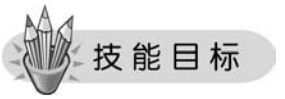
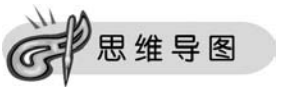
绪　论



- 会用数据结构的概念解释日常生活的问题。
- 掌握逻辑结构、存储结构等知识点。
- 能找到现实生活中线性、树和图三种结构的实例。
- 掌握算法的五个特性准则。
- 会用算法效率的评价指标评价算法的执行效率。



本模块思维导图请扫描右侧二维码。

绪论思维导图

计算机是对各种各样数据进行处理的机器。要对数据进行处理，首先要对数据进行组织。计算机中如何有效地组织数据和处理数据是一个关键的问题，而“数据结构与算法”就能解决这个问题。

1. 学习数据结构的意义

早期人们都感觉计算机是用来计算的，用来处理数值计算非常方便快捷。随着计算机处理能力越来越强，计算机不再只是简单的数值计算工具，而是被赋予越来越多的功能，而现实世界有很多非数值计算的问题，需要一些更科学有效手段的帮助才能更好地处理这些问题。所以数据结构是一门研究非数值计算的程序设计中计算机的操作对象以及它们之间的关系和操作等相关问题的学科。

针对实际问题，要编写出一个高效率的处理程序，就需要解决如何合理地组织数据，建立合适的数据结构，设计较好的算法，来提高程序执行效率这样的问题。数据结构和算法就是在此背景下形成和发展起来的。

简而言之，软件开发要多动脑筋，想到好的解决办法才能更快更好地编写出效率更高的程序。数据结构和算法这门课程的目的正是使学生更快地编写出更高效的程序。

计算机完成的任何操作都是在程序的控制下进行的，而程序的根本任务就是进行数据处理。随着计算机在各行各业应用的日益深入，计算机所处理的数据对象也由纯粹的数值型发展到字符、表格、图形、图像和声音等多种形式，计算机要处理这些数据，首先要将这些数据存储在计算机内存中。如何将程序中要处理的数据进行合理地存储，以及采用何种方法能够高效地进行数据处理是程序设计的关键，也是数据结构要解决的重要问题。

即使是在广泛采用可视化程序设计的今天，借助于集成开发环境可以很快地生成程序，但要想成为一个专业的程序开发人员，至少需要以下三个条件。

(1) 能够熟练地选择和设计各种业务逻辑的数据结构和算法。

(2) 至少能够熟练地掌握一门程序设计语言。

(3) 熟知所涉及的相关应用领域知识。

其中,后两个条件比较容易实现,而第一个条件则需要花很多时间和精力才能达到,但它恰恰是区分程序设计人员水平高低的一个重要标志。数据结构贯穿程序设计的始终,缺乏数据结构和算法的功底,很难设计出高水平的具有专业水准的应用程序。瑞士著名计算机科学家尼古拉斯·沃思(Niklaus Wirth)提出了"算法+数据结构=程序"的观点,这正说明了数据结构的重要性。

例如,计算机要处理一批杂乱无章的数据,需对其进行有序化处理,计算机解决问题的步骤是什么呢?

要解决这个问题,首先要考虑应如何将这批数据进行合理地存储;然后考虑应采用什么有效的方法对这些数据进行有序化;最后选择一种编程语言实现以上方法。其实用计算机解决任何数据处理的问题,都需要经过以上过程才能实现。

数据结构主要研究和讨论以下三个方面的问题。

(1) 数据集合中各数据元素之间的关系。

(2) 在对数据进行处理时,各数据元素在计算机中的存储关系,即数据的存储结构。

(3) 针对数据的存储结构进行的运算。

解决好以上问题,可以大大提高数据处理的效率。

微课 0-1 数据结构学什么

2. 数据结构研究什么

数据结构可定义为一个二元组:Data_Structure=(D,R),其中 D 表示数据元素的有限集,R 表示 D 关系上的有限集。数据结构与算法具体应包括以下方面:数据的逻辑结构及算法、数据的物理结构和数据的运算集合。

