**第一章 绪 论**

**一、教学基本要求**

1．掌握数据结构中有关基本概念。

2．掌握算法的概念，理解算法设计的要求及如何衡量算法效率。

3．学会分析算法的时间复杂度。

重点：了解数据结构的逻辑结构、存储结构及数据的运算三方面的概念及其相互关系

难点：抽象数据类型和算法复杂度的分析方法。

**二、学时安排**

讲课学时：2学时

**三、教学内容**

1.什么是数据结构

我们知道无论使用计算机做工程计算，还是进行工业控制或信息管理，都属于数据处理的范畴。那么如何在计算机中组织、存储、传递数据就成为一个必须解决的问题，下面我们看几个例子。

(1)图书馆的书目检索自动化问题----计算机处理的每一个对象是一种图书信息，而在这些图书信息之间存在着线性关系，称为线性的数据结构。

(2)人机对奕问题----计算机处理的对象是一个个格局。每一种格局之后都可能再分出几种格局，所有可能出现的格局是一棵倒置的树。这种结构叫做树型结构。

(3)多岔路口交通灯的管理问题----书上图1.3，总共有五个路口，(其中有两个路口只能单向行驶)，经过计算这五个路口共有13条通路，(这个问题是很容易算出来的，请大家想一想如果有100个路口，会有多少个通路。) 将每一条通路看作一个节点，用圆圈表示，如果两条通路不能同时通行，就用直线将两个节点相连，那么这个问题变成了一个节点染色的问题，即要求图的每个顶点染一种颜色，而有连线的顶点不能染同样的颜色，要求所染颜色最少，那么这个问题就变成为数学模型是图的数据结构。

2、《数据结构》课程

数据结构：是一门研究非数值计算程序设计问题中计算机操作对象以及它们之间关系和操作的一门学科。（三个要素：对象、关系及操作（运算））许多学校计算机系研究生入学考试有一门就是数据结构，而且大多数学校使用的是这本教材。

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、数据库系统机其他系统程序和大型应用程序的基础。

二、基本概念和术语

1.数据

声音、图象、字符等各种信息要想被计算机处理，必须先转换成数据，那么什么是数据呢，我们给数据下一个定义。

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”。

事实上我们平时所用到的数据主要有两类，一类是数值型数据，如整数，实数，浮点数，主要用于工程和科学计算中，而另一类是非数值数据，主要包括字符和字符串，以及文字、图形、语音等数据。

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。

数据项:有时，一个数据元素可由多个数据项组成。数据项是数据的不可分割的最小单位。

2.数据对象、数据结构

在日常所遇到的问题中，总把数据按其性质归类到一些称之为数据对象的集合中，在数据对象中所有数据成员(数据元素)，都具有相同的性质，数据对象是数据的子集。下面给出数据对象的定义。

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。

如果综合考虑数据对象以数据对象中所有数据元素的关系，就得到了数据结构的定义。

数据结构：是相互之间存在一种或多种特定关系的数据元素的集合。

四类基本结构：集合、线性结构、树形结构、图形结构或网状结构。

数据结构的形式定义：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：课题小组 Group=(P,R)

P={T,G1,…,Gn,S11,…,Snm}1≤n≤3，1≤m≤2，

R={R1,R2}

R1={<T,Gi> |1≤i≤n, 1≤n≤3,}

R2={<Gi,Sij> |1≤i≤n, 1≤j≤m,1≤m≤2,}

数据结构一般包括三方面的内容：

① 逻辑结构：数据元素之间的逻辑关系。

② 存储结构(物理结构)：数据元素及其关系在计算机存储器的表示。

③ 数据的运算：对数据施加的操作。

算法的设计取决于选定的数据逻辑结构，而算法的实现依赖于采用的存储结构。

3.数据的两种存储结构：

顺序存储结构：把逻辑上相邻的结点存储在物理位置上相邻的存储单元里，结点间的逻辑关系由存储单元的邻接关系来体现。通常顺序存储结构是借助于语言的数组来描述的。

链式存储结构：不要求逻辑上相邻的结点物理上也相邻，结点间的逻辑关系是由附加的指针字段表示的，通常要借助于语言的指针类型来描述。

4.数据类型、抽象数据类型

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。

数据类型主要用于计算机高级语言和低级语言中，例如，高级语言中的整型、字符型、浮点型等。汇编语言中的位、字节、字等类型，这些数据类型定义了某些值的集合以及这些值所能进行的操作。

数据类型可分为：非结构的原子类型和结构类型。原子类型的值是不可分解的，结构类型的值是由若干成分按某种结构组成的。

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。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。抽象数据类型和数据类型实质上是一个概念，所谓的抽象本质就是反映事物本质的东西，而忽略非本质的细节，一般来说抽象数据类型是由用户自己定义的。

