数 据 结 构

王　　勇

计算机/软件学院 大数据分析与信息安全团队

21#518　　TEL/**WeChat**:13604889411　　QQ:75767923

Email：　wangyongcs@hrbeu.edu.cn

哈爾濱工程大學
Harbin Engineering University

《数据结构》课程简介

1. 课程性质
2. 教学内容
3. 教学目标及考核
4. 前期课程与后续课程
5. 主要参考书目

课程性质与教学目的

课程性质

操作系统、数据库、编译原理等

数据结构课程对培养目标、毕业要求的支撑

培养目标

培养德智体全面发展，掌握自然科学、人文科学与数理基础知识，系统掌握软件工程领域基本理论、基本技术和应用知识，具备软件工程科学研究、工程实践开发和应用服务能力，具有系统的科学思维方法、工程实践技术、工程创新能力，能够综合运用学科基础知识与工程技术解决软件工程及交叉领域的复杂工程问题；未来五年内，培养的毕业生能够紧跟软件工程领域的国际发展前沿，具备国际视野与合作能力，是满足软件工程及交叉领域的项目管理、产品研发及理论研究等岗位需求的复合型专门人才。

数据结构

1-2掌握软件工程的专业基础理论和相关方法，并具有将其应用于工程问题的能力；——毕业要求1：工程知识

2-2掌握信息系统中离散量的结构和相互间的关系，具备识别和表达软件工程领域中的软件建模分析、系统设计等复杂工程问题的能力；——毕业要求2：问题分析

4-2掌握相关实验方法，具备设计、实施相关实验并能够对实验数据进行具体分析的能力；——毕业要求4：研究

《数据结构》课程简介

1. 课程性质
2. 教学内容
3. 教学目标及考核
4. 前期课程与后续课程
5. 主要参考书目

教学内容

绪论

组织、存储

线性数据结构

树形数据结构

图状数据结构

查　　找

内部排序

文件结构

课程学习主要内容 Contents

《数据结构》课程简介

1. 课程性质
2. 教学内容
3. 教学目标及考核
4. 前期课程与后续课程
5. 主要参考书目

数据结构课程教学目标

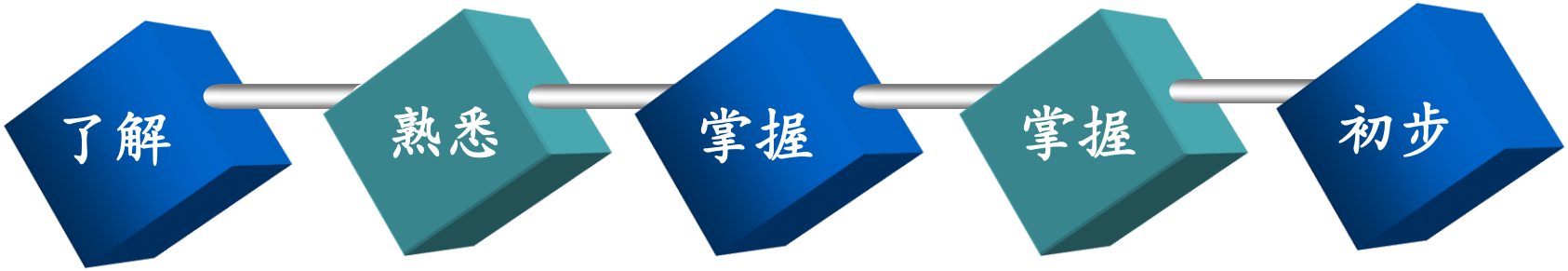
了解数据的基本概念，掌握数据及其结构在计算机内的表示、关系、操作

掌握典型数据结构的定义、分类、特点等

数据结构

掌握查找、排序等常用典型算法，培养运用数据结构进行程序设计与算法分析的能力

课程教学目标



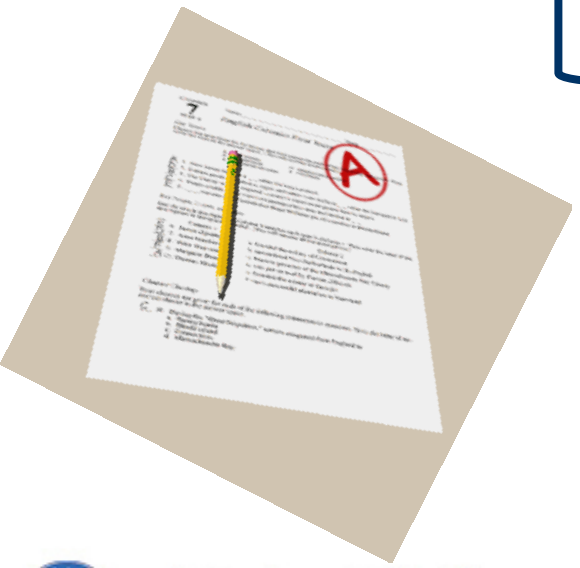
了解	熟悉	掌握	掌握	初步
• 数据结构及其分类 • 数据结构与算法的密切关系	• 基本数据结构及其操作 • 根据实际问题要求来选择数据结构	• 经典算法的思想和代码 • 设计算法的步骤 • 算法分析	• 数据结构在排序和查找等常用算法中的应用算法设计	• 文件组织方法 • 索引技术

认真完成预习、课堂、作业、复习

考核方式

30%

- 平时情况——出勤情况与课堂表现
- 课后作业——习题、研讨（小组）
- 期中测验——基础知识、综合问题
- 期末考试——笔试、闭卷 → 70%



《数据结构》课程简介

1. 课程性质
2. 教学内容
3. 教学目标及考核
4. 前期课程与后续课程
5. 主要参考书目

前期课程与后续课程

操作系统、数据库原理、编译原理等

很重要！复习和练习

数据结构

C语言

计算机基础

离散数学

《数据结构》课程简介

1. 课程性质
2. 教学内容
3. 教学目标及考核
4. 前期课程与后续课程
5. 主要参考书目

参考书目

- 严蔚敏等著《数据结构》C语言 清华大学出版社
- 范策等著《算法与数据结构》 机械工业出版社
- 李春保《数据结构与习题解析》清华大学出版
- （严蔚敏）数据结构视频教程（C语言版48集）

