2    var a: array [0..10] of integer;
3      x: integer;
4    procedure Proc11;
5      var i: integer;
6      begin
7        ... a ...
8      end {Proc11};
9    procedure Proc12 (i, j: integer);
10     begin
11       x = a[i];
12       a[i] = a[j];
13       a[j] = x;
14     end {Proc12};
15   procedure Proc13 (m, n: integer);
16     var k, v: integer;
17     function Proc131 (y, z: integer) : integer;
18       var i, j: integer;
19       begin
20         ... a ... v ...
21         Proc12(i, j);
22         ...
23       end {Proc131};
24     begin
25       ...
26     end {Proc13};
27 begin
28   ...
29 end {Proc1}.
```

符号表栈解决方案



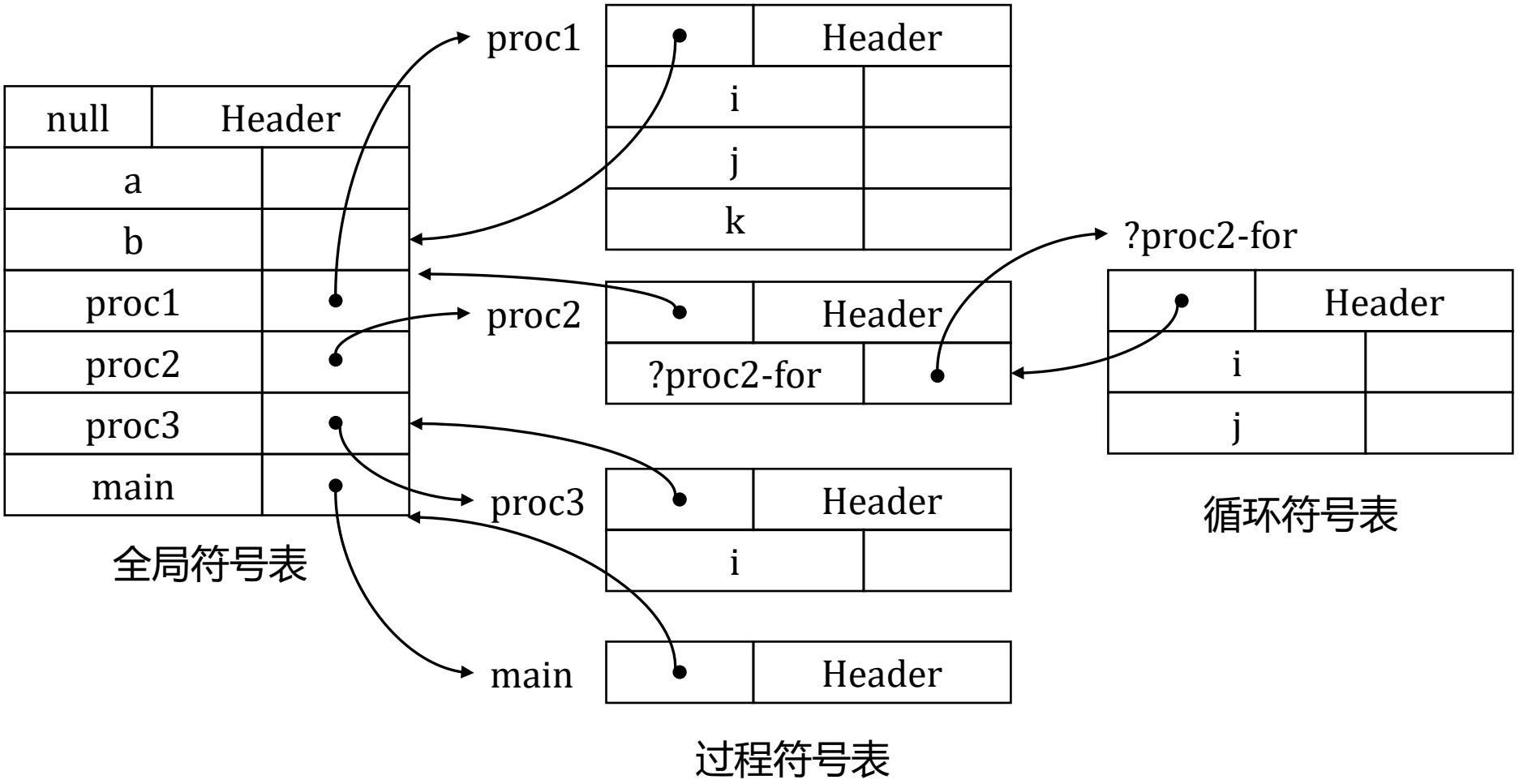
第五章 符号表管理

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

```
1   int a, b;
2   void proc1(int i, int j) {
3     int k;
4     ...