1) 数据的逻辑结构及算法

(1) 数据的逻辑结构。数据的逻辑结构是指数据元素之间逻辑关系的描述。

根据数据元素之间关系的特性,数据的逻辑结构主要有以下三种,如图 0-1 所示。

① 线性结构。结构中的数据元素之间存在着一对一的线性关系。线性结构将在模块 1 中详细讲解。

② 树结构。结构中的数据元素之间存在着一对多的层次关系。树结构将在模块 6 中详细讲解。

③ 图结构。结构中的数据元素之间存在着多对多的任意关系。图结构将在模块 7 中详细讲解。

下面重点介绍线性结构。

线性结构是最基本的结构,掌握好它是学习其他结构的基础。线性结构中几种典型的类型,分别是线性表、栈、队列、字符串和数组。

① 线性表是一种最简单、最常用的线性结构。线性表的操作特点主要是可以在任意位置插入一个数据元素或删除一个数据元素。线性表将在模块 1 中详细讲解。

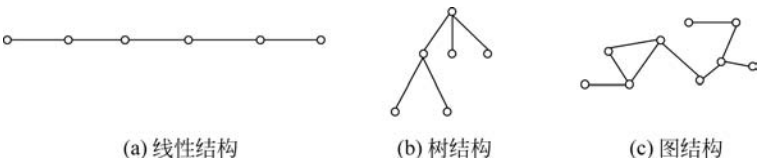


图 0-1　数据的逻辑结构

② 栈是一种只能在栈顶一端进行插入或删除操作的线性结构，它的特点是后进先出。栈结构将在模块 2 中详细讲解。

③ 队列是只能在队头删除队尾插入的线性结构，它的特点是先进先出。队列结构将在模块 3 中详细讲解。

④ 字符串的简称是串。串是一种特殊的线性结构，它的每个数据元素都由一个字符组成。串结构将在模块 4 中详细讲解。

⑤ 数组是程序设计最经常使用的一种数据结构，所有线性结构（包括线性表、栈、队列、串、数组和矩阵）的顺序存储结构都是使用数组来存储。可见，它是其他数据结构实现顺序存储结构的工具，将在模块 5 中详细讲解数组与矩阵。

（2）数据的算法。编程开创者瑞士计算机科学家尼古拉斯·沃斯的名言“算法＋数据结构＝程序”，很好地阐述了这三者的关系。数据结构和算法两个概念间的逻辑关系贯穿了整个程序世界，没有数据间的关系，程序根本无法设计。数据结构是底层，算法是高层；数据结构为算法提供服务，算法围绕数据结构操作。所以数据结构要和算法一起讲解才更有效。本书介绍了查找和排序两种基本的算法。

① 查找。数据结构要跟算法结合起来才有意义，查找算法是数据结构在算法中的应用，在现实生活中也经常用到查找。查找算法将在模块 8 中详细讲解。

② 排序。排序算法将在模块 9 中详细讲解。

2）数据的物理结构

物理结构（又称存储结构）是逻辑结构在计算机中的存储映像，是逻辑结构在计算机中的实现（或存储表示），它包括数据元素的表示和关系的表示。有数据结构 Data_Structure＝（D，R），对于 D 中的每一数据元素都对应着存储空间中的一个单元，D 中全部元素对应的存储空间必须明显或隐含地体现关系 R。逻辑结构与存储结构的关系为：存储结构是逻辑结构的映像与元素本身的映像。逻辑结构是抽象，存储结构是实现，两者综合起来建立了数据元素之间的结构关系。存储结构一般有顺序存储和链表存储两种方式。

3）数据的运算集合

讨论数据结构的目的是在计算机中实现所需的操作，施加于数据元素之上的一组操作构成了数据的运算集合，因此运算集合是数据结构很重要的组成部分。

3. 数据结构的基本概念

1）数据

数据是描述客观事物的数值、字符以及所有其他能输入计算机中，且能被计算机处理的各种符号的集合。简言之，数据就是存储在计算机中的信息。

数据不仅包括整型、实型等数值类型,还包括字符、声音、图像、视频等非数值类型。比如,现在常用的搜索引擎,一般会有网页、图片、音乐、视频等分类,音乐是声音数据;图片是图像数据;网页是全部数据的集合,包括数字字符和图像等数据。

2) 数据元素

数据元素是组成数据的基本单位,是数据集合的个体,在计算机中通常作为一个整体进行考虑和处理。

比如,在学生这个群体中,每一个学生就是数据元素。

3) 数据项

数据项是有独立含义的不可再分割的最小单位。一个数据元素可由一个或多个数据项组成,如每一个学生的信息是一个数据元素,它包含学号、姓名等多个数据项。

4) 数据对象

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 $\boldsymbol{N}=\{0,\pm1,\pm2,\cdots\}$,字母字符数据对象是集合 $\boldsymbol{C}=\{'A','B',\cdots,'Z'\}$。无论数据元素集合是无限集(如整数集)、有限集(如字符集),还是由多个数据项组成的复合数据元素,只要性质相同,都是同一个数据对象。