《数据结构》课程之问

- **是什么？**

数据及其关系；对象、关系与操作；非数值程序设计问题；可以理解为数据类型。

- **为什么？**

专业基础且核心课程；全面系统培养计算思维和提高程序设计能力；

- **学什么？**

基本概念；典型数据结构；经典算法；计算思维。

- **怎么学？**

C语言、程序设计是基础；理论与实验并重，实验验证性同步进行；基础、内容和课时因素，采用回顾、问题牵引、过程式考核（研讨、情景、翻转）

《数据结构》课程之问—个人观点供讨论

- 课程学习的数据结构与当前数据管理领域的结构有无区别和联系？

有联系也有区别；联系：两者之间都是研究数据及其结构，在微观上有部分概念相同，如二维数组等。

区别：当前数据管理领域结构分为结构化数据、半结构化数据和非结构化数据，如关系数据库数据、HTML，JASON等、文档等，是从宏观上研究数据的存储、管理和应用的角度，课程学习数据结构的内容更多关注微观上数据处理的内容。

《数据结构》课程之问—个人观点供讨论

- 数据结构与大数据之间的关系？

首先，两者研究侧重点不同，前者是数据及关系，偏向结构，后者强调数量的大量、多样、高速、高维、价值；

其次，前者更强调微观，基本在内存中，用控制台程序研究，后者更加宏观，需要巨大的存储空间和专门的平台，如Hadoop平台下的Hase、Hive，还有Mongodb、Redis等；