5   }
6   void proc2() {
7     ...
8     for (int i = 1; i < 100; i++) {
9       int j = 0;
10      ...
11    }
12    ...
13  }
14  void proc3() {
15    int i;
16    ...
17  }
18  int main() {
19    ...
20    return 0;
21  }
```

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

非嵌套定义过程符号表结构组织



第五章 符号表管理

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

多符号表组织

- 主要问题
  - 不同种类的符号，属性信息有差异。
- 第1种组织方式：构造多个符号表，具有相同属性种类的符号组织在一起
  - 优点：每个符号表中存放符号的属性个数和结构完全相同；
  - 缺点：一遍编译程序同时管理若干个符号表。

符号	属性1	属性2	属性3

符号	属性1	属性2	属性4

符号	属性2	属性3	属性5	属性6	属性7	属性8

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

单符号表组织

- 第2种组织方式：把所有符号都组织在一张符号表中
  - 优点：管理集中单一；
  - 缺点：增加了空间开销。

符号	属性1	属性2	属性3	属性4	属性5	属性6	属性7	属性8

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

相似表合并

- 第3种组织方式：根据符号属性相似程度分类组织成若干张表
  - 优点：减少了空间开销；
  - 缺点：增加了表格管理的复杂性。

符号	属性1	属性2	属性3	属性4

第1、2种符号

符号	属性2	属性3	属性5	属性6	属性7	属性8

第3种符号

第五章 符号表管理

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

表记录组织

- 如果选择了不支持表格、结构体和对象的编译器实现语言，就需要自己组织表记录在内存的存储方式。
- 整数
  - 大尾编码即高位字节放低地址端：如果地址左低右高，则0x1234 用两个字节存放，大尾编码为12 34；如果用4 字节存放，大尾编码为00 00 12 34。
