第3章 计算机软件

[导语]

软件，相对于硬件，是不可见的，无法确定其大小、形状和体积。软件表现为程序和文档的集合，其中渗透了软件开发人员大量的脑力劳动，是其思维的数字化的表达结果。

从计算机系统内部观察，软件犹如精神领袖，硬件则是执行者。软件指挥硬件完成各项任务，硬件向软件反馈执行状态及执行结果。从计算机系统外部观察，软件工作于用户和硬件之间，协调双方的信息交流，并满足双方的工作需求。

本章先给出软件的基本含义和软件的分类信息，然后解释操作系统的基本功能和工作过程，最后介绍软件工程的相关内容。

[教学建议]

教学要点	建议课时	呼应的思政元素
软件和硬件的区别	1	硬件犹如人的身体，软件犹如人的思想，人的成长过程中需要在自己大脑中不断安装各种“软件”，以培养自身多样的素质能力
操作系统的功能和工作周期	3	操作系统CPU管理体现了效率与公平原则，自然科学知识与人文科学知识有美妙的联系，青年人学习时应文理兼收，让自己秀外慧中
软件工程的含义、软件生命周期及软件过程模型	1	计算机安装正版软件，如同人不断学习获得正确的价值观，塑造自身真善美的精神指向

3.1 软件的含义


视频讲解

计算机软件，是计算机可执行的指令的集合及其说明文件。换言之，软件由程序及说明文档组成，程序是计算机可理解的指令序列，说明文档是给用户阅读的操作说明。

从物理层面理解，软件是用户与硬件的接口。用户不直接操控硬件设备，而是通过使用软件向硬件发布命令，完成与硬件的数据交换或指挥硬件进行某种操作。

硬件是“实”，软件是“虚”。计算机硬件是物理上看得见摸得着的实物，而软件是无形的，是一种知识产品，一种智力成果。

若将计算机硬件视为人的躯体,那么软件则是人的思想。人的思想可以指挥人的躯体做出动作,完成任务等。没有思想指导,人的躯体将无法做出合理的行动。这恰如计算机软件可以准确指挥硬件完成指定的操作,若无软件作为沟通媒介传达用户的指令,计算机硬件是无法独自完成输入/输出以及数据分析处理任务的。

3.2 软件的分类

(1) 根据功能的不同,计算机软件可分为系统软件和应用软件两大类。

系统软件(system software)是计算机的管理者,是用户与应用软件、用户与计算机硬件之间的沟通桥梁。系统软件保证计算机按照用户的指令正常运行,满足用户及应用软件的各种需求,并完成管理计算机、维护资源、执行用户命令、控制和调度等任务。

应用软件(application software)是面向某一应用环境,完成用户在具体应用领域的各种具体任务的程序集合。

这两类软件虽然各自的用途及功能不同,但其本质都是存储在计算机中、以某种格式编码书写的程序集合。

图 3.1 描绘了应用软件、系统软件和硬件之间的关系。相对而言,应用软件与用户的关系更为紧密,系统软件与硬件的关系更为紧密。系统软件协调各种外部设备流畅配合地工作,为用户以及应用程序提供各种资源服务。

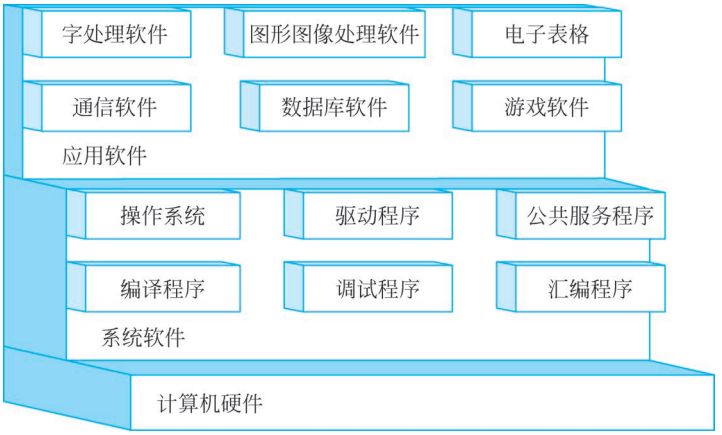

图 3.1 计算机软件

所以在一定程度上,系统软件可被视为应用软件和硬件之间的接口。系统软件为应用软件提供服务,当应用软件需要硬件完成某些动作时,并不是直接向硬件发出信号,而是告知系统,再由系统软件调度硬件,并指挥硬件完成相应任务。

(2) 根据运行载体的不同,软件可分为桌面软件与移动软件。

桌面软件运行在台式机或笔记本电脑上,其功能一般较复杂,支持多种输入与输出方式。

移动软件也称为移动应用软件,即平日里经常提到的 App(来源于 application 这一英文词语)。App 运行于智能手机或平板电脑等手持设备上,相对于桌面软件,其功能较简单。而且由于工作于移动设备,为方便操作,App 一般以触摸方式或实施某个动作进行输

入，以文字、图像、声音等形式输出。

近年来，随着手持移动设备性能的提高以及功能的丰富，桌面软件和移动软件之间的差距也在逐渐减小。

(3) 根据运行地点的不同，软件可分为本地软件和云软件。

本地软件是安装在本地计算机上，由本地计算机进行调用及运算处理的软件。

云软件也称为云应用，是利用云技术实现某地的计算机调用互联网上的云端软件。云软件的使用过程是互联网上的计算资源被调度、被租用的过程。云软件在云端运行，用户通过某一平台与云软件进行数据交换。

上述分类中，系统软件和应用软件的意义最为重要。因此下文从二者的具体含义、详细功能及特点等方面逐一进行介绍。

3.3　系统软件

系统软件是控制和协调计算机及其外部设备，同时支持应用软件的开发和运行的一类计算机软件。从另一角度理解，系统软件是无须用户干预的各种程序的集合。系统软件的功能包括调度、控制和协调计算机及外部设备，支持应用软件开发和运行，监控和维护计算机系统，管理并协调计算机系统中各独立硬件的工作。系统软件为计算机用户和其他软件提供多种服务，使用户和其他软件只需发布任务指令便可指挥硬件工作，而不需要考虑这些任务涉及的底层硬件的工作细节及参数设定。

一般地，系统软件包括操作系统、驱动程序和实用工具程序等。

3.3.1　操作系统

操作系统(Operating System，OS)是计算机最重要的系统软件，它直接运行于计算机裸机上，管理计算机系统的所有资源，是计算机系统所有运行活动的总控制、总管理、总调度，任何其他的软件都需要操作系统的支持才能顺利运行。

通俗地说，操作系统是计算机系统的一个超级管家，该系统里所有的资源都由操作系统高效地组织和管理。应该认识到，计算机系统资源除了包括所有的硬件、软件，还包括抽象的信息资源，如存储空间的地址空间。正是由于操作系统的强大管理能力，才可以正确、合理且高效地组织、协调及管理这些资源，使得用户只需专注于自己的工作，而不必关心如何与硬件对话以及如何完成外围设备操作等，从而使计算机成为一个为用户服务的智能化机器。

1. 操作系统发展

计算机诞生之时，并没有操作系统。当时的计算机采用纯手工操作的方式工作。那时候，人们先用穿孔的纸带表示等待运行的程序和数据，然后把纸带装入输入设备，从而将程序和数据输入内存。接着，操作控制台开启程序，程序运行，完成对数据的处理，并输出结果。最后用户取走结果，卸下纸带，完成一次程序的运行任务。

这种工作方式的弊端很明显：计算机的处理器被单个用户独占，机器工作效率很低。渐渐地，人们摒弃了这种低端的工作方式，开始考虑设计并开发一种能够帮助自己操控计算机各种设备的程序。而操作系统，正是随着计算机技术及其应用的发展，在人们使用计算机

的过程中被逐步设计、开发并完善而成的。操作系统一出现,就成为人们使用计算机不可或缺的帮手,成为提高硬件及软件的资源利用率的一种管理程序,更对计算机硬件和软件的发展起到了巨大的推动作用。

操作系统经历了若干代的发展,分别叙述如下。

1) 批处理系统

批处理系统是操作系统的雏形,它是加载在计算机上的一个系统软件,可以控制计算机自动且成批地处理一个或多个用户的作业,包括程序、数据和命令。批处理系统实际是将多步操作集成为阶段性操作,一个阶段完成若干操作步骤。

批处理系统是计算机从手动操作向自动化操作迈进的第一步。

2) 多道程序系统

多道程序设计技术允许多个程序同时进入内存并运行,即同时把多个程序放入内存,并允许它们轮流在 CPU 上运行。这些程序可以共享系统中的各种硬件和软件资源,且当其中一个程序因输入/输出请求而暂停运行时,另一程序可以占用 CPU,得以运行。

值得注意的是,此时的多道程序系统中,多个进入内存的程序在宏观上是并行的工作状态,即同时进入系统且都处于运行过程中;但在微观上,这些相互独立的程序仍是串行的工作方式,即各道程序轮流使用 CPU,彼此交替运行。