最后，结构分类体系不同，线性、非线性等，后者更加宏观，更加关注的是非结构化数据。

第一章 绪论

王　　勇

计算机学院 软件与数据科学研究所

21#533　　电　话：13604889411

课程网站：**http://cstcsjjg.hrbeu.edu.cn**

Email：　wangyongcs@hrbeu.edu.cn

哈尔滨工程大学

Harbin Engineering University

本章内容

第一章 绪论 本章说明

知识点	重　点	难　点
数据 数据元素 数据结构 数据类型 抽象数据类型 算法 算法设计原则 时间复杂度 空间复杂度	名词和术语 时间复杂度	无

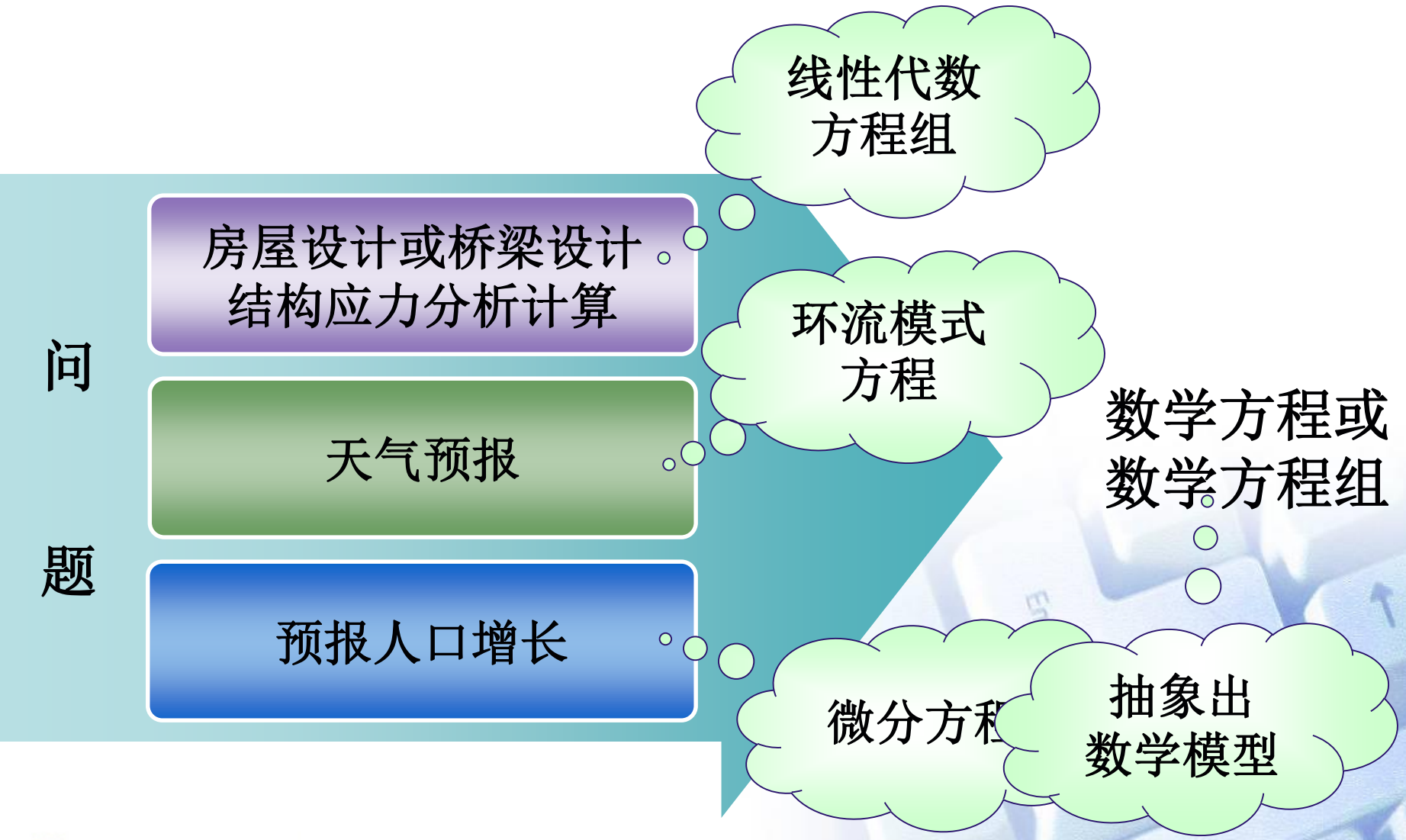
什么是数据结构

解决

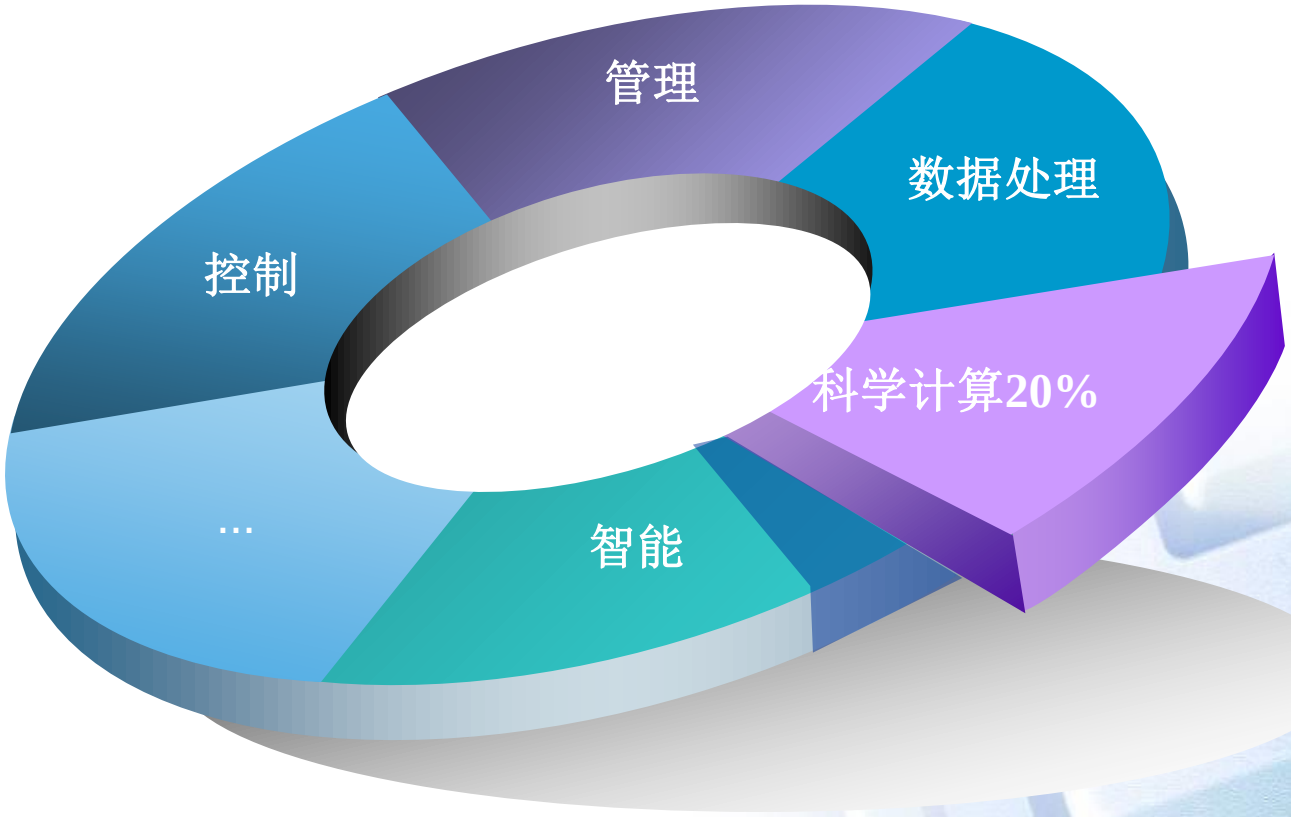
寻求数学模型的实质：

分析问题，从中提取操作的对象，并找出这些操作对象之间含有的关系，然后用数学的语言加以描述。

什么是数据结构


问
题
房屋设计或桥梁设计
结构应力分析计算
天气预报
预报人口增长
线性代数
方程组
环流模式
方程
微分方程
数学方程或
数学方程组
抽象出
数学模型

非数值计算的处理80%



本课程要讨论的数据结构

非数值计算问题的数学模型

什么是数据结构

- 例1-1 书目检索自动化问题

书目卡片

登录号：

书名：

作者名：

分类号：

出版单位：

出版时间：

价格：

什么是数据结构

◆ 例1-1 书目检索自动化问题

线性表

目文件

001	高等数学	樊映川	S01
002	理论力学	罗远祥	L01
003	高等数学	华罗庚	S01
004	线性代数	栾汝书	S02
……	…	……	……

索引表

按书名

高等数学	001， 003 ……
理论力学	002， ……..
线性代数	004， ……
……	……..