  - 小尾编码即高位字节放高地址端：如果地址左低右高，则0x1234 用两个字节存放，小尾编码为34 12；如果用4 字节存放，小尾编码为34 12 00 00。

表记录组织

字符串

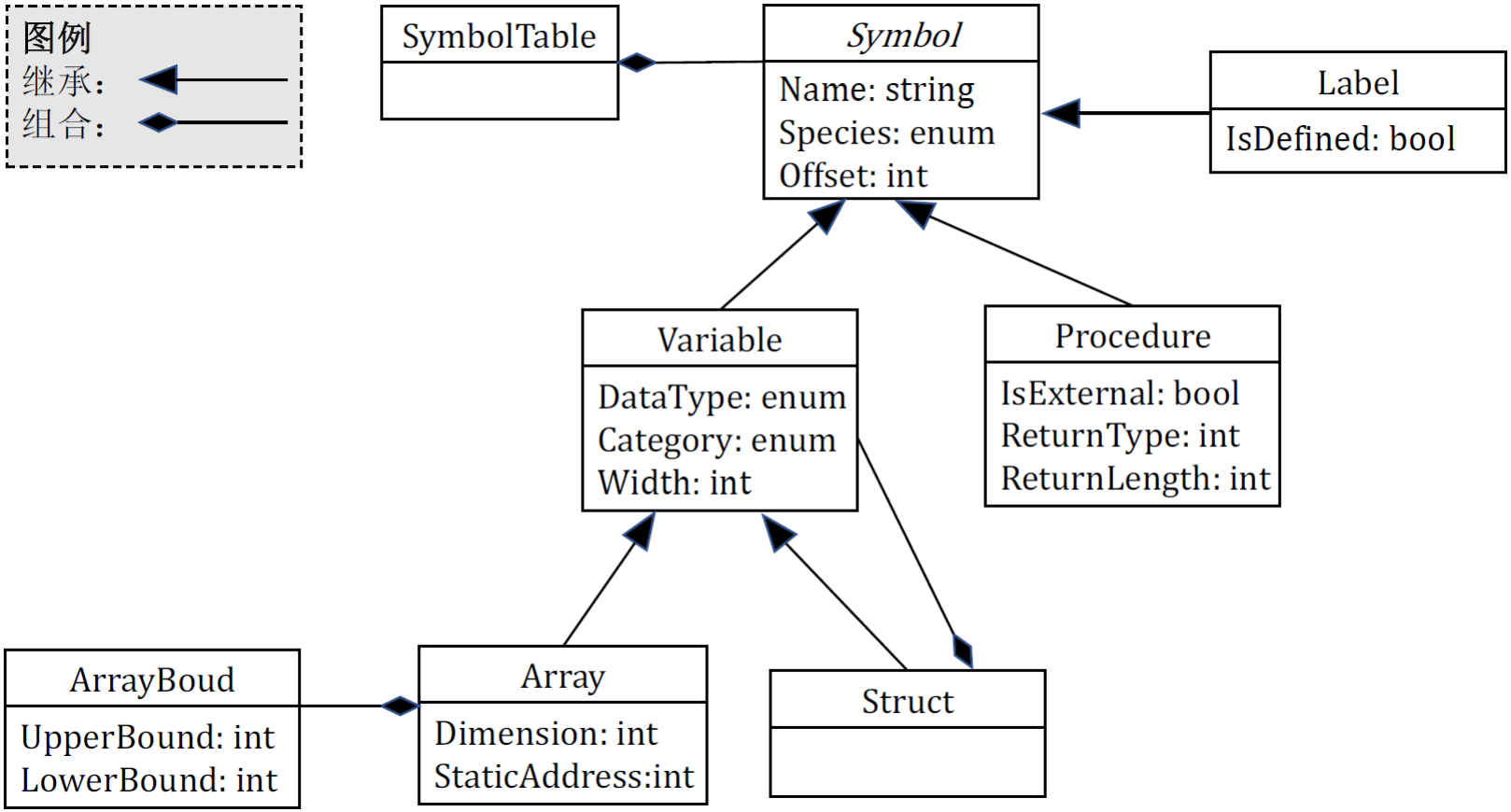
- 固定宽度法：比如约定名字最长为256 字符，那么可以用256 字节长度的空间表示名字字符串，不足256 字节的可以用特殊符号如整数0 补足。
- 终止符法：C 语言等用整数0 作为字符串的结束，这显然能避免空间的浪费，但同时带来了地址计算的问题。
- 指定长度法：即先用一个数字指定字符串长度，后面再接字符串。如用2 个字节小尾编码表示字符串长度，则0C 00 48 65 6C 6C 6F 20 77 6F 72 6C 64 21 的前两个字节表示字符串长度为12，后面12 个字节就是其内容，ASCII 编码为“Hello world!”。

第五章 符号表管理

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

面向对象的组织

- 使用对象组织
  - 优点：对象可变长，减少了空间开销，也便于管理；
  - 缺点：需要编译器实现语言的支持。



第五章 符号表管理

- 5.1 符号表的作用
- 5.2 符号表内容
- 5.3 符号表结构组织
  - 5.3.1 嵌套定义过程
  - 5.3.2 符号表栈
  - 5.3.3 非嵌套过程
- 5.4 符号表内容组织
  - 5.4.1 表格组织
  - 5.4.2 表记录组织
  - 5.4.3 面向对象的组织
- 5.5 符号表排序与查找
  - 5.5.1 线性组织
  - 5.5.2 二叉树
  - 5.5.3 平衡二叉树
  - 5.5.4 哈希表

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

线性组织

线性组织

- 优点：插入快，空间效率高；
- 缺点：查询慢，时间效率差。

```
1  int i, j;
2  float sum, product;
3  int num, length;
```

符号	属性
i	
j	
sum	
product	
num	
length	

第五章 符号表管理

- 5.1 符号表的作用
- 5.2 符号表内容
- 5.3 符号表结构组织
  - 5.3.1 嵌套定义过程
  - 5.3.2 符号表栈
  - 5.3.3 非嵌套过程
- 5.4 符号表内容组织