多道程序系统采用并行工作方式管理程序的运行,在一定程度上提高了 CPU 的工作效率,也提高了计算机系统中其他软硬件资源的利用率。

多道程序系统的出现,标志着操作系统趋向成熟,并促使操作系统的具体功能中开始出现作业调度管理、处理器管理、存储器管理、外部设备管理、文件系统管理等功能。

3) 分时系统

分时系统把 CPU 的运行时间分成很短的时间片,按时间片轮流把 CPU 分配给各作业使用。若某个作业在分配给它的时间片内不能执行完毕,则该作业暂时中断,把 CPU 让给另一作业使用,等待下一次轮到该程序的时间片到来时再继续运行。由于计算机速度很快,各作业运行的时间片实际是非常短暂的时间段,所以给每个作业的感觉好像是自己独占了一台计算机。在分时系统中,每个作业可以向系统发出各种操作控制命令,进行人机交互,完成作业的运行。

分时系统可以及时响应作业,提高了系统资源的利用率,避免各作业对 CPU 的漫长等待。更重要的是,分时系统的思想对当今操作系统的 CPU 调度策略有深刻的启发。

4) 个人操作系统

个人计算机产生后,配置在个人计算机上的操作系统应运而生。个人操作系统旨在为用户提供友好的计算机使用环境,较高的交互响应速度,以及丰富的应用软件和娱乐程序等。

微软磁盘操作系统(MicroSoft-Disk Operating System,MS-DOS)是早期个人操作系统的重要代表,它使用命令行界面来接收用户指令,因此用户必须系统地学习 DOS 各种命令的格式及功能意义,才能正确使用这些命令完成相应操作。随着时代的推进和个人计算机的普及,个人操作系统已在各类操作系统中拥有最高的市场占有率。

5) 网络操作系统

时代的发展总能催生新的技术。随着计算机网络技术的发展,网络操作系统的出现成

为必然。这类操作系统通常建立在主机的操作系统之上，按照网络体系结构的各个协议标准，在非网络操作系统基础上增加了网络管理模块，提供网络通信、资源共享及管理、系统安全和各种网络应用服务等功能。

6）分布式操作系统

在网络操作系统出现并发展的同时，还出现了一种类似于网络操作系统的分布式操作系统。这种操作系统通过通信网络，将地理上分散的、具有自治功能的计算机系统（或数据处理系统）互连起来实现信息交换和资源共享，协作完成任务。分布式操作系统主要功能包括资源管理、分布式进程同步和通信、任务分配等。

在分布式系统中，一个作业可以被分解为若干部分，由多个不同地址的计算机分别完成。它强调分布式计算和处理、多机合作的鲁棒性、快速的响应时间、高吞吐量和高可靠性。

分布式操作系统和网络操作系统有明显的区别，其中最主要的两点不同在于：其一，分布式系统中的各台计算机地位平等，无主从关系，而网络操作系统使用环境中各台计算机地位不平等，有一台主机，其他是从机；其二，分布式系统中的资源为所有计算机系统无条件共享，而网络操作系统中，资源是有限制条件的共享。

7）嵌入式操作系统

嵌入式操作系统（Embedded Operating System，EOS），指嵌入式系统的操作系统。

嵌入式系统指在各种设备或系统中完成特定功能的软件及硬件，它们自身构成了一个大的系统中的一部分，这部分作为一个整体被嵌入大的设备或系统中，因此称为嵌入式系统。

在嵌入式系统中使用的操作系统，具有通用操作系统的基本特点，能够有效管理复杂的系统资源，并且把硬件虚拟化。

2. 操作系统的地位

操作系统在计算机系统中的地位极其重要，这一点毋庸置疑。

从用户角度分析，操作系统是其他应用软件与硬件之间的接口，它承上启下，协调、管理和控制计算机的各种资源。

从操作系统与应用软件及硬件之间的信息交互分析，操作系统隐藏了复杂的硬件调用接口，为应用软件提供了更便捷、更简单、更清晰的系统调用接口。用户可通过这些接口操控硬件，而无须考虑硬件控制的细节，只需专心开发自己的应用程序即可。

从资源管理角度分析，操作系统是计算机系统资源的管理者。此时的资源包括计算机的硬件，如 CPU、存储器、输入/输出设备等，也包括各种软件、数据以及文件。操作系统管理并调度计算机系统的各种资源，并在多个用户对同一资源的申请产生冲突时，选择最优方案进行资源分配，保证各种资源都有较高的工作效率。

3. 操作系统的功能

操作系统的主要功能包括 CPU 管理、存储管理、设备管理和文件管理等。

1）CPU 管理

CPU 管理主要指进程管理，包括创建和撤销进程，协调各个进程的运行状态，并完成进程间的信息交换。

进程是程序的一次运行，注意区别程序和进程。**程序**是静态的，是为完成某一任务而编写的一组指令语句。进程是动态的，是运行中的程序。

一个程序是不变的,但它的各次运行都会对应不同的状态,产生不同的进程。当一个程序被选中,被允许占用 CPU 得以运行时,相应的进程就被创建出来了。

因为进程是一个运行中的程序,所以它具有生命周期。在一个生命周期内,一个进程可以处于以下 5 种状态之一。

- 创建:进程正被创建。
- 运行:进程被分配了 CPU,其指令正在执行。
- 等待:进程正等待输入/输出动作的完成或某个信号的接收等。
- 准备就绪:进程正等待被分配 CPU。
- 终止:进程已经执行完毕。

图 3.2 描绘了操作系统进行 CPU 调度时进程的各种状态之间的转换,图中矩形框内表示了进程的某种状态,两个矩形框之间箭头上的文字表示了两种状态之间的转换条件。可见,一个进程在其生命周期内只有一次“创建”状态和一次“终止”状态,而“准备就绪”“运行”“等待”这 3 种状态可以出现多次。

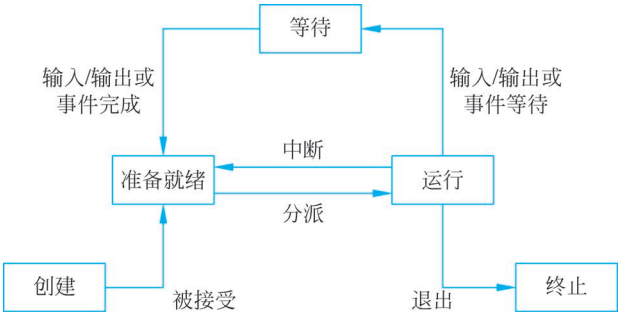

图 3.2 进程之间的状态转换

当一个程序被调用,操作系统判断其是否可以运行,如果可以则创建一个新的进程并将其放入就绪队列;此后,操作系统根据 CPU 调度算法让其在合适的时机占用 CPU;若执行完毕,则进程终止;在执行过程中,进程可能会因调度算法而被迫放弃 CPU 而重新进入就绪队列,或者因为等待输入/输出请求而进入等待队列,待输入/输出请求完成后再次进入就绪队列,如此反复,直至进程终止。

在进程调度中,调度算法非常重要。所有的程序指令都必须由 CPU 执行,因此,一个 CPU 要为多个进程服务。那么,将 CPU 最大效率地分配给每个进程,便是进程调度算法的任务。

常用的进程调度算法如下。

(1) 先来先服务算法(First Come First Serve,FCFS)。

顾名思义,FCFS 算法将准备就绪的进程组织为一个队列,每次从就绪队列中选择最先进入队列的那个进程,为其分配 CPU。该进程一直运行到终止(该进程将转入“终止”状态)或有输入/输出事件发生时才放弃 CPU(该进程将转入“等待”状态)。该算法思想简单,一定程度上可以实现基本的公平。

例如,当前 CPU 就绪队列中有 3 个进程 Process1、Process2、Process3,如图 3.3 所示,三者需要占用 CPU 的时间依次为 67、120、89(此处忽略时间单位)。

此时,若采用 FCFS 策略,Process1、Process2、Process3 将按其在队列中的次序逐一占

用 CPU。图 3.4 给出了三个进程依次执行时的简易时序图。若设 CPU 开始工作的时间为 0，那么图 3.4 中矩形框上方的各个数字表示了各进程开始执行的时间及结束时间。

图 3.3　CPU 的就绪队列

图 3.4　FCFS 策略下三个进程的执行时序图

这里引入**进程周转时间**的概念来衡量 CPU 调度策略的性能。进程的周转时间指该进程从进入就绪状态到其最终执行完成所需要的时间。显然，一个 CPU 系统中所有进程的**平均周转时间**越短，CPU 的工作效率越高，系统所采用的 CPU 调度算法性能越好。

对于上面就绪队列中的三个进程，在采用 FCFS 调度算法时，它们的平均周转时间为 (67＋187＋276)/3＝176.6。

(2) 短进程优先算法(Shortest Process Next，SPN)。

SPN 算法优先选择最短进程，即估计运行时间最短的进程，占用 CPU，使其立即执行直至终止或因输入/输出事件而放弃 CPU。SPN 算法的弊端明显：①它需要预估进程的运行时间，这种估计未必准确；②它对长进程不利；③它未考虑进程的紧急程度。

仍以上面的三个进程为例，设此时它们占用 CPU 的次序依次为 Process1、Process3、Process2，那么具体执行过程的简易时序图如图 3.5 所示。