按作者名

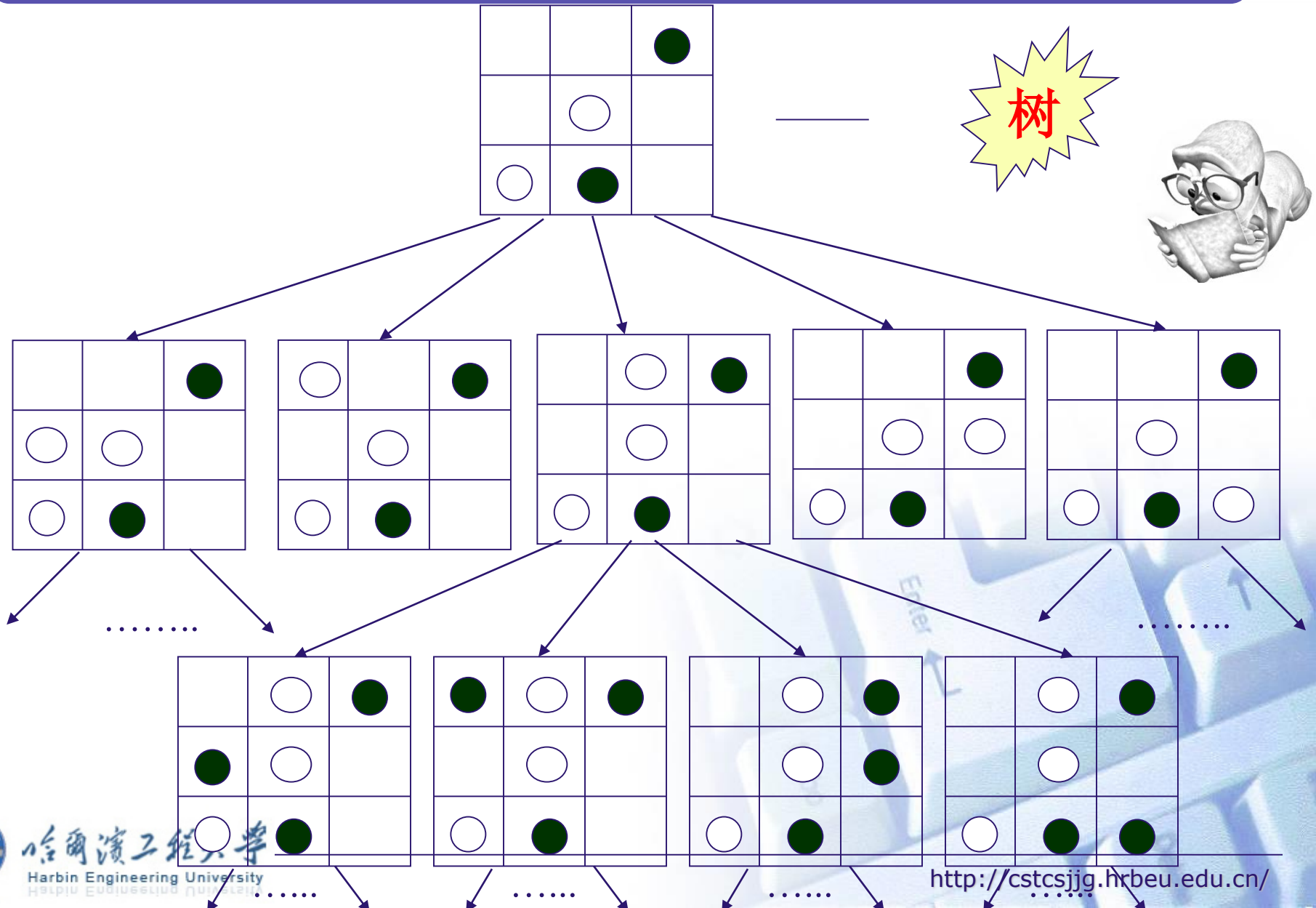
樊映川	001, …
华罗庚	002, ….
栾汝书	004, ….
…….	…….

按分类号

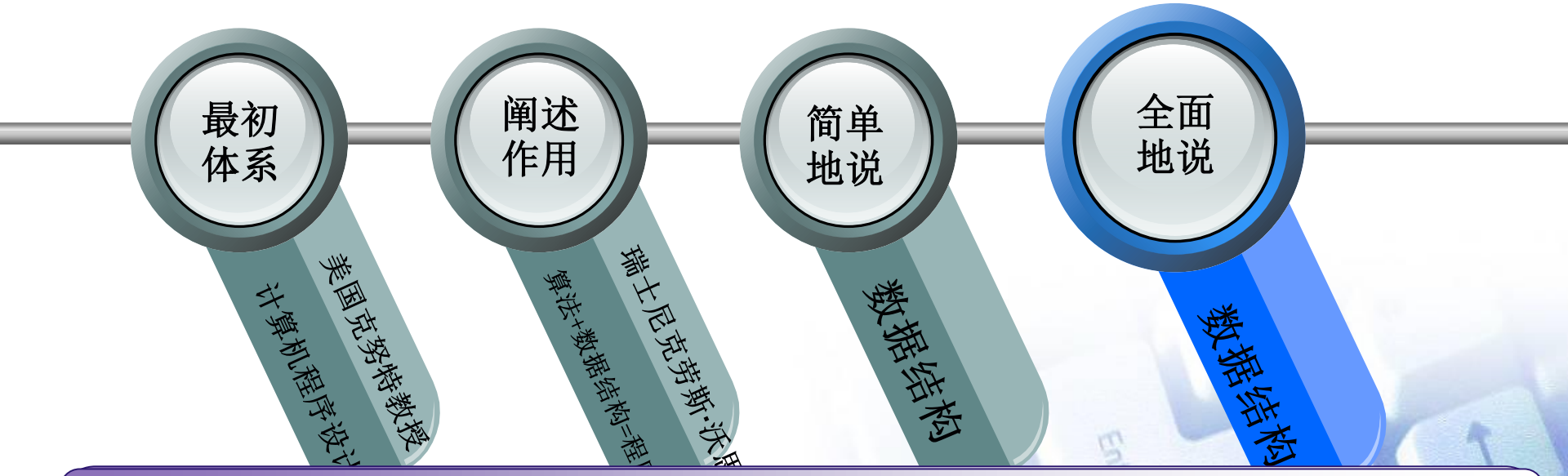
L	002, …
S	001, 003,
……	……

计算机处理的对象之间存在着线性关系，称为线性的数据结构

例1-2 人机对奕问题



什么是数据结构



是一门综合性的专业课程，是一门介于数学、计算机硬件、计算机软件之间的一门核心课程。是设计和实现编译系统、操作系统、数据库系统及其他系统程序和大型应用程序的基础

什么是数据结构

数据结构

是一门研究非数值计算的程序设计问题中计算机的操作对象以及它们之间的关系和操作等的学科

研究讨论

- 数据集合中各数据元素之间所固有的逻辑关系，即数据的逻辑结构；
- 在对数据进行处理时，各数据元素在计算机中的存储关系，即数据的存储（物理）结构；
- 对各种数据结构进行的运算