5) 数据类型

数据类型是一组性质相同的值集合以及定义在这个值集合上的一组操作的总称。值集合确定了该类型的取值范围,操作集合确定了该类型中允许使用的一组运算。例如,高级语言中的数据类型就是已经实现的数据结构。

6) 抽象数据类型

抽象数据类型是指基于一类逻辑关系的数据类型。抽象数据类型的定义取决于客观存在的一组逻辑特性,而其在计算机内如何表示和实现无关。

7) 数据结构

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。

在计算机中,数据元素之间是具有内在联系的数据集合。数据元素之间存在的关系,就是数据的组织形式。为编写出一个高效的程序,必须分析处理对象的特性及各对象之间的关系,这就是数据结构要研究的内容。

4. 算法及其描述

算法是解决问题的方法,是程序设计的精髓。程序设计的实质就是构造解决问题的算法。算法的设计取决于数据的逻辑结构,算法的实现取决于数据的物理存储结构。

1) 算法的概念和特性

算法是对特定问题求解步骤的一种描述,它是指令的有限序列。做任何事情都必须事先想好进行的步骤,然后按部就班地进行,才不会发生错误。计算机解决问题也是如此。对于一些常用的算法应该熟记,比如求阶乘、求素数、计算是否闰年等算法,在解决实际问题时,可参考已有的类似算法,按照业务逻辑设计出符合自己的算法。

一个算法应该具有以下五个重要特性。

(1) 有穷性。一个算法应包含有限的操作步骤,即一个算法在执行若干个步骤之后应该能够结束,而且每一步都在有限时间内完成。

(2) 确定性。算法中的每一步都必须有确切的含义,不能产生二义性。

(3) 可行性。算法中的每一个步骤都应该能有效地执行,并得到确定的结果。

(4) 输入。所谓输入,是指在算法执行时,从外界取得必要的数据。计算机运行程序的目的是进行数据处理,在大多数情况下,这些数据需要通过输入得到。有些情况下,数据已经包含在算法中,算法执行时不需要任何数据,所以一个算法可以有零个或多个输入。

(5) 输出。一个算法有一个或多个输出,这是算法进行数据处理后的结果。没有输出的算法是毫无意义的。

算法的这些特性可以约束程序设计人员正确地书写算法,并使之能够正确无误地执行,达到求解问题的预期效果。

提示:为了方便广大学生学习,本书所讨论的算法,全部用面向对象的 Java 语言为描述工具。

2) 算法设计的要求

算法设计的好坏关乎程序的执行效率,算法的设计必须满足下列四个要求。

(1) 正确性。正确性的含义是算法对于一切合法的输入数据都能够得出满足要求的结果,事实上要验证算法的正确性是极为困难的,因为通常情况下合法的输入数据量太大,用穷举法逐一验证是不现实的。所谓的算法正确性是指算法达到了测试要求。

(2) 可读性。算法的可读性是指人对算法阅读理解的难易程度,可读性高的算法便于交流,有利于算法的调试和修改。通常增加算法的可读性是在书写算法时采用按缩进格式书写、分模块书写等方法可增加算法的可读性。

(3) 健壮性。对于非法的输入数据,算法能给出相应的响应,而不是产生不可预料的后果。

(4) 效率与低存储量需求。效率是指算法的执行时间。对于解决同一问题的多个算法,执行时间短的算法效率高。存储容量需求指算法执行过程中所需要的最大存储空间。存储容量需求越小的算法效率越高。

3) 算法的分析

解决一个问题可以有多种算法,那么该怎样判断它们的优劣呢?判断算法优劣的标准很多,这里不做深入讨论,但一个算法除了正确性必须保证外,一个主要指标就是它的效率。

(1) 算法效率的度量。算法执行的时间是其对应的程序在计算机上运行所消耗的时间。程序在计算机上运行所需时间与下列因素有关:

① 算法本身选用的策略;

② 书写程序的语言;

③ 编译产生的代码质量;

④ 机器执行指令的速度;