图 3.5　SPN 调度策略下三个进程的执行时序图

当采用 SPN 调度策略时，三个进程的平均周转时间为(67＋156＋276)/3＝166.3。对比 FCFS 和 SPN，在这个例子中，SPN 的调度效率更高。

(3) 时间片轮转法(Round-Robin based on time piece)。

时间片轮转法把所有的就绪进程按先来先服务原则排成队列，每次将 CPU 分配给队首进程，并令其执行一个时间片。时间片长度可以是几毫秒或几百毫秒。当时间片用完该进程暂停执行，并前往就绪队列的末尾；然后再将 CPU 分配给就绪队列新的队首进程，令其在下一个时间片使用 CPU，如此反复，直至进程获得了全部的执行时间并完成了执行过程。

该算法在一定程度上可保证公平，即所有进程均可获得执行机会，并在给定的时间内被响应。但对于输入/输出任务密集的进程，它们在自己的时间片可能多次遇到输入/输出请求，而被迫放弃 CPU，这浪费了时间片的剩余时间，降低了当前进程在时间片内的执行效率。

时间片轮转法的关键是为每个进程分配等长的时间片，并在时间片结束时强迫其转入就绪状态。仍以上面的三个进程为例，假设此时 CPU 的时间片为 50，那么它们依次执行过程的简易时序图如图 3.6 所示，其中的阴影区域只为强调三个程序执行完毕的时间点。

若采用时间片轮转策略进行 CPU 调度，Process1 首先得到时间片，当第一个时间片使

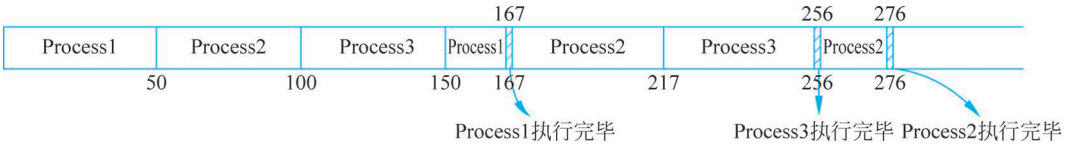

图 3.6 时间片轮转法的简易时序图

用完毕,Process1 被迫转为就绪态,Process2 得以使用 CPU。当 Process2 的时间片耗尽,Process3 开始使用它的时间片。Process3 的时间片耗尽之时,三个进程都得到了执行机会,且得到了相同的占用 CPU 的时间。

之后,下一轮的轮转执行过程开始,仍然是 Process1 首先得到时间片。注意,此时 Process1 还需要占用 CPU 的时间长度为 17,所以这一次该进程并不完全耗尽 50 的时间片,而是只占用其实际需要的时间。因此在图 3.6 的简易时序图中,第二轮的 Process1 的实际耗费时间是 17。至此,Process1 执行完毕,它的周转时间是 150+17=167。

在第二轮的轮转执行过程中,当 Process2 使用时间片结束,Process3 在其时间片内也执行完毕,但它实际占用 CPU 的时间是 39。所以 Process3 的周转时间是 217+39=256。

在第三轮的轮转执行过程中,只有 Process2 是就绪进程,它直接获得时间片,但其实际执行时间是 20,因此其周转时间为 256+20=276。

至此,所有的进程执行完毕,三个进程的平均周转时间为(167+256+276)/3=233。

在上面的例子中,时间片轮转法的调度效率比 FCFS 和 SPN 都低,但这并不能说明该策略的调度能力不佳。实际上,每种调度算法都有其适用的场合,都会在某一类情形下呈现较好的调度性能,可以根据不同的情况选择不同的调度算法以获得系统最高工作效率。

(4) 优先级调度算法。

优先级调度算法为每个进程赋予一个优先级(必须保证用户进程的优先级不得高于内核进程的优先级),每次选择优先级最高的进程占用 CPU。该算法可以根据不同的标准确定不同的优先级设定细节,以保证在特定应用需求下,最紧急的进程最先得以执行。

(5) 多级反馈队列调度算法。

多级反馈队列调度算法设置多个就绪队列,并为各个队列赋予不同的优先级。第一个队列的优先级最高,此后的优先级依次递减。同时,算法令各个队列中进程的时间片长度与优先级情况反向设置,即优先权越高的队列,其中进程的时间片越小。

创建了一个新进程后,它先进入第一队列的末尾,按 FCFS 原则排队等待调度。当轮到该进程执行时,如果它能在该时间片内完成,便离开 CPU;如果它在一个时间片内未完成,那么算法将其转入第二队列的末尾,再按 FCFS 原则等待调度。若它在第二队列中运行一个时间片后仍未完成,再将它放入第三队列,如此循环下去。

注意,仅当第一队列空闲时,调度程序才调度第二队列中的进程运行;仅当前面的 $n-1$ 个队列均为空时,才会调度第 n 个队列中的进程运行。如果 CPU 正在执行第 n 个队列中的进程,此时有新进程进入优先权较高的队列(第 1~$(n-1)$个队列中的任何一个),则此时新进程将抢占 CPU,同时调度程序把正在运行的进程放回到第 n 个队列的末尾。

注意,除了完成进程调度的任务,操作系统进行 CPU 管理的工作内容还包括保证进程间安全流畅的通信,并解决进程的互斥及同步问题等。

【思政 3-1】　操作系统 CPU 管理的深入思考

操作系统的功能包括 CPU 管理、存储管理、设备管理等。表面上看，这些管理内容和细节只是技术上的各种规范和设定，与日常生活毫无关联，但实际上，操作系统各种管理功能的思想中蕴含了丰富的哲学意义，仔细思考，可以启迪我们在工作和学习中以正确的观点对待事物，分析并处理问题。

例如，CPU 管理过程中体现了追求公平和效率的思想。操作系统首先保证每个程序都有机会使用 CPU，且任何程序不能因等待输入/输出任务而妨碍其他程序使用 CPU。这些情况和人类社会类似。我们坚持公平，保障所有人的基本权利。但当有特殊紧急情况时，一些程序可以被赋予更高的优先级，以得到优先使用 CPU 的权利。这种优先级的设定，很多时候是以提高系统整体的效率为目标的，如最短作业优先策略的实施，便是基于此初衷。

操作系统的 CPU 管理与追求公平和效率的思想相呼应，这是自然科学领域的技术知识和人文科学领域的知识之间的美妙关联。虽然自然科学研究的是自然界物质形态、结构、性质和运动规律，人文科学研究的是人类的信仰、情感、道德和美感等，但二者都是为了提升人的生活质量。

确实，自然科学的技术方法的飞速发展推动了人类社会进步，极大地改善了人们生活的物质条件。同时，在人的生活学习过程中，需要人文科学知识素养的加持来保证人的精神层面能够拥有正确的价值指导，以对事物做出正确的判断，塑造高尚的价值观，建立积极阳光的处世态度和生活态度。这一切是如此重要，以至于如果没有这些指导，自然科学的发展将去往何处？

那么，一个人的学习除了自身的专业，也应去吸收其他多个领域的知识。亲爱的读者，你已经在自然科学的计算机技术领域有了一定的学习积累，倘若平日读些人文社科书籍，让自己文理兼修，成长为一个能文善武、秀外慧中的人，那么真可以拥有较高的人生价值！

2）存储管理


视频讲解

在存储管理功能中，操作系统要完成内存分配、存储保护、地址映射、内存扩充等任务。

内存分配，指在程序调入内存时，为其分配内存空间，在运行完毕程序撤销时回收内存空间。

存储保护，即确保每一道程序在专属内存区域内正确存储，各道程序存储区域互不干扰，且操作系统的存储区域不受干扰。

地址映射，即操作系统要将程序地址空间中的逻辑地址转换为真正存储区域的物理地址。逻辑地址，是各条指令语句在程序中使用的地址；物理地址是指令语句被调入内存后在真正存储空间中的地址。逻辑地址并不对应真正的物理存储单元，因此直接通过逻辑地址访问数据是无法找到真正的目标数据的。操作系统的存储管理功能提供地址映射功能，在访问前将逻辑地址转换为物理地址，实现数据的正确访问。

内存扩充，即在逻辑意义上来扩充内存容量，以帮助程序更顺利的执行。

（1）内存分配。

一个程序经过编译之后，操作系统将为其分配存储空间，完成程序的装入。分配存储空间时操作系统有三种分配方式：绝对分配、可重定位分配、动态运行时分配。

绝对分配仅在运行单道程序时可以使用，也称静态分配方式。该方式下，操作系统将内

存分配给程序,并使得程序中的逻辑地址与实际内存地址完全相同。

可重定位分配是根据内存的当前情况,将程序分配到内存中适当的区域。以这种方式为程序分配内存后,程序所占据的内存区域将对应一个物理地址的区间。程序中的各条指令的物理地址与其在程序内部的逻辑地址完全不同。在将程序装入内存时,程序中各指令和数据的地址将会被修改,这是在存储分配时一次性完成的地址变换,此后随着程序的执行这些地址也不再改变,故此方式也称为静态重定位。

动态运行时分配在把程序装入内存后,并不立即把程序中的逻辑地址转换为物理地址,而是把地址转换工作推迟到程序真正被执行时再进行。

(2) 存储管理。

操作系统管理内存空间的常见方式有分区管理、分页管理,分段管理以及段页式管理。

分区管理是内存空间管理最简单、最直观的方式,即为整个程序在内存中分配一个区域。由于程序必须整体连续存放,故该方式会在分区之间产生难以被利用的小空间,即存储碎片。

在分区管理方式下,各分区通常按大小进行排队,同时操作系统建立一张分区使用表,每一表项记录各分区的起始地址、大小及是否已分配的状态。当有用户程序要装入时,操作系统的内存分配程序将检索该表,找出一个能满足存储要求且尚未分配的分区,将其分配给该程序,然后将该分区对应的表项中的状态置为“已分配”。