本章内容

1. 数据结构
2. 基本概念和术语
3. 抽象数据类型的表示和实现
4. 算法和算法分析
5. 本章小结

基本概念和术语

数　　据

是指所有能输入到计算机中并被计算机程序处理的符号的总称，是计算机加工的“原料”。

数据元素

是数据的基本单位，在计算机程序中通常作为一个整体进行考虑和处理如：记录、格局、顶点

数 据 项

数据项是数据元素的不可分割的最小单位。

关 键 字

能识别一个或多个数据元素的数据项。主次关键字

数据对象

是性质相同的数据元素的集合，是数据的一个子集

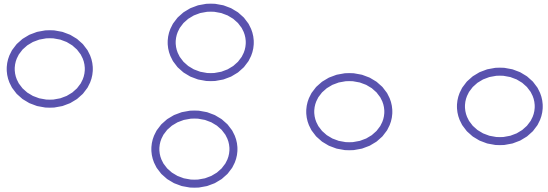
例 如

整数数据对象是集合N={0，±1，±2，……}

字母字符数据对象是集合C={“A”，“B”，…，“Z”}

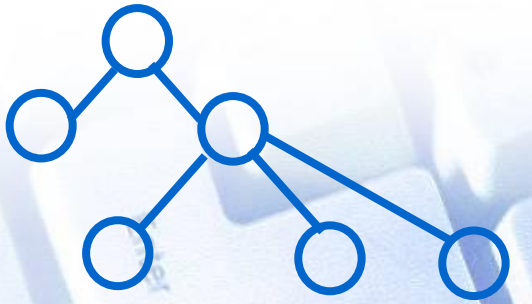
四种基本数据结构：

- 集　合：数据元素间除“同属于一个集合”外，“无其它关系”



- 线性结构：一个对一个，如线性表、栈、队列

- 树形结构：一个对多个，如树



- 图状结构：一个或多个对多个，如图

形式定义

数据结构是一个二元组

Data_Structure=(D,S)

其中：D是数据元素的有限集，S是D上关系的有限集

例 如

复数 Complex=(C,R)

C={c1,c2} ,　R={<c1,c2>}

图形表示

list=(D,R)