  - 5.4.1 表格组织
  - 5.4.2 表记录组织
  - 5.4.3 面向对象的组织
- 5.5 符号表排序与查找
  - 5.5.1 线性组织
  - 5.5.2 二叉树
  - 5.5.3 平衡二叉树
  - 5.5.4 哈希表

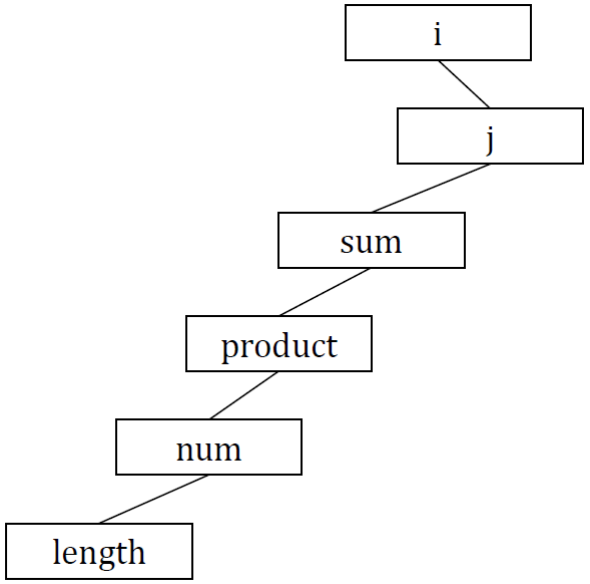
山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

二叉树

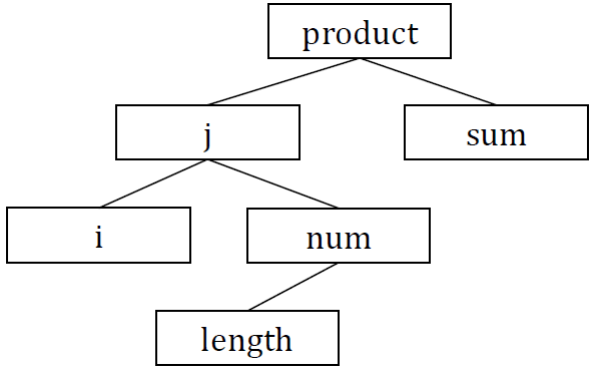
二分法

- 优点：查询效率高，空间效率高；
- 缺点：插入效率低，算法复杂一些。

```
1  int i, j;
2  float sum, product;
3  int num, length;
```




```
1  float product, sum;
2  int j, i;
3  int num, length;
```

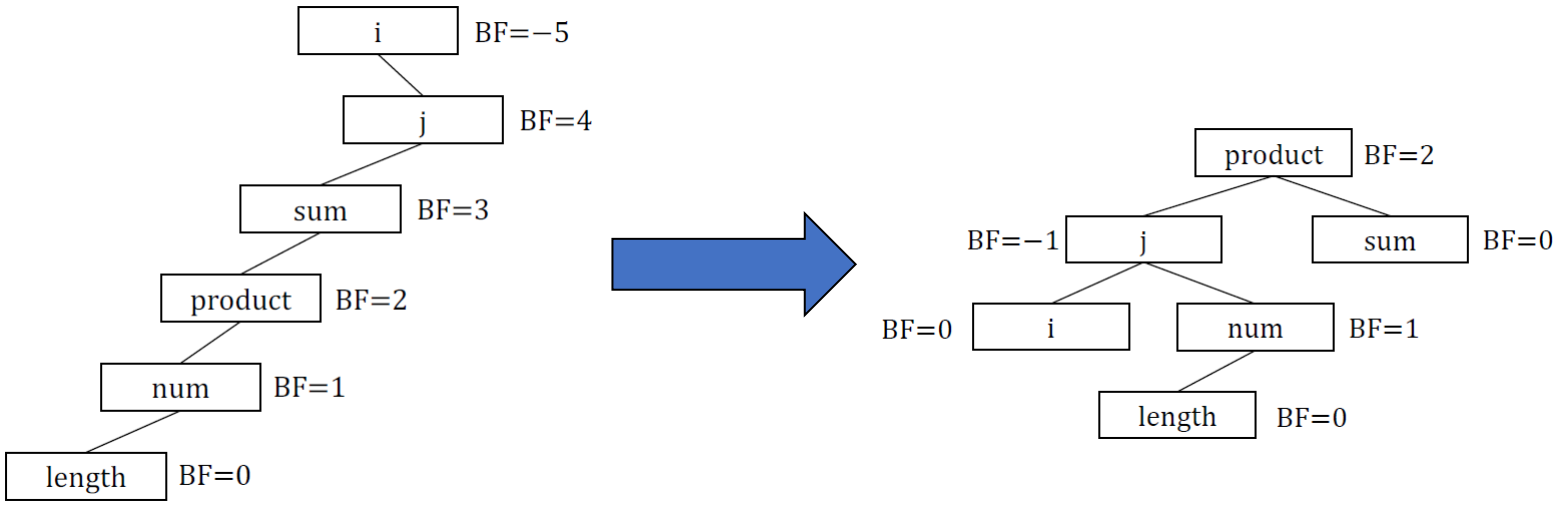


第五章 符号表管理

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

平衡二叉树

- AVL树：每个结点的左右子树高度差都不超过1
  - 优点：查询效率高，空间效率高；
  - 缺点：插入效率低，算法复杂一些。



第五章 符号表管理

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

哈希表

- Hash表
  - 优点：插入、查询效率都高；
  - 缺点：空间效率有所降低。
- 直接定址法：H(key) = (a * key + b) % m，其中m是哈希表的长度。
  - 例：H(key) = key % m，其中m = 10。

1, 30, 34

0	1	2	3	4	5	6	7	8	9

	1								
30	1								
30	1			34					

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

哈希函数

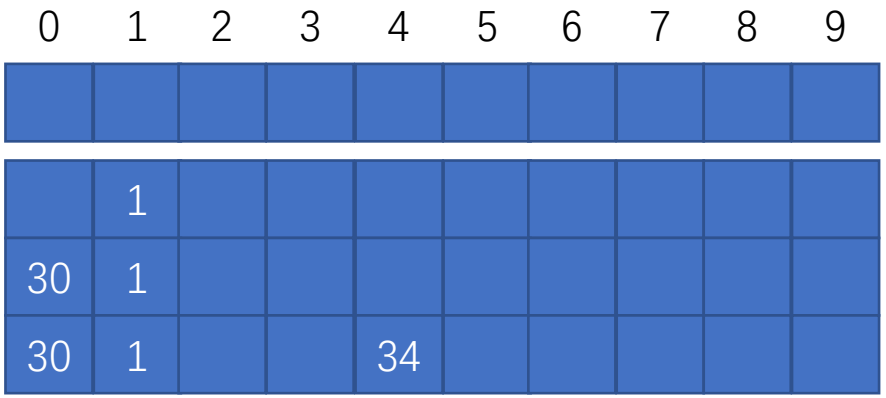
- 直接定址法：H(key) = (a * key + b) % m，其中m是哈希表的长度。
  - 例：H(key) = key % m。
- 数字分析法：取中间某些有区分度的数字。
  - 例：身份证作为key，同一个地区的可以取生日开始的8+3位。