分页管理将内存分成固定大小的部分,称为页,同时将程序装入连续或不连续的若干页。因为被分配的页面可以不连续,所以该方式不会产生存储碎片,进而提高了内存利用率。但在一页之内,可能出现存储内容不能占满整个页面的情况,所以可能产生存储空间的浪费。

分段管理类似于分区管理,将程序整个装入一个存储区域;同时将程序分为若干段,如数据段和代码段,加以不同的保护。该方式也具有和分区管理类似的缺点,易产生碎片。

段页式管理将程序分段,但是各段不再被分配至连续的存储空间,而是采用离散分配方式存放各段。可见,段页式管理综合了分页管理和分区管理的优点。

(3) 地址映射。

地址映射的目标是**逻辑地址**,或称其为相对地址,是高级语言源程序中为各指令及变量赋予的地址。逻辑地址的提出可以使程序员专注于算法设计和程序指令的编写,而不必操心指令及变量在内存中的存储单元的分配。**物理地址**是指令及变量所存放的物理存储单元的地址,也被称为绝对地址。当访问指令及变量时,需要将相对于程序而言的逻辑地址转换为相对于真正存储单元的物理地址,这称为地址映射,该过程由硬件——内存管理单元完成。

考虑一种最简单的情形。若将内存视为一块整体的存储空间,不采用分页或分段式管理,那么一个程序执行前,操作系统将寻找恰当的内存空间将其整体都装入内存。设定程序中各条指令的逻辑地址是一个相对于程序起始位置的数值,即各条指令语句相对于程序头部的偏移量。这可理解为,程序中各指令语句的逻辑地址是从 0 开始的一串渐增的整数值。此时,逻辑地址到物理地址的转换,只需要将逻辑地址与程序在物理存储空间中存放的真正起始地址相加即可。

图 3.7 给出了上述从逻辑地址到物理地址的转换。设某程序在内存中存放的起始地址

为 A，程序中某一条指令的逻辑地址为 L，那么该指令在内存中真正的物理地址是 A＋L。当这条指令被执行时，CPU 将根据 A＋L 的地址信息在内存中进行寻址，取出该指令。

在分页式管理的内存系统中，从逻辑地址到物理地址的转换需要考虑每页的起始地址。此时，操作系统将为各程序维护一个页映射表(Page Map Table，PMT)，用于记录各个页与相应的物理存储块之间的地址对应关系。

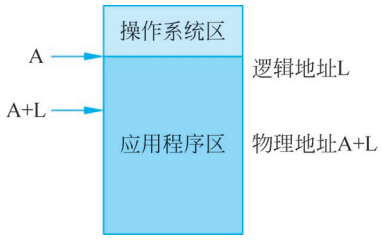

图 3.7　物理地址和逻辑地址的关系

以名为 Program1 的程序为例，表 3.1 给出了该程序的页映射表。从表中可知 Program1 被分成了 3 个页面，这 3 个页面所对应的物理存储块号也一一在表中列出了。表 3.2 则给出了内存中各物理存储块对应的存储内容。

表 3.1　程序 Program1 的页映射表

页　　号	物理存储块号
0	10
1	6
2	9

表 3.2　内存中程序 Program1 的存放信息

物理存储块号	内　　容
0	
1	
2	
3	
4	
5	
6	Program1-页面 1
7	
8	
9	Program1-页面 2
10	Program1-页面 0
11	
12	

此时的逻辑地址由两个值组成——页编号和页内偏移量，表示为：<页编号，偏移量>。

根据逻辑地址是相对于程序起始位置的相对偏移量的理解，页编号和偏移量的计算方式分别为：

页编号＝逻辑地址 / 页面大小

偏移量＝逻辑地址 MOD 页面大小(其中 MOD 是求余运算)

此时，要把一条指令语句的逻辑地址转换为物理地址，首先需查找页映射表，找到该指令所在页对应的物理存储块号，然后用物理存储块号乘以块的大小，得到该页头部的物理地址，这个物理地址再加上偏移量，便得到了指令语句的物理地址。

当然，不同的操作系统在进行内存管理时会采用不同的方式，这决定了进行逻辑地址和

物理地址之间相互转换将采用不同的方式。

(4) 内存扩充。

当内存不够时,操作系统可以自动“扩充”内存,为程序提供一个容量比实际内存大的存储空间。现代操作系统一般使用虚拟内存技术实现内存扩充。

虚拟内存技术,是从逻辑上扩充内存容量的方法。它利用程序运行时的局部性原理进行内存容量的扩充。**程序运行的局部性原理**,是指程序在运行时,某一时刻不会用到全部的程序指令。因此仅把程序的一部分装入内存,程序就可运行。

作为一种存储管理技术,虚拟内存使得应用程序认为它拥有连续可用的内存(一个连续完整的地址空间),且这个内存空间比真正的物理内存空间要大。而实际上,应用程序通常是被分割成多个物理内存碎片,部分已经装入内存,而其他部分暂时存储在外存上,在需要时才进行数据交换。

虚拟内存技术由硬件和操作系统通过存储信息调度和离散的管理方式实现。下面以页式内存管理方式为例,简单介绍虚拟内存技术的实现细节,段式及段页式内存管理方式下虚拟内存的实现过程可触类旁通。

当操作系统使用页式内存管理方式,将设置请求调页功能和页面置换功能,以离散的分配方式,多次将程序的不同部分调入内存运行。此时的页映射表要进行扩充,以记录页面被调入的状态,如表 3.3 所示。

表 3.3 扩充的页映射表

页号	物理存储块号	状态位 (是否调入内存)	访问字段 (被访问的次数)	修改位 (是否被修改)	外存地址

当要访问的页面不在内存上时,发出缺页中断请求,操作系统将所缺页调入内存。若所缺页面是从未运行过的页,操作系统从硬盘文件区将页面调入;若所缺页面是曾经运行过又被换出的页,操作系统从对换区调入。

若是页面调度算法不恰当,可能会出现页面一直在被调入调出的情形,这种现象称为抖动。可以用式(3-1)计算缺页率衡量页面调度算法的优劣:

$$缺页率 = 缺页次数 / 内存访问次数 \tag{3-1}$$

实际的系统中,缺页率与程序覆盖的存储页数、置换算法及页面大小有关。

常见的页面调度算法有如下几种:

- 最佳页面置换(OPTimal replacement,OPT)算法。OPT 算法是理想化的算法,选择那些永不使用或最长时间不再被访问的页面置换出去。
- 先进先出(First In First Out,FIFO)算法。FIFO 选择最先进入内存的页面,将其调出。该算法实现简单,但有时会有缺页率升高的情形。
- 最近最久未使用(Least Recently Used,LRU)置换算法。LRU 选择最近最久未使用的页面予以淘汰,优先考虑调入反复使用的页面。该算法需要记录页面使用时间的先后关系,硬件开销大。
- 最少使用(Least Frequently Used,LFU)置换算法。该算法选择使用次数最少的页面予以淘汰。

除了虚拟内存技术，**交换技术**也可以提高内存使用率。

交换技术也称作对换技术，包括换出和换入两个过程。换出，是将处于等待状态或在CPU调度原则下被剥夺运行权力的程序从内存移到辅存以腾出内存空间；换入，是把准备好竞争处理机的程序从辅存移到内存。理想情况下，内存管理器的交换过程速度足够快，可以保证总有进程在内存中得以执行。

交换技术，需要注意以下几方面的问题。首先，交换过程需要备份存储，因此必须保证磁盘足够大。其次，为有效使用CPU，需设定每个进程的执行时间比交换时间长，以避免出现CPU的等待状态。最后，若换出进程，则必须确保该进程完全处于空闲状态。

交换技术与虚拟内存技术的思路有两点明显的不同。

① 目的不同。交换技术旨在提高内存利用率，若从程序的视角观察，并没有感到明显的内存扩充。而虚拟内存技术的目的却是力图使程序感到自己的所有代码已经完全进入内存，且所处的这个内存区域是一个巨大的存储空间。

② 调度单位不同。交换技术以程序为单位，在程序之间选择哪一个进入内存，哪一个调出内存。而虚拟内存技术则是以一个程序的不同页面为单位，选择哪个页面进入内存，哪一个页面调出内存。

实际上，操作系统存储管理功能的细节是随着用户应用需求的增加而不断加以完善的。从存储单元的地址分配，到存储空间的分页及分块式管理方法，以及虚拟内存的设计，这些都是由问题和需求驱动而产生的解决方案。这一过程恰对应了科研活动中发现问题，分析问题，然后解决问题的思维路径。虽然现实生活中，人们往往是被动地遇到诸多问题，但当面对难题时，如果可以保持冷静的头脑执行上述思维路径中的步骤，去深刻分析问题的条件和缘由，从而发现问题的本质，那么将更容易找到正确的解决办法。

3）设备管理


视频讲解

操作系统的设备管理功能是为有输入/输出请求的进程分配相应设备，并完成相关的操作。具体而言，操作系统将完成缓冲区管理、设备分配、设备处理、虚拟设备及实现设备独立性等工作。由于很多设备是输入/输出设备，这些设备不仅种类繁多，而且其特性和操作方式差异较大，因此设备管理是操作系统中复杂且与硬件关联紧密的一项工作。