⑤ 问题的规模。

第①条是算法好坏的根本,第②、③条要看具体的软件支持,第④条要看硬件的性能。度量一个算法的效率应抛开具体机器的软、硬件环境,而书写程序的语言、编译产生的机器代码质量、机器执行指令的速度都属于软硬件环境。所以抛开计算机软硬件相关的因素,一个程序的运行时间,仅依赖于算法的好坏和问题的规模。

对于一个特定算法只考虑算法本身的效率,而算法自身的执行效率是问题规模的函数。对于同一个问题,选用不同的策略就对应不同的算法,不同的算法对应各自的问题规模函

数,根据这些函数就可以比较算法的优劣。算法的效率包括时间与空间两个方面,分别称为时间复杂度和空间复杂度。

(2) 算法的时间复杂度。一个算法的执行时间大致上等于其所有语句执行时间的总和,对于语句的执行时间是指该条语句的执行次数和执行一次所需时间的乘积。语句执行一次实际所需的具体时间与机器的速度、编译程序质量、输入数据等密切相关,与算法设计的好坏无关,所以,可用算法中语句的执行次数来度量一个算法的效率。

语句在一个算法中重复执行的次数称为语句频度。以下给出了两个 $n \times n$ 阶矩阵相乘算法中的各条语句以及每条语句的语句频度。

语句	语句频度
`for(i = 0; i < n;i++)`	$n+1$
`for(j = 0; j < n;j++)`	n^2+n
`{`	
`  c[i][j] = 0;`	n^2
`  for(k = 0;k < n; k++)`	n^3+n^2
`  c[i][j] = c[i][j] + a[i][k] * b[k][j];`	n^3
`}`	

算法中所有语句的总执行次数为 $Tn=2n^3+3n^2+2n+1$,即语句总的执行次数是问题的规模 n 的函数 $f(n)[Tn=f(n)]$。进一步地简化,可用 Tn 表达式中 n 的最高次幂来度量算法执行时间的数量级,即算法的时间复杂度,记作:

$$T(n)=O[f(n)]$$

上式是 $T(n)=f(n)$中忽略其系数的 n 的最高幂次项,它表示随问题规模 n 的增大算法的执行时间的增长率和 $f(n)$的增长率相同,称作算法的渐进时间复杂度,简称时间复杂度。如上式算法的时间复杂度 $T(n)=O(n^3)$。

算法中所有语句的总执行次数 $T(n)$是问题规模 n 的函数,即 $T(n)=f(n)$,其中 n 的最高次幂项与算法中称作原操作的语句频度对应,原操作是算法中实现基本运算的操作。在上面的算法中,原操作是 c[i][j]=c[i][j]+a[i][k] * b[k][j]。一般情况下,原操作由最深层循环内的语句实现。

$T(n)$随 n 的增大而增大,增长得越慢,其算法的时间复杂度越低。下列三个程序段中分别给出了原操作 count++的三个不同数量级的时间复杂度。

```
① count++;
```

其时间复杂度为 $O(1)$,称为常量阶时间复杂度。

```
② for(i = 1; i <= n; i++)
    count++;
```

其时间复杂度为 $O(n)$,是线性阶时间复杂度。

```
③ for(i = 1; i <= n; i++)
    for(j = 1;j <= n; j++)
        count++;
```

其时间复杂度为 $O(n^2)$，平方阶时间复杂度。

此外，算法能呈现的时间复杂度还有对数阶 $O(\log_2 n)$、指数阶 $O(2^n)$等。

(3) 算法的空间复杂度。采用空间复杂度作为算法所需存储空间的量度，记作：

$$S(n)=O[f(n)]$$

其中 n 为问题的规模。

程序执行时，除了需存储本身所用的指令、常数、变量和输入数据以外，还需要一些对数据进行操作的辅助存储空间。

其中对于输入数据所占的具体存储量只取决于问题本身，与算法无关，这样只需要分析该算法在实现时所需要的辅助空间单元数就可以了。

算法的执行时间和存储空间的耗费是一对矛盾体，即算法执行的高效通常是以增加存储空间为代价的，反之亦然。不过，就一般情况而言，常常以算法执行时间作为算法优劣的主要衡量指标。