- 平方取中法：如果关键字的每一位都有某些数字重复出现频率很高的现象，可以先求关键字的平方值以扩大差异，取中间数位作为最终存储地址。
  - 例：key=1234 1234^2=1522756 取2275作hash地址
  - key=4321 4321^2=18671041 取6710作hash地址。
- 数字折叠法：如果数字的位数很多，可以将数字分割为几个部分，取他们的叠加和作为hash地址。
  - 例：key=123 456 789，折叠(123 + 456 + 789) % 1000 = 491。
- 除留余数法：H(key) = key % p (p≤m, m为表长)

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

哈希表函数示例

- 直接定址法：H(key) = (a * key + b) % m，其中m是哈希表的长度。
  - 例：H(key) = key % m。

Key: 1, 30, 34, 50, 77, 60, 44, 37



山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

哈希冲突

- 开放定址法解决哈希冲突：如果$H(key_i) = H(key_j)$，则$H_i = [H(key) + d_i] \% m$，其中$d_i$有三种取法：
  - 线性探测再散列：$d_i = c * i$
  - 平方探测再散列：$d_i = 1^2, -1^2, 2^2, -2^2, \ldots$
  - 随机探测在散列（双探测再散列）：d_i是一组伪随机数列

$H(key) = key \% m$，其中$m = 10$，取$d_i = i$

Key: 1, 30, 34, 50, 77, 60, 44, 37

0	1	2	3	4	5	6	7	8	9
	1								
30	1								
30	1			34					
30	1	50		34					

	0	1	2	3	4	5	6	7	8	9
1	30	1	50		34					
2	30	1	50		34			77		
3	30	1	50	60	34			77		
4	30	1	50	60	34	44		77		
5	30	1	50	60	34	44		77	37	

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

哈希冲突

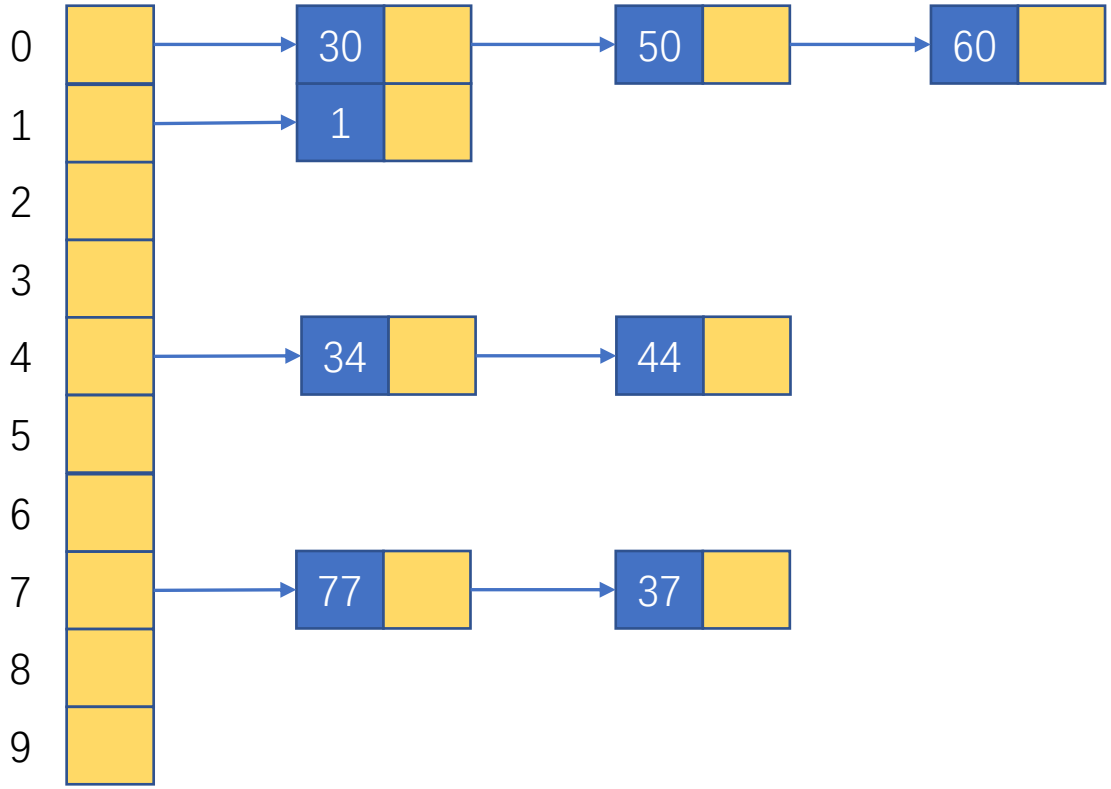