其中：D={1,2,3,4,5,6,7}

R={<1,2>,<2,3>,<3,4>,<4,5>,<5,6>,<6,7>}

逻辑（抽象）结构

只抽象反映数据元素之间的逻辑关系。可以用一个数据元素的集合和定义在此集合上的若干关系表示

存储（物理）结构

数据的逻辑结构在计算机存储器（内存）中的实现

位

在计算机中表示信息的最小单位，二进制数的一位

位串

位串是由若干个位组合表示一个数据元素（元素/结点）；

顺序存储结构

next

借助元素在存储器中的相对位置来表示数据元素间的逻辑关系

Click

链式存储结构

借助指示元素存储地址的指针表示数据元素间的逻辑关系

Click

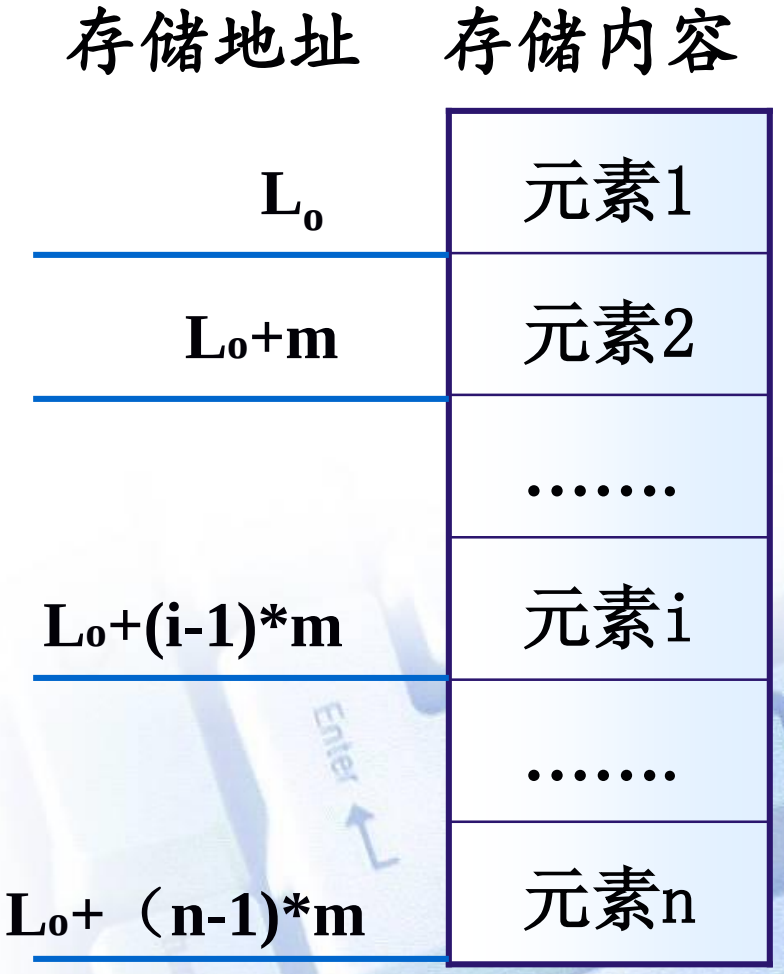
数据的逻辑结构与存储结构密切相关

算法设计　　取决于选定的逻辑结构

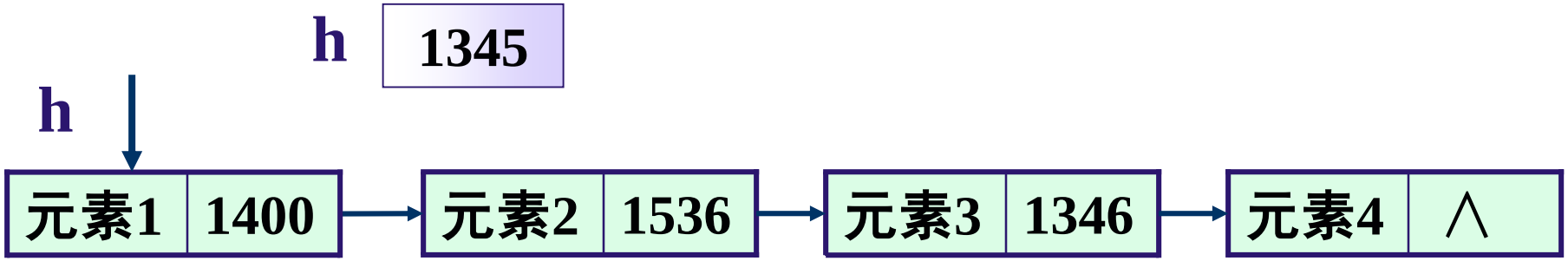
算法实现　　依赖于采用的存储结构

顺序存储

Back



$$Loc(元素i)=Loc+(i-1)*m$$



链式存储

存储地址	存储内容	指针
1345	元素1	1400
1346	元素4	∧
…….	……..	…….
1400	元素2	1536
…….	……..	…….
1536	元素3	1346

Back

数据结构的三个方面

- 数据的逻辑结构
  - 线性结构
    - 线性表
    - 栈
    - 队
  - 非线性结构
    - 树形结构
    - 图形结构
- 数据的存储结构
  - 顺序存储
  - 链式存储
- 数据的运算 → 检索、排序、插入、删除、修改等

本章内容

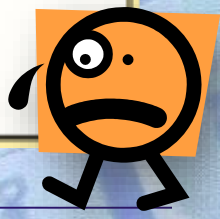
1. 数据结构
2. 基本概念和术语
3. 抽象数据类型的表示和实现
4. 算法和算法分析
5. 本章小结

数据类型

是一个值的集合和定义在这个值集上的所有的操作，如整型。

可分为

- 原子类型的值是不可分解的。如整型，实型
- 结构类型的值是由若干成分按某种结构组成的。如：数组、结构体;抽象数据类型基本都是结构类型



数据类型

在高级语言中指数据的取值范围及其上可进行的操作的总称

例　如

C语言提供：int, char, float, double等基本数据类型，数组、结构体、共用体、枚举等构造数据类型，还有指针、空(void)类型等。用户也可用typedef 自己定义数据类型

计算机领域，数据类型可以看成是已经实现了的数据结构

```
typedef   struct
{       int   num;
        char  name[20];
        float   score;
}STUDENT;
STUDENT    stu1,stu2, *p;
```

抽象数据类型

是指一个数学模型以及定义在该模型上的一组操作。（载体（万能）、通用、非具体的）

实 质 上

抽象数据类型和数据类型是一个概念，它仅取决于数据类型的逻辑性，而与其在计算机内部如何表示和实现是无关的

1.抽象数据类型的定义由一个值域和定义在该值域上的一组操作组成；
2.抽象数据类型是相对于基本的、具体的、已经实现了的数据类型的角度来讲，数据对象一般是结构体或者不确定的复杂类型

抽象数据类型格式

ADT 抽象数据类型名 {
　　数据对象：〈数据对象的定义〉
　　数据关系：〈数据关系的定义〉
　　基本操作：〈基本操作的定义〉
} ADT 抽象数据类型名

其中：数据对象和数据关系的定义用伪码描述。

基本操作的定义格式：

　　基本操作名（参数表）
　　初始条件：〈初始条件描述〉
　　操作结果：〈操作结果描述〉

两种参数：赋值参数只为操作提供输入值；引用参数以&开头，除提供输入值外，还返回操作结果

例1-6　抽象数据类型三元组的定义

ADT Triplet {

数据对象：D={e1,e2,e3 |e1,e2,e3∈Elemset
(定义了关系运算的某个集合)

数据关系：R1={<e1,e2>,<e2,e3>〉

基本操作：

InitTriplet(&T,v1,v2,v3)//构造三元组T，
e1,e2,e3分别被赋以参数v1,v2,v3的值

DestroyTriplet（&T）//撤消三元组T

Get(T,i,&e)//三元组T已存在，1=<i<=3
　　用e返回第i元的值
Put(&T,i,e) //三元组T已存在，1=<i<=3
　　将T的第i元值修改为e
IsAscending(T)//三元组T已存在，1=<i<=3
　　三元素若升序排列返回1，否则返回0
IsDescending(T)
Max(T,&e)
Min(T,&e)

} ADT Triplet

抽象数据类型的表示与实现

抽象数据类型往往用来抽象、描述现实世界中复杂的问题，可通过固有数据类型来表示和实现，即利用处理器中已存在的数据类型来说明新的结构，用已经实现的操作来组合新的操作。

抽象数据类型的表示与实现

- 预定义常量和类型
- 数据结构
  - 表示（存储结构）用类型（Typedef）定义来描述
  - 数据元素类型约定为ElemType，由用户在使用时自行定义
- 基本操作的算法用函数描述
- 函数类型 函数名（函数参数表）{

  //算法说明

  语句序列

  }//函数名

抽象数据类型的表示与实现

- 赋值语句
  - 简单赋值 变量名=表达式;
  - 串联赋值 变量名1=变量名2=...=变量名k=表达式;
  - 成组赋值 (变量名1，...，变量名k)=（表达式1，...，表达式k);

    结构名=结构名;

    结构名=（值1，...，值k);

    变量名[]=表达式;

    变量名[起始下标...终止下标]=变量名[起始下标...终止下标];
  - 交换赋值 变量名↔变量名;
  - 条件赋值 变量名=条件表达式？表达式T：表达式F;