习题

1. 填空题

(1) 数据逻辑结构包括_____、_____和_____三种类型。

(2) 线性结构中的元素之间存在_____的关系，树结构的元素之间存在_____的关系，图结构的元素之间存在_____的关系。

(3) 算法的设计要求包括正确性、可读性、健壮性和_____，可读性的含义是_____，健壮性是指_____。

(4) 算法的时间复杂度与空间复杂度相比，通常以_____作为主要度量指标。

2. 选择题

(1) 数据结构中，从逻辑上可以把数据结构分成(　　)。

A. 动态结构和静态结构　　B. 紧凑结构和非紧凑结构

C. 线性结构和非线性结构　　D. 内部结构和外部结构

(2) 计算机算法指的是(　　)。

A. 计算机方法　　B. 排序方法

C. 解决问题的有限步骤　　D. 调度方法

(3) 算法分析的目的是(　　)。

A. 找出数据结构的合理性　　B. 分析算法的效率以求改进

C. 研究算法中的输入和输出的关系　　D. 分析算法的易懂性和文档性

(4) 数据结构这门学科的研究内容最准确的选项是(　　)。

A. 研究数据对象和数据之间的关系

B. 研究数据对象和数据的操作

C. 研究数据对象

D. 研究数据对象、数据之间的关系和操作

3. 简答题

(1) 什么是数据结构?有关数据结构的讨论涉及哪三个方面?

(2) 数据的逻辑结构分为线性结构和非线性结构两大类。线性结构包括数组、链表、栈、队列等,非线性结构包括树、图等,这两类结构各自的特点是什么?

(3) 什么是算法?算法的 5 个特性是什么?试根据这些特性解释算法与程序的区别。

(4) 算法可读性的含义是什么?

(5) 算法的健壮性是指什么?

(6) 给出下列程序中原操作语句的语句频度及程序段的时间复杂度。

程序 1:

```
i = 1;k = 0;
while(i < = n - 1)
{
    k = k + 2 * i;
    i++;
}
```

程序 2:

```
i = 1;k = 0;
do
{
    k = k + 2 * i;
    i++;
}
while(i!= n)
```

程序 3:

```
x = 91;n = 100;
while(n > 0)
    if(x > 100)
    {
        x = x - 10;
        n = n - 1;
    }
    else
        x++;
```

程序 4:

```
x = n;y = 0;
while(x > = (y + 1) * (y + 1))
    y++;
```

模块 1　线性表——排队叫号器

技能目标

- 理解线性表的定义，掌握线性表的特征和基本运算。
- 会使用顺序表的存储结构解决问题。
- 能实现顺序表的各种基本运算。
- 能够使用链表的存储结构解决问题。
- 能实现链表的各种基本运算。

思维导图

本模块思维导图请扫描右侧二维码。

线性表思维导图

在日常生活中，很多场合都需要排队，排队使公共场所有秩序，使各项服务、工作能有序、高效地运行。这种一人占据一个位置，有前驱、有后继的结构，就是线性表结构。

1.1　项目描述

排队叫号是银行营业厅日常工作的重要组成部分，是管理营业厅秩序及提高客户体验的重要手段。某银行有普通客户和 VIP 客户两种客户类型，需要开发一个排队软件，实施银行排队叫号的具体业务，使用排队叫号系统可以使银行大厅业务更有序。图 1-1 所示为银行大厅示意图。

图 1-1　银行大厅示意图

1. 功能描述

排队叫号器功能如图 1-2 所示。

图 1-2 排队叫号器功能示意图

(1) 普通客户取号:普通客户取号后,按号顺序排队,等待呼叫办理。

(2) VIP 客户取号:取号后,该号将优先排到队伍前端。

(3) 叫号:从取号队伍中获取最前面的号码,展示到显示区。

(4) 办理:将正在办理业务的号码显示在显示屏。

(5) 过号:当叫号后,无人上前办理业务时,可进行过号操作,过号显示在过号区。

2. 设计思路

本项目的实质是完成排队号的建立、提取、办理等功能。首先定义项目的数据结构,然后将每个功能写成一个方法模块对业务逻辑数据进行操作,最后完成展示界面以验证各个模块功能并得出运行结果。

1.2 相关知识

本项目的数据是一组客户的排队号信息,每个客户都会取一个排队号,这些排队号具有相同特性,属于同一数据对象,相邻数据元素之间是有序的。由此可以看出,这些数据具有线性表中数据元素的性质,所以该项目的数据采用线性表结构来存储。由于设计了队长的限制,所以采用顺序表来存储数据。

1.2.1 线性表的定义

线性表顾名思义,是具有像线一样性质的表。一根线把元素串联在一起,有线头,有线尾。线性表是最基本、最简单,也是最常用的一种数据结构。线性表中数据元素之间的关系