- 开放定址法：如果$H(key_u) = H(key_v)$, 则$H_i = [H(key) + d_i] \% m$，其中$d_i$有三种取法：
  - 线性探测再散列：$d_i = c * i$
  - 平方探测再散列：$d_i = 1^2, -1^2, 2^2, -2^2, \ldots$
  - 随机探测在散列（双探测再散列）：d_i是一组伪随机数列
- 再哈希法：如果$H_1(key_i) = H_1(key_j)$, 则使用$H_2(key_i) = H_2(key_j)$, 如果还冲突，再使用$H_3(key_i) = H_3(key_j)$, …
- 链地址法：将所有关键字为同义词的记录存储在同一线性链表中。

山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

哈希冲突

- 链地址法：将所有关键字为同义词的记录存储在同一线性链表中。

Key: 1, 30, 34, 50, 77, 60, 44, 37



山东大学 SHANDONG UNIVERSITY 郑艳伟，zhengyw@sdu.edu.cn

哈希表示例

表 5.3 采用 16 进制 ASCII 码首尾相连实现名字转为数字表示

名字	16 进制 ASCII 值	对应 10 进制数字
i	69	105
j	6A	106
sum	73756D	7566701
product	70726F64756374	31651020143944564
num	6E756D	7239021
length	6C656E677468	119182899770472

0	1	2	3	4	5	6	7	8	9
	sum/num	length		product	i	j			

图 5.15 直接定址法

0	1	2	3	4	5	6	7	8	9
	sum	num	length	product	i	j			

(a) 按 i、j、sum、product、num、length 顺序填表

0	1	2	3	4	5	6	7	8	9
	sum	length	num	product	i	j			

(b) 按 i、j、sum、product、length、num 顺序填表

图 5.16 线性探测再散列 $d_i = i$

山东大学
SHANDONG UNIVERSITY

第五章 符号表管理

The End

谢谢！

授 课 教 师 ：郑艳伟
手 机 ：18614002860（微信同号）
邮 箱 ：zhengyw@sdu.edu.cn