设备管理的目标首先是向用户提供使用方便且独立于设备的界面，即让用户无须了解各种具体设备的物理特性，只需按照统一的规则使用设备即可。同时，设备管理会对各种设备的使用情况进行协调，保证合理正确地为各个程序分配所需的设备。另外，设备管理还以提高各种设备的使用效率为目标，通过通道技术和缓冲技术来提高CPU与外设以及各种外设之间的工作并行程度。

（1）设备分配。

一个计算机系统中会有多种设备，同类设备也可能不止一台。当希望使用某种设备的进程数大于该种设备的数量时，将会引起进程对设备的竞争。此时操作系统会对这些设备进行合理分配。常用的设备分配技术为：独占——固定地将设备分给一个进程；共享——运行设备为若干进程共用；虚拟——用共享设备模拟独占设备。

（2）通道技术。

通道实际是一台小型外围处理机，旨在建立独立的输入/输出操作。通道使数据的输入/输出操作、输入/输出操作的组织、管理和终止均独立于CPU，从而保证CPU不必在输

入/输出设备管理上花费时间。因为在执行输入/输出任务过程中,需要完成诸如询问输入/输出设备状态、等待输入/输出设备、处理输入/输出中断等操作,这些工作如果由通道完成,那么可显著提高 CPU 的工作效率。

通道建立后,CPU 仅需要向通道发出输入/输出指令,告知需要访问的通道程序和具体设备,通道接到指令后,便从主存指定位置取出通道程序,执行程序来完成输入/输出操作。

根据信息交换方式的不同,通道分为三种类型:①字节多路通道,按字节方式交叉工作即每次子通道控制输入/输出设备交换 1 字节后,便将控制权交给另一个子通道,让其交换 1 字节,多用于连接大量低速或中速的输入/输出设备;②选择通道,控制输入/输出设备一次交换一批信息,多用于高速或中速输入/输出设备;③成组多路通道,结合前两者优点,可以连接多台高速输入/输出设备,为其提供成批信息交换方式,也可提供信息交替传送方式。

通道和 CPU 之间的通信是双向的,CPU 可向通道发出输入/输出指令(如启动输入/输出设备、查询输入/输出设备、查询通道、停止输入/输出设备等),完成任务后,通道也以中断方式向 CPU 发出信息。

(3) 缓冲技术。

缓冲技术是在计算机系统的某些设备上设置能存储信息的缓冲区,以解决进行信息传输的两个部件之间的信息传输速度不匹配问题。缓冲技术带来了设备并行度的增加,但这种并行度的增加主要依赖于进程内部存在的并发性和进程之间的并发性。常见的三种缓冲方式是单缓冲、双缓冲、循环缓冲。

4) 文件管理

(1) 文件的意义。

文件是相关数据的完整集合,这个集合用文件名作为标识。

从用户角度解释,文件是组织数据的基本单位,是信息存取的基本单位,用户可以创建、修改、删除、打印或检索某个文件。从硬件底层的角度解释,文件是记录在存储介质上数据的集合,是若干字节的有序序列。

与文件紧密相关的一个概念是文件夹。**文件夹**是某些文件的集合,为方便组织和管理文件而设定。文件夹也是一种文件,可成为另一文件夹中的成员。换言之,一个文件夹既可以包含不同类型的文件,也可以包含其他文件夹。

用文件及文件夹组织计算机系统的数据,使得系统中的信息呈现出一个树状目录结构下的数据视图。操作系统中的文件管理功能实现了这种树状目录结构的数据管理方式,同时也为每个文件分配外存空间,并建立目录项以记录文件的各种特征信息,对文件读写过程进行管理和保护。

(2) 文件管理的功能。

文件管理的功能包括:建立、修改、删除文件;按文件名访问文件;决定文件信息的存放位置、存放形式及存取权限;管理文件间的联系并提供支持文件的共享、保护和保密等。其中,文件的共享是指允许一个文件可以被多个用户共同使用,这可以减少用户的重复性劳动,节省文件的存储空间,减少输入/输出文件的次数等;文件的保护主要是为防止由于错误操作而对文件造成的破坏;文件的保密是为了防止未经授权的用户对文件进行访问。

文件的保护、保密实际上是用户对文件的存取权限控制问题。一般为文件的存取设置两级控制:第 1 级是访问者的识别,即规定哪些人可以访问;第 2 级是存取权限的识别,即

有权参与访问者可对文件执行何种操作。

(3) 文件存储空间管理。

由于文件存储时通常是被分成许多大小相同的物理块，并以块为单位交换信息，因此，操作系统要进行有效的文件存储空间的管理，实质上是对存储单元区域中的空闲块进行组织和管理，包括空闲块的组织、分配与回收等问题。

索引法、链接法和位示图法是三种不同的空闲块管理方法。

索引法把空闲块作为文件并采用索引技术管理这些表示了空闲块的文件。链接法使用链表把空闲块组织在一起，当申请者需要空闲块时，分配程序从链首开始摘取所需的空闲块；当放弃存储区域时，管理程序把回收的空闲块逐个挂入队尾。位示图法是在外存上建立一张位示图(bitmap)，它可用一个二维数组表示，用于记录文件存储器的使用情况。每一位仅对应文件存储器上的一个物理块，取值 0 和 1 分别表示空闲和占用。

(4) 树状目录。

目录，可理解为文件的有名字的分组。操作系统的**树状目录结构**是非常重要的文件逻辑组织方式。在树状目录结构中，树的根节点为根目录，数据文件作为树叶，其他所有目录均作为树的节点。

根目录隐含于一个硬盘的一个分区中，根目录在最顶层。它包含的子目录是一级子目录。每个一级子目录又可以包含若干二级子目录，以此类推，这样的组织结构被称为目录树。

操作系统的树状目录从最高层的文件对象开始逐渐分解开去，层层展开，将各层的文件或文件夹一一铺排开来，这种文件组织方式与人类的知识归纳整理的思路较为一致，非常方便用户的理解和使用。在树状目录结构中，从根目录到任何数据文件之间，只有一条唯一的通路。若从树根开始，把到达某一文件的通路上经过的全部目录名与文件名收集并连接起来构成序列，那么这个序列被称为路径。

图 3.8 给出了某计算机系统 Windows 操作系统下 D 盘的目录的例子。图 3.8 中，在硬盘 D 盘上存放了两个文件夹(userfile 和 uservedio)以及一个文件(useraudio)。userfile 文件夹包含两个文件(file2020 和 file2021)和一个文件夹(file2022)，file2022 文件夹是上层 userfile 文件夹的子文件夹，其中包含了三个文件(catfile、duckfile 和 Danielfile)。

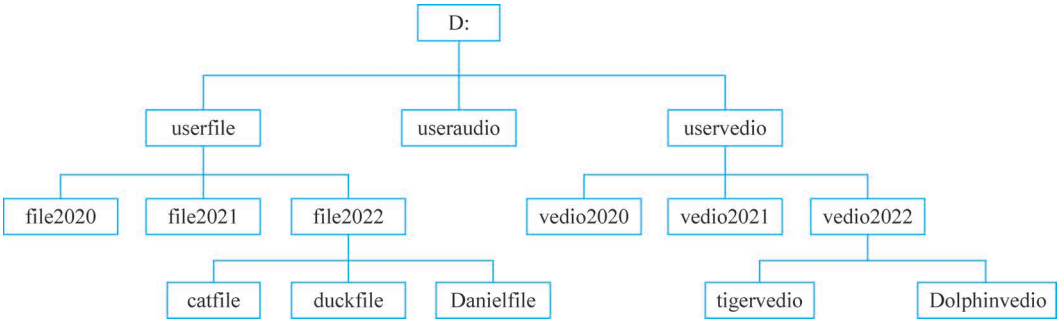

图 3.8　Windows 系统下的树状目录示意

“d:\userfiles\file2022\Danielfile”描述了上述目录结构中的一条路径，这条路径表示在 D 盘的 userfiles 目录中的 file2022 子目录下的 Danielfile 对象的位置。

从根节点开始到达某一文件对象的通路上的信息构成了该文件的**绝对路径**。显然,每个数据文件的绝对路径是唯一的。但如果文件系统包含了多层的目录,使用绝对路径表示文件的位置稍显烦琐,所以引入相对路径的概念。**相对路径**是由从当前目录到目标文件的通路上经过的所有目录名和文件名连接而成。

父目录是指当前路径的上一层目录。每个目录下都有代表当前目录的“.”文件和代表当前目录父目录的“..”文件,相对路径名一般就是从“..”开始的。

树状目录结构以清晰的结构组织众多文件,路径概念的提出允许在不同的目录下可以存在同名文件——甚至是同名且同类型文件,因为有不同的路径信息对这两个文件进行区分。这给用户带来文件命名时的灵活性,以及温馨的使用体验。

5) 提供用户接口

操作系统提供的用户接口有两种类型:程序接口和命令接口。

(1) 程序接口。

程序接口也被称为系统调用接口,是操作系统提供的系统与用户程序之间的接口。此类接口把应用程序的请求传给系统,调用相应的内核函数完成所需的操作,将处理结果返回给应用程序。