抽象数据类型的表示与实现

- 选择语句
  - 条件语句1 if(表达式) 语句；
  - 条件语句2 if(表达式) 语句；
    else 语句；
  - 开关语句1 switch(表达式){
    case 值1：语句序列1；break;
    …
    case 值n：语句序列n；break;
    default：语句序列n+1; }
  - 开关语句2 switch(表达式){
    case 条件1：语句序列1；break;
    …
    case 条件n：语句序列n；break;
    default：语句序列n+1; }

抽象数据类型的表示与实现

- 循环语句
  - for语句 for(赋初值表达式序列；条件；修改表达式序列)　语句；
  - while语句　　while(条件)　语句；
  - do-while语句　do{
    语句序列；
    }while (条件)；
- 结束语句
  - 函数结束语句　return　表达式；
    return；
  - case结束语句　break；
  - 异常结束语句　exit(异常代码)；

抽象数据类型的表示与实现

- 输入输出语句
  - 输入语句 scanf([格式串]，变量 1 ，…变量n);
  - 输出语句 printf([格式串]，表达式 1 ，…表达式n);
- 注释语句
  - 单行注释 //文字序列
- 逻辑运算约定
  - 与运算&& 对于A&&B，当A的值为 0 时，不再对B求值
  - 或运算|| 对于A||B，当A的值为非 0 时，不再对B求值

抽象数据类型的表示与实现

- 基本函数
  - 求最大值　　max(表达式1，…表达式n)
  - 求最小值　　min(表达式1，…表达式n)
  - 求绝对值　　abs(表达式)
  - 求不足整数值　floor(表达式)
  - 求进位整数值　ceil(表达式)
  - 判定文件结束　eof(文件变量)或eof
  - 判定行结束　　elon(文件变量)或eoln

抽象数据类型的表示与实现

例　如　三元组Triplet的表示和实现

//--采用顺序存储结构

Typedef ElemType *Triplet;//由InitTriplet分配三个元素存储空间

//--基本操作的函数原型说明

Status InitTriplet(Triplet &T, ElemType v1,
ElemType v2, ElemType v3)

Status DestroyTriplet（&T）

Status Get（T, i, &e）

抽象数据类型的表示与实现

```
Status Put(&T, i, e)
Status IsAscending(T)
Status IsDescending(T)
Status Max(T, &e)
Status Min(T, &e)
//--基本操作的实现
Status InitTriplet(Triplet &T, ElemType v1,
    ElemType v2, ElemType v3){
//构造三元组T，依次置T的三个元素的处值为v1,
    v2和v3
```

抽象数据类型的表示与实现

```
T=(ElemType*)malloc(3*sizeof(ElemType));
                //分配三个元素的存储空间
If(!T)exit(OVERFLOW);//分配存储空间失败
T[0]=v1; T[1]=v2; T[2]=v3;
return OK;
}//InitTriplet
Status DestroyTriplet（Triplet &T）{//销毁T
   free(T);  T=NULL;  return OK;
}//DestroyTriplet
```

抽象数据类型的表示与实现

```
Status Get（Triplet T, int i, ElemType &e）{
    //1=<i<=3,用e返回T的第i元的值
    if (i<1 || i>3) return ERROR;
    e=T[i-1];
    return OK;
}//Get
Status Put（Triplet & T, int i, ElemType e）{
    //1=<i<=3，置T的第i元的值为e，用T返回
    if (i<1 || i>3) return ERROR;
    T[i-1]=e;
    return OK;
}//Put
```

抽象数据类型的表示与实现

```
Status Max（Triplet T, ElemType &e）{
   //用e返回指向T的最大元素值
   e=(T[0]>=T[1])?((T[0]>=T[2])?T[0]:T[2])
                 :((T[1]>=T[2])?T[1]:T[2]);
   return OK;
}//Max
```