抽象数据类型按其值的不同特性，分为三种类型：

① 原子类型:变量的值是不可分解的。

②固定聚合类型:变量的值由确定数目的成分按某种结构组成。（数据类型的成分和数目均确定）

③可变聚合类型:其值的成分数目不确定。（成分的数目不确定，例如，可以定义一个“有序序列的整数的集合”，这个集合是由多个有序序列组成的，但是这个序列的长度即数目是不确定的。）

抽象数据类型的形式定义：我们用一个三元组来表示一个抽象数据类型。

（D，S，P）

D是数据对象， S是D上的关系集， P是对D的基本操作。

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。

数据对象和数据关系的定义用伪码描述。

数据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〉

例：

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)

DestroyTriplet（&T）

Get（T,i,&e）

Put(&T,i,e)

IsAscending(T)

IsDescending(T)

Max(T,&e)

Min(T,&e)

}ADT Triplet

       多形数据类型:是其值的成分不确定的数据类型。（其值成分不确定）

三、抽象数据类型的表示与实现

抽象数据类型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。

1.类C语言

精选了C 的一个子集，扩充修改，增强了语言的描述功能。

       预定义常量和类型

       数据结构的表示：存储结构用类型（typedef）定义来描述。

数据元素类型约定为ElemType.

       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

增加了引用调用的参数传递方式。赋值语句、选择语句、循环语句、结束语句输入输出语句、注释语句， 基本函数， 逻辑运算约定

例：Triplet的表示和实现

//采用动态分配的顺序存储结构

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）

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

【算法1.1】

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。

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）{

//销毁三元组T。

······

}//Min

四、算法和算法分析

1.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。

算法的描述有语言方式、流程图方式等等，本书主要用类C语言描述。

算法具有五个重要特性：有穷性、确定性、可行性、输入、输出

2.算法设计的要求

正确性、可读性、健壮性和效率与低存储量要求

3.算法效率的度量

一个算法所用的时间等于编译时间加上运行时间，算法所需要的编译时间和它所使用的编译程序有关，而且一个算法经编译后可以重复运行多次，我们我们讨论算法效率的时候只考虑算法的运行时间。

而一个算法的运行时间包括加、减、乘、除、存取等运算，要想计算算法运算所需要的总的时间是不可行的，因为一个算法的运行时间不仅依赖于硬件和操作环境，而且还和参与运算的数字的大小有关，并且和编译程序编译出来的目标代码有关，所以度量算法的运行时间主要考虑算法的程序步数。

但是程序步也不是一个精确的概念，比如赋值语句

x=a;

和x=a+b\*(a+d);

具有相同的程序步数，但是它们的执行时间肯定不同，但是这里我们只要能得出一个数量级就可以了。所以我们提出了时间复杂的概念。

时间复杂度：算法中基本操作重复执行的次数是问题规模n的某个函数f（n），那么它的时间复杂度是：

T（n）=O（f（n））

它表示随问题规模n的增大，算法执行时间的增长率和f（n）的增长率相同。对于多项式来说，我们只保留最高次幂的项，而低次幂的项和常数项可以忽略不计，例如对于g(n)=2n3+3n2+n+1

当n充分在时，取时间复杂度为T(n)=O(n3),因为和n3相比，n2和n的值不起决定性的作用，可以忽略不计。

语句的频度：是该语句重复执行的次数。

例1：求两个n阶方阵的乘积C=A×B，其算法如下：

【算法1.2】

#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)

for (j=1;j<=n;++j)

C[i][j]=0;

for (k=1;k<=n,k++)

C[i][j]=C[i][j]+a[i][k]\*b[k][j];

}

}

以乘法做为基本操作，乘法执行的次数为n3，

T(n)=O(n3)

例：

（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n2

时间复杂度是O(1),O(n)和O(n2)。

时间复杂度有时与输入有关。自己看一下书上起泡法排序中输入数据不同时，时间复杂度也不同的情况。

4.算法的存储空间需求

当一个具体实例的n充分大时，需要的存储空间体积也将随之变化，也可以象分析时间复杂度一样，用大O法象分析时间复杂度一样分析算法的空间复杂度，通常只有完成同一功能的几个算法之间才具有可比性。

例如，同样是排序算法，待排序数据都是n个，作为存放这n个数据的数组或链表都是n个，因此它们所占用的存储空间不用比较，可以只比较那些辅助的或附加的存储空间。

空间复杂度：S（n）=O（f（n））

5.作业：用C语言编写程序：

1.输入系数a,b,c,求方程ax2+bx+c=0的解。（包括无解、单解和双解的情况）。

2.习题集1.3.4.5题