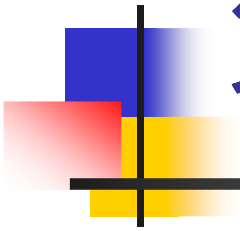
数据结构——广义表

主讲：张昱

yuzhang@ustc.edu

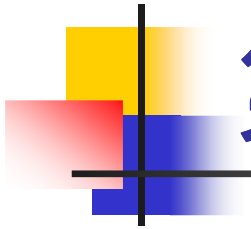
0551-3603804



第五章　数组和广义表

说明：数组移到«算法基础»课中介绍

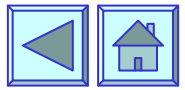
重点：广义表的定义和存储结构、广义表的递归算法



第五章 广义表

5.4 广义表的定义(1)

- 广义表是线性表的推广
  - 元素类型可以是原子，也可以是子表，习惯上用大写字母表示广义表的名称，小写字母表示原子。$LS = (a_1, a_2, \ldots, a_n)$
  - 应用举例：表处理语言LISP中，把广义表作为基本的数据结构
  - 表头：表中的第1个元素，即a_1
  - 表尾：除第1个元素外，其余元素组成的表，即$(a_2, \ldots, a_n)$



ADT GList{

数据对象： D={ e_i | i=1,2,…,n; n≥0； e_i∈ AtomSet 或 e_i∈ GList，AtomSet 为某个数据对象 }

数据关系： R1={ <e_{i-1}, e_i> | e_{i-1}, e_i∈ D, 2≤i≤n }

基本操作：

InitGList(&L);

操作结果：创建空的广义表 L。

CreateGList(&L, S);

初始条件：S 是广义表的书写形式串。

操作结果：由 S 创广义表 L。

DestroyGList(&L);

初始条件：广义表 L 存在。

操作结果：销毁广义表 L。

CopyGList(&T, L);

初始条件：广义表 L 存在。

操作结果：由广义表 L 复制得到广义表 T。

GListLength(L);

初始条件：广义表 L 存在。

操作结果：求广义表 L 的长度，即元素个数。

GListDepth(L);

初始条件：广义表 L 存在。

操作结果：求广义表 L 的深度。

GListEmpty(L);

初始条件：广义表 L 存在。

操作结果：判定广义表 L 是否为空。

GetHead(L);

初始条件：广义表 L 存在。

操作结果：取广义表 L 的头。

GetTail(L);

初始条件：广义表 L 存在。

操作结果：取广义表 L 的尾。

InsertFirst_GL(&L, e);

初始条件：广义表 L 存在。

操作结果：插入元素 e 作为广义表 L 的第一个元素。

DeleteFirst_GL(&L, &e);

初始条件：广义表 L 存在。

操作结果：删除广义表 L 的第一个元素，并用 e 返回其值。

Traverse_GL(L, Visit());

初始条件：广义表 L 存在。

操作结果：遍历广义表 L，用函数 Visit 处理每个元素。

}**ADT** GList

5.4 广义表的定义(2)

- 示例

  A = ( )　　//A是一个空表，A的长度为0

  B = (e)　　// 表B只有一个原子e，B的长度为1

  C = (a, (b, c, d))　//C的长度为2, 第1个元素为原子，第2个元素为子表

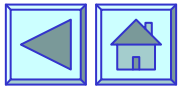
  D = (A, B, C)　　//D的长度为3, 3个元素都是子表

  E = (a, E)　　// E是递归的表，它的长度为2

  GetHead(B) = e,　　GetTail(B) = ( )

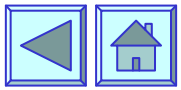
  GetHead(D) = A,　　GetTail(D) = (B, C)

  GetHead((( ))) = ( ),　　GetTail((( ))) = ( )



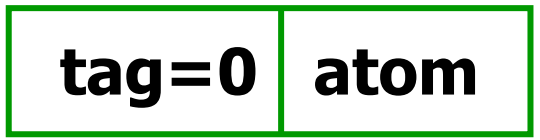
5.4 广义表的定义(3)

- 广义表是线性表的推广
  - 表可以为其它表所共享(P108, D、A)
  - 表可以是一个递归的表(P108, E)
- 广义表还可以看成是图的推广



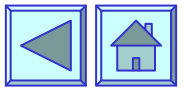
5.5 广义表的存储结构

- 广义表难以用顺序存储结构表示
  (∵数据元素可以具有不同的结构)
- 头尾链表存储表示
  - 结点类型
    - 原子结点：原子值
    - 表结点：表头、表尾
  - 类型定义 P109



5.5 广义表的存储结构

- 扩展线性链表存储表示
  - 结点类型
    - 原子结点
      原子值、下一结点的指针
    - 表结点
      表的头指针、下一结点的指针
  - 类型定义 P110

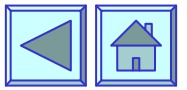


5.6 m元多项式的表示

- 一元多项式的表示：线性表
  元素为(exp, coef) exp-指数域，coef-系数域
- m元多项式
  - 将m个变元看成有主次之分
  - 一个m元多项式首先是其主变元的多项式，而其系数又是第二个变元的多项式
  - 结点类型
    - 原子结点：指数、系数、下一结点的指针

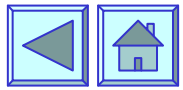
    - 表结点：指数、系数子表、下一结点的指针

  - 类型定义 P111



5.7 广义表的递归算法(1)

- 递归定义
  - 基本项：描述一个或几个递归过程的终结状态．
    终结状态：不需要继续递归而可直接求解的状态．
  - 归纳项：描述如何实现从当前状态到终结状态的转化．
- 递归函数的设计：归纳思维
  - 首先应书写函数头和规格说明，严格定义函数的功能和接口，对求精函数中所得的和原问题性质相同的子问题，只要接口一致，便可进行递归调用．
  - 对函数中的每个递归调用都看成只是一个简单的操作，只要接口一致，必能实现规格说明中定义的功能，切忌想得太深太远．



5.7 广义表的递归算法(2)

- 求广义表的深度
  - 广义表的深度：广义表中括号的重数

    $LS = (a_1, a_2, \ldots, a_n)$
  - 基本项 $DEPTH(LS) = 1$ 当LS为空表时

    $DEPTH(LS) = 0$ 当LS为原子时
  - 归纳项

    $DEPTH(LS) = 1 + Max_{1 \le i \le n} \{DEPTH(a_i)\}$ $n \ge 1$
  - 算法实现
    - 基于头尾链表存储结构 算法5.5 P114
    - 基于扩展线性链表存储结构

5.7 广义表的递归算法(3)

- 复制广义表
  - 基本项 InitGList(NEWLS) 当LS为空表时
  - 归纳项 当LS非空时

    CopyGList(GetHead(LS), GetHead(NEWLS))

    CopyGList(GetTail(LS), GetTail(NEWLS))
  - 算法实现
    - 基于头尾链表存储结构 算法5.6 P115
    - 基于扩展线性链表存储结构
- 建立广义表的存储结构