表达式1？表达式2：表达式3

当表达式1成立时其值为表达式2的值，

否则，其值为表达式3的值

例：T[0]=3, T[1]=6, T[2]=2 ， e=6

本章内容

1. 数据结构
2. 基本概念和术语
3. 抽象数据类型的表示和实现
4. 算法和算法分析
5. 本章小结

算法和算法分析

算　法

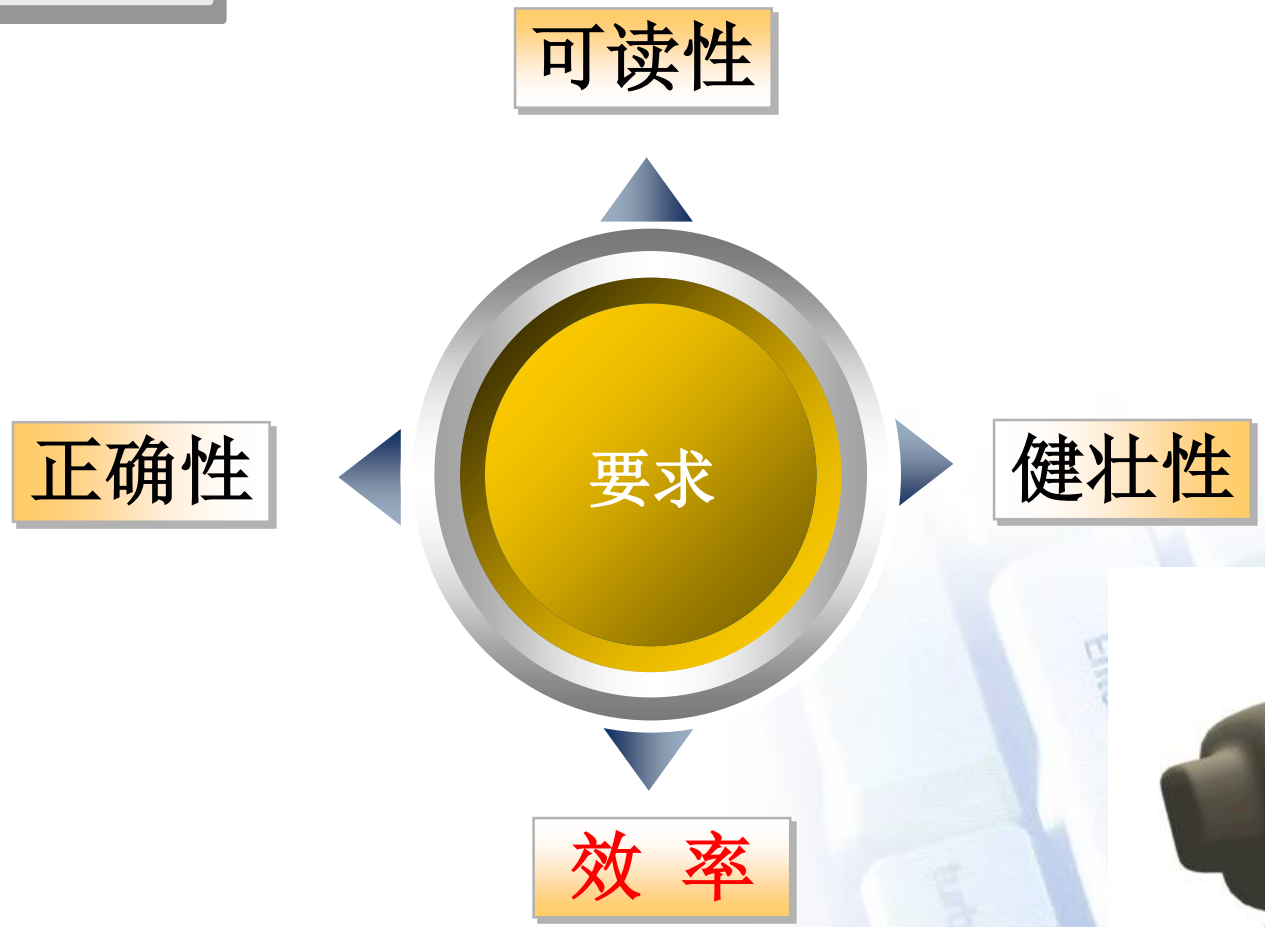
（Algorithm）是对特定问题求解步骤的一种描述，它是指令的有限序列

算法特性

- 可行性：算法是能实现的
- 确定性：算法的每一步必须是确切定义的，不能产生二义性
- 有穷性：算法必须在执行有限步骤后结束
- 输　　入：一个算法有零个或多个输入（如：排序、判断）
- 输　　出：一个算法有一个或多个输出

算法和算法分析

算法设计



算法效率

用依据该算法编制的程序在计算机上执行所消耗的时间来度量。（算法时间是由控制结构和原子操作的决定的）

记号 引用了“O”，表示一个 数量级的概念。

本书中根据算法中语句执行的最大次数（频度）来估算一个算法执行时间的数量级。

时间复杂度

基本操作重复执行的次数是问题规模n的某个函数F(n)：T（n）=O（F（n））问题规模n增大则执行时间增大

语句的频度 是该语句重复执行的次数。

例 如　求两个n阶方阵的乘积C=A×B

```
#define n 100
void MatrixMultiply(int A[n][n], int B[n][n], int C[n][n])
{   int i, j, k
    for (i=1; i<=n; ++i)                    //n+1
        for (j=1; j<=n; ++j)                 // n*(n+1)
          { C[i][j]=0;                       //n^2
              for (k=1; k<=n, k++)           //n^2* (n+1)
                  C[i][j]=C[i][j]+A [i][k] *B [k][j];   //n^3
          }
}
```

$T(n)=n+1+n*(n+1)+n^2+n^2*(n+1)+n^3=2n^3+3n^2+2n+1$

量级：$T(n)=O(n^3)$

例 如

```
（a）{++x; s=0;}
（b）for (i=1;i<=n;++i)
        {++x; s+=x;}
（c）for (j=1;j<=n;++j)
        for (k=1;k<=n;k++)
          {++x; s+=x;}
```

含基本操作“x增1”的语句的频度分别为1,n和n^2

时间复杂度是O(1)、O(n)和$O(n^2)$。

时间复杂度有时与输入有关。

算法存储量

指的是算法执行过程中所需的最大存储空间。

- 输入数据所占空间
- 程序本身所占空间
- 辅助变量所占空间
- 若算法所需存储量依赖于特定的输入，则通常按

最坏情况考虑

本章内容

本章小结

- 数据是计算机操作对象的总称，它是计算机处理的符号的集合，集合中的个体为一个数据元素
- 数据元素可以是不可分割的原子，也可以由若干数据项合成，因此在数据结构中讨论的基本单位是数据元素，而最小单位是数据项
- 数据结构是由若干特性相同的数据元素构成的集合，且在集合上存在一种或多种关系。由关系不同可将数据结构分为四类：集合结构、线性结构、树形结构和图状结构

本章小结

- 数据的存储结构是数据逻辑结构在计算机中的映象，由关系的两种映象方法可得到两类存储结构：一类是顺序存储结构，它以数据元素相对的存储位置表示关系，则存储结构中只包含数据元素本身的信息；另一类是链式存储结构，它以附加的指针信息（后继元素的存储地址）表示关系
- 抽象数据类型是一个数学模型以及定义在该模型上的一组操作。抽象数据类型的三大要素为数据对象、数据关系和基本操作，同时数据抽象和数据封装是抽象数据类型的两个重要特性

本章小结

- 算法是对问题求解的一种描述，是为解决一个或一类问题给出的一种确定规则的描述。一个完整的算法应该具有下列五个要素：有穷性、确定性、可行性、有输入和有输出。一个正确的算法应对苛刻且带有刁难性的输入数据也能得出正确的结果，并且对不正确的输入也能作出正确的反映。
- 算法的时间复杂度的估算基于算法中基本操作的重复执行次数，或处于最深层循环内的语句的频度。算法空间复杂度主要取决于算法的输入量和辅助变量所占空间。

继续学习下一章！