(2) 命令接口。

命令接口也被称为操作接口,它以命令方式执行接口操作,命令所带的参数指明了操作的具体细节。

命令接口可分为联机用户接口和脱机用户接口。前者是交互式命令接口,其命令方式包括命令行方式、批处理方式、图形界面方式。其中批处理命令接口,包括作业控制语句或命令、作业控制说明书。

其中,图形用户界面是对计算机发展和普及有重要意义的用户接口。它将计算机操作方式从命令输入方式转换为图形化操作方式,对用户非常友好。在图形用户界面出现之前,用户必须掌握全面的专业知识才能正确使用命令操控计算机,这使得计算机成为普通用户在技术上难以高攀的工具。而图形用户界面出现之后,菜单、图标、窗口、对话框、按钮等图形化操作元素的设计,使用户可以直接选择相应的图标来运行程序,这大大降低了计算机的使用难度,加快了计算机走入寻常家庭的速度,使计算机成为更多非专业人士在工作学习和生活中不可或缺的工具。

6) 一些特殊的操作系统

(1) 嵌入式操作系统(embedding operating system)。

嵌入式操作系统是以应用为中心、以计算机技术为基础的操作系统,适用于对功能、可靠性、成本、体积、功耗等有严格要求的专用计算机系统。

嵌入式操作内核是嵌入式操作系统的核心基础和必备部分,其他部分要根据嵌入式操作系统的需要来确定。嵌入式操作系统的最大特点就是可定制性,即能够提供对内核的配置或裁剪功能,可以根据应用需要有选择地提供或删减某些功能,以减少系统开销。

(2) 移动操作系统(mobile operating system)。

移动操作系统实际上是嵌入式操作系统的一种。包括为移动设备开发的操作系统、从桌面操作系统派生而来的操作系统,以及从嵌入式操作系统 Linux 派生而来的操作系统。

安卓和 iOS 是市面上主流的移动操作系统。安卓是基于 Linux 的嵌入式移动操作系统，自 2010 年以来已成为智能手机的主流操作系统之一。iOS 是从苹果公司的桌面操作系统 Mac OSX 派生而来，安装在一系列苹果公司的移动设备上，如 iPhone、iPad、iPod touch 等。

(3) 分布式操作系统(distributed operating system)。

分布式系统是配置在分布式系统上的共用操作系统，是由多个处理机通过通信线路互连而构成的松散耦合系统。分布式操作系统的分布性、自治性、并行性、全局性的特征，使其具有资源可共享、计算速度可加速、可靠性高、通信方便快捷的优点。

但是分布式操作系统也存在若干不足，如可用软件不足，系统软件、编程语言、应用程序以及开发工具相对很少。同时，分布式操作系统还存在通信网络饱和、信息丢失以及网络安全问题，毕竟，方便的数据共享也意味着机密数据容易被窃取。

3.3.2　驱动程序

驱动程序(device driver)是使计算机和设备可以通信的特殊程序。通俗地说，驱动程序是帮助管理系统并指挥硬件设备完成指定任务的一类程序，所以其全称为“设备驱动程序”。驱动程序相当于硬件的接口，操作系统通过这个接口来控制硬件设备的工作。

在安装了 Windows 9X 及以前版本的操作系统的计算机中，需要专门为一些硬件设备(如显卡、声卡、扫描仪、摄像头、Modem 等)安装驱动程序，这些硬件才能正常工作。但随着操作系统功能的完善，操作系统本身就已经支持多种硬件设备的运行，因此无须再额外安装驱动程序。

3.3.3　实用工具程序

实用工具程序(utilities)是计算机系统中执行特定功能的程序，包括磁盘管理程序和硬件故障检测程序等，也可以执行一些操作系统无法完成的任务。

3.4　应用软件

应用软件是指为特定领域开发，并为特定目的服务的一类软件。应用软件是直接面向用户需要的，它们可以直接帮助用户提高工作质量和效率，甚至可以帮助用户解决某些难题。

应用软件一般分为两类：一类是为特定需要开发的专用软件，其使用范围限定在某些特定的单位和行业，如会计核算软件、工程预算软件和教育辅助软件等；另一类是为了方便用户使用计算机而提供的某种具体应用服务的通用工具软件，也称基本应用软件，如用于文字处理的 Microsoft Word、用于辅助设计的 AutoCAD，用于图像处理的 Photoshop，用于系统维护的各种杀毒软件及防火墙软件等。

对于各种应用软件，安装其正版软件具有重要而深刻的意义。安装并使用正版软件是保护知识产权、保持经济高速发展的需要，也与国家和企业信息安全、企业的诚信和规范管理有密切关系。


视频讲解

3.5 软件工程

3.5.1 软件工程的含义

软件工程的出现是为了克服软件危机。计算机诞生初期,软件主要是由个人设计并实现,也被个人使用,因此制作的软件规模较小,而且没有与之匹配的操作说明文档。

随着软件应用领域的扩大,软件功能上需求的增加,软件的制作发展到必须由若干编程人员共同协调完成。此后,时代的飞速发展使得软件的规模日益增大,软件复杂度持续提高,软件可靠性要求也不断提高,这些都使得之前的软件制作流程和技术完全不能满足软件发展的需要,甚至出现大量劣质软件并带来生产和经济上的损失。这些在软件开发和维护过程中出现的种种严重问题便构成了软件危机。

软件危机造成了软件开发成本和进度难以预测,用户需求与软件功能之间的不匹配,软件运行时的可靠性较低,更因缺少系统的说明文档而难以维护,等等。所有这一切要求用规范化的、可定量的过程化方法去管理并控制软件开发及维护过程。

软件工程,是阐述用规范化、可定量的过程化方法来管理并控制软件开发及维护过程的学科,其研究内容覆盖了软件开发和软件管理两方面。前者包括软件开发方法及技术、软件开发工具及环境;后者包括软件管理技术、软件规范。

若是从数据处理的角度分析,软件工程是设计可处理多样数据的完备程序的方法学。广义地说,软件并不仅仅指包含指令的程序,还包括文档甚至程序处理的数据。就这一观点而言,软件可被视为一种抽象的、具有逻辑结构和正确意义的实体。

3.5.2 软件工程的基本理解

软件工程的提出是为了在给定成本的前提下,开发出高质量的软件。此时的“高质量”指软件应具有适应性、有效性、可靠性、可重用性、可移植性、可理解性、可修改等特点。

基于此,软件工程必须给出软件开发的方法、工具和过程。软件开发方法描述设计软件的形式化过程。软件开发工具提出软件设计及实现所需的设备、环境及系统方面的要求。软件开发过程则给出将方法与工具有效结合,真正实施软件开发的具体步骤。

如同人的一生要经历出生、婴幼儿、儿童、少年、青年、中年、老年、去世这些阶段一样,软件一生也会经历被设计、被开发、被使用、被维护、被废弃这几个阶段。软件的一生通常称为软件的生命周期。

显然,就像一个人小时候需要接受正确且良好的教育才能成长为有用的人才一样,软件设计人员在设计开发软件之前,也需要非常认真而全面地了解所设计的软件需要解决的问题,进行可行性分析,才能正确全面定义软件的具体功能,进而尽可能地保证所设计并实现的软件能够满足用户的应用需求,并易于后续维护。毕竟,无论是人还是软件,在年幼时所犯的错误,大多为小错误,此时进行教育并改正所付出的代价是最小的。

既如此,软件工程自然地使用生命周期方法学,在软件开发的时间轴上将开发过程分解为若干阶段,明确规定各阶段的任务,并要求软件开发按照这样的规划细节具体执行。这其实也体现了中国传统的分而治之的思想,即将开发及维护软件这个复杂任务分解为若干阶

段的子任务，前一阶段完成的结果是后一阶段的前提和基础。而且，每一阶段的成果都需要经过严格评审，并提交正确的文档后，才能进入下一阶段。这种科学而系统的管控方式可以有效保证软件的质量和功能达到预定要求，提高软件开发的效率。

3.5.3　软件生命周期

软件的生命周期是软件自产生到报废的整个过程，也称为软件生存周期。软件生命周期内需要完成多个任务，这些任务按层次推进，同时包含对工作细节的审查，以提高软件质量。一般而言，软件生命周期分成三个大的阶段，软件定义、软件开发和软件维护，在每一阶段又有若干子阶段。图 3.9 给出了软件生命周期的三个阶段及各阶段的子阶段，其中的虚线框说明了子阶段需要完成的任务。下面对各个子阶段的工作内容稍作解释。

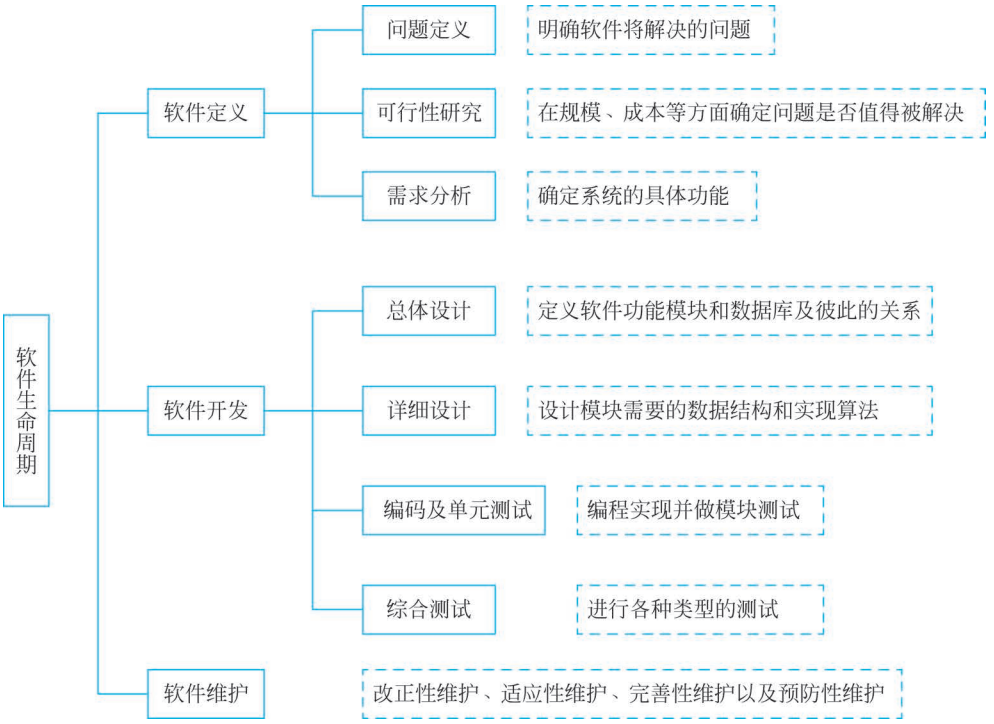

图 3.9　软件生命周期

1. 问题定义

这一阶段中，软件开发方和需求方共同确定软件开发的目的，制订项目总体开发计划，明确软件的初步需求，并初步确认软件开发的可行性。

2. 可行性研究

此阶段需要在技术、经济及社会等方面确定软件是否能够、是否值得开发。技术可行性分析要确定软件的功能和条件，并分析在现有的硬件及软件资源下，开发人员的技术能力和工作基础是否能够满足开发要求。经济可行性分析要估算开发过程的成本，进行软件经济效益评估等。社会可行性分析需在责任、合同、侵权等方面确认软件过程的管理制度、人员素质及相关资源是否符合要求。

3. 需求分析

此阶段需要明确客户对软件项目的需求，通过建立逻辑模型和数据流图来确定用户对软件项目的功能、性能、数据格式、界面等方面的要求，并最终用软件需求规格说明书的形式详细阐述用户需求。

需要注意，初期确定的需求在软件开发过程中会不断变化或细化，此时需要制订需求变更计划来应对这种变化，以保证整个项目的正常进行。

4. 总体设计

总体设计阶段明确软件开发的大致框架，需要给出实现目标系统的几种可能方案，搭建架构，划分软件的各功能模块，确定各模块之间的接口连接及数据传递等。

5. 详细设计

详细设计也称模块设计，即详细地设计每个模块的功能细节，确定各模块所需的算法、数据结构及数据库设计细节。此阶段将给出详细的规格说明，程序员可以根据规格说明写出实际的程序代码。

6. 编码及单元测试

这一阶段是将详细设计的结果转换为计算机可以运行的代码。在编码中要制订统一、符合标准的编写规范，以保证代码的可读性和易维护性。

7. 综合测试

软件设计完成之后，需要经过严密的测试，以发现整个软件设计过程中存在的问题并加以改正。测试的方法主要有白盒测试和黑盒测试。软件测试之前需要制订详细的测试计划文档，并且严格按照测试计划文档进行测试，以减少测试的随意性。测试的过程主要有单元测试、集成测试、系统测试、验收测试。

8. 运行维护

软件维护是软件生命周期中持续时间最长的阶段，其工作量可达软件开发阶段工作量的数倍。在软件开发完成并投入使用之后，由于多方面的原因，软件无法继续满足用户的需求，此时若需延续软件的寿命，就必须对软件进行维护。软件维护包括改正性维护、适应性维护、完善性维护和预防性维护。

3.5.4 软件过程模型

软件过程模型也称为**软件开发模型或软件生命周期模型**，是为了规范和约束软件开发过程中各项任务和活动而设定的工作模型。软件过程模型旨在使软件生命周期中的各项任务能够有序地按照规程进行，保证软件生命周期的进展达到预定计划。软件过程模型确定了软件生命周期各阶段的次序，以及各阶段实施时需要遵守的规则；同时也保证了参与软件开发的人员对各种工作任务有统一的评判标准，可进行高效的沟通。软件过程模型的使用有助于软件模块的重用及管理。

常见的软件过程模型有瀑布模型、快速原型模型、螺旋模型、喷泉模型、V 模型，以及增量模型。

1. 瀑布模型

瀑布模型是用传统软件生命周期方法学开发软件时所遵从的模型，如图 3.10 所示。

此模型中，各阶段按时间顺序从早到晚依次开始，每一个阶段的输出是下一个阶段工作

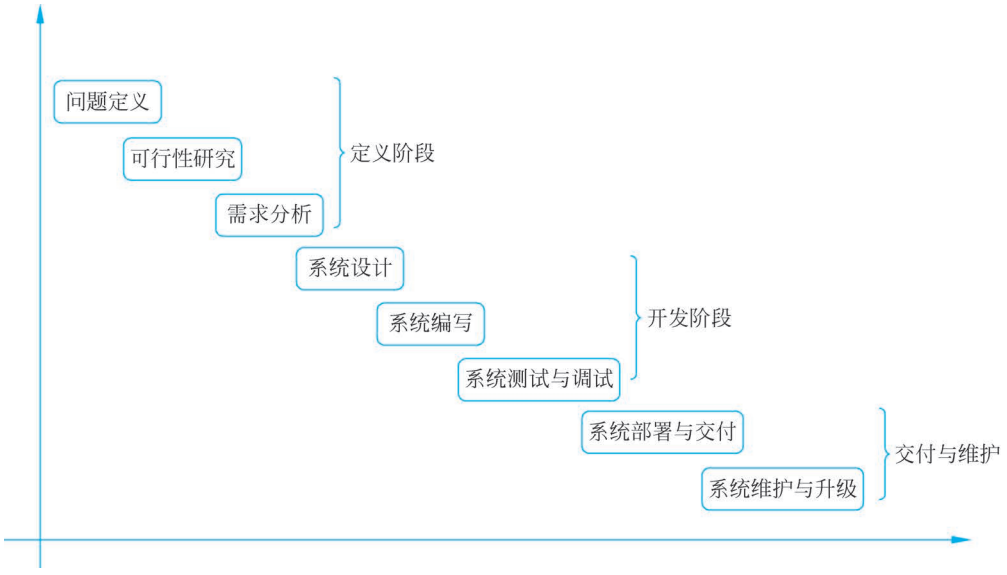

图 3.10　瀑布模型

的输入。瀑布模型的工作方式犹如“疑是银河落九天”的瀑布，表示工作流程从一个阶段自然地流动到下一阶段，这也代表着软件开发工作从抽象设计到具体实现的过程。

2. 快速原型模型

快速原型模型的思想是快速制作一个可以在计算机上运行的、能模拟最终的软件产品的程序，该程序或者能给出目标软件的功能框架，或者能实现目标软件的某一功能子集。换言之，快速原型模型是先建立其目标软件的一个简易版本，确认并理清各种功能需求以及各方面细节设计的要求，然后在该简易版本的基础上，逐渐完成全部软件开发工作。快速原型模型的开发思路如图 3.11 所示。

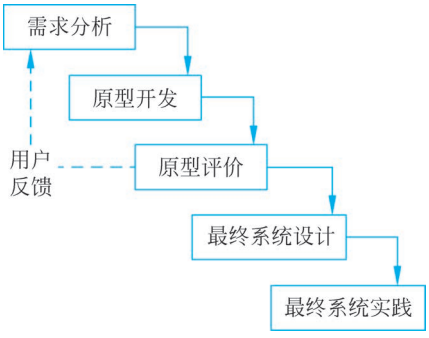

图 3.11　快速原型模型

这个首先建立起来的软件简易版本被称为软件原型，用户可对该原型进行测试评定，给出具体的改进意见以丰富、细化软件需求，此后技术人员基于用户的反馈调整软件设计细节。显然，软件原型是用户和软件开发人员之间的良好沟通媒介。

采用快速原型模型时，对软件原型的使用有两种方式：抛弃方式和附加方式。前者在充分使用原型之后，弃之不用，即软件原型完全作废。后者则将软件原型用于软件开发的全过程，通过对其做完善、改进和优化，进而完成所有的软件开发工作。

3. 螺旋模型

螺旋模型将瀑布模型和快速原型模型结合起来，在软件开发的每个阶段之前都增加了快速原型模型以进行风险分析，如图 3.12 所示。

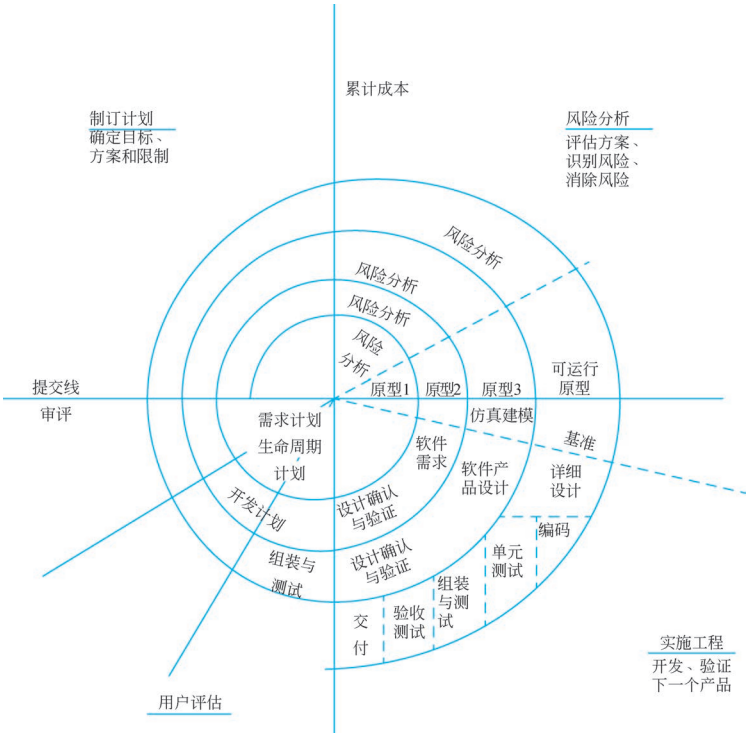

图 3.12 螺旋模型

由图 3.12 可见，螺旋模型的软件开发过程构成了一条螺旋线，从内部的需求分析开始，该螺旋线在四个象限内旋转延伸，表示软件开发工作沿着螺旋线进行若干次迭代。左上、右上、右下以及左下的四个象限依次代表了以下的工作内容。

- 制订计划：确定开发目标，选定实施方案，明确开发活动的具体限制条件。
- 风险分析：分析评估所选开发方案，设计方法来识别和消除风险。
- 实施工程：实施软件开发，并进行验证。
- 用户评估：评价开发工作，提出修正建议，制订下一步计划。

上述四项工作内容依次进行并构成循环，从而尽量降低软件开发过程中的风险。也正是因为具有可降低风险的特点，螺旋模型主要适用于一些大规模软件项目的开发过程。

4. 喷泉模型

喷泉模型是以用户需求为动力，以对象为驱动的软件开发模型，主要用于面向对象的软件开发过程。该模型认为软件开发过程中的各阶段可以相互重叠和多次反复，而且各开发阶段没有特定的次序要求，可以交互进行，也可以在某个开发阶段中随时补充其他任何开发阶段中的遗漏。这种方式允许开发过程的某一阶段可以随时融入其他阶段的工作细节，恰如一个喷泉，喷至高处的水可以向下落入正在上升的水中，而落入水池中的水滴也可以再次被向上喷出到达最高处，如图 3.13 所示。

从图 3.13 中可观察到，软件开发过程的各阶段不必严格按照线性次序依次开始，各阶

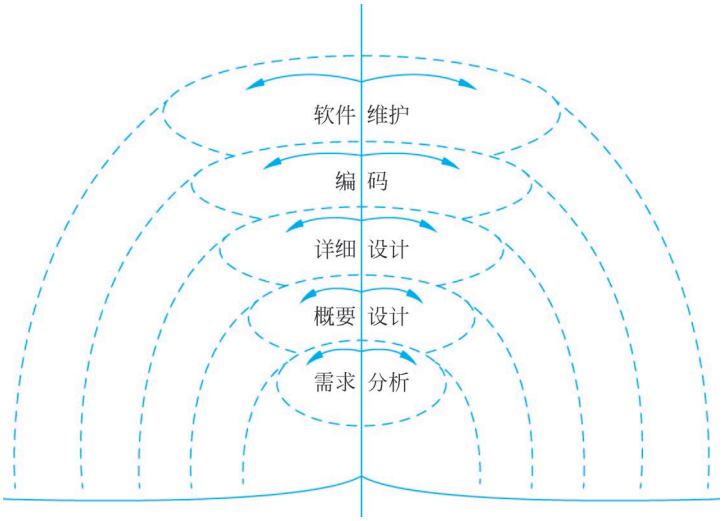

图 3.13　喷泉模型

段之间无明显边界，即各个阶段的工作可以同步进行。这种工作方式需要大量的开发人员，加大了项目管理难度。

5. V 模型

V 模型是瀑布模型的变形，如图 3.14 所示。

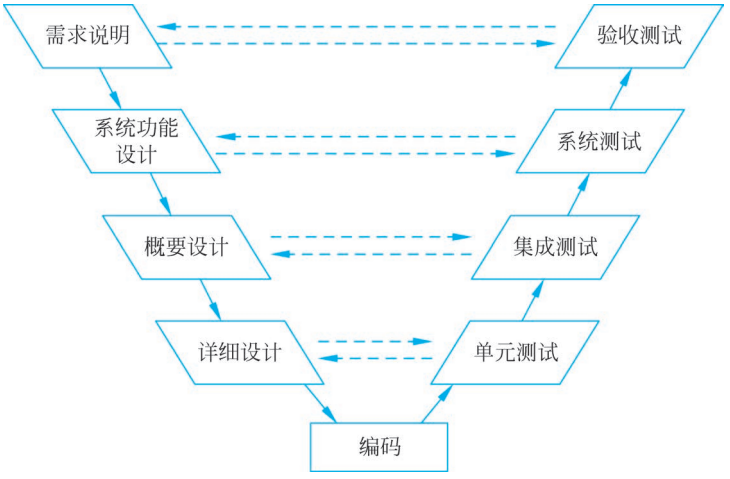

图 3.14　V 模型

与瀑布模型不同的是，V 模型不是以直线的串联方式安排开发过程的各阶段，而是将各阶段的实施安排成典型的 V 形，V 模型的思想是令开发和测试活动相互对应，每个开发阶段都应以对该阶段工作成果的测试活动作为结束。V 模型的测试活动包括了从具体到抽象的不同层次的测试内容：单元测试、集成测试、系统测试以及验收测试。V 模型这种工作一步、测试一步的思路有助于轻松地触发并定位软件问题，进而及时分析并修复软件缺陷。

6. 增量模型

增量模型也被称为连续版本模型，如图 3.15 所示。增量模型的思想是首先构建一个仅

实现一些基本功能的简单工作系统，将其交付给客户，然后在该简单系统的基础上设计并实现若干连续的迭代版本，将其交付给客户，直到完成最终目标的软件系统。

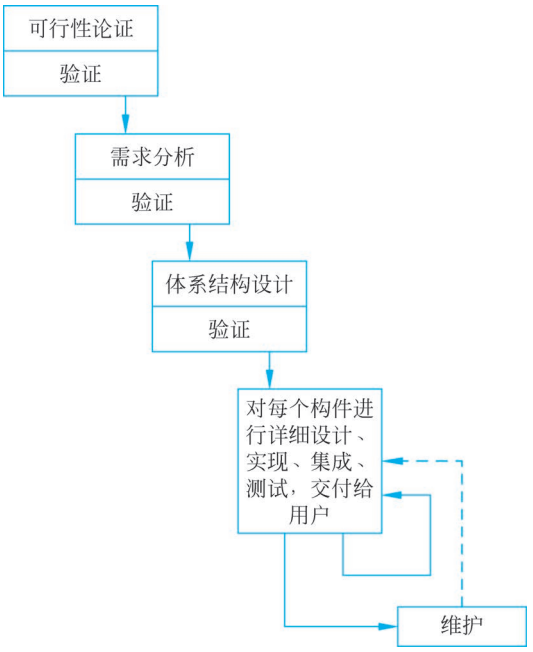

图 3.15　增量模型

具体实施时，增量模型分批地向用户提交产品，即整个软件产品被分解成许多个增量构件，开发人员向用户逐个提交这些构件。

至此，可以更深刻理解软件工程的目的，它给出了明确的软件开发过程的各种规定和细节，能够帮助程序员正确理解问题需求、写出高质量的代码。软件工程持有的观点是：开发软件是一个长期的、持续迭代更新的过程。基于此，软件工程提出的很多要求，如及时添加注释、保持良好的命名习惯、及时进行单元测试、及时总结分析等，都是软件开发过程中有力而高效的工具。

软件工程的提出到底有何意义？世界上有遵守软件工程开发规则的开发人员，也有不遵守软件工程开发规则的开发人员。前者能够创造出规范的、可靠的、有效的、易于维护的软件产品，他们被称为工程师。后者创造出的软件产品可能结构混乱、代码层次模糊、性能不高。显然，工程师们的作品才是人们乐于选择的作品，而工程师们遵守正确的设计规则，沿用良好的工作习惯，都值得各个领域的工作者借鉴，以使自己做事情的过程更加顺利、高效。

【思政 3-2】　关于人的思想软件

掩卷而思，深深感到一个人从小长大的过程，就如同一台刚出厂的计算机不断被加装各种软件，最终形成一台功能丰富、可以在某一方向完成专业任务的高效机器的过程。

人刚出生时，天真可爱，只具有人的基本生物本能；这如同一台计算机刚出厂，那些硬件设备只有电子元件的基本性质一般。渐渐地，人慢慢长大，在成长过程中直接或间接地学习各种生活技能，接收外界的常识信息，建立自己的思维体系；这如同那台计算机被装入操

作系统，具有处理常规问题的基本能力。

然后人进入小学、中学、大学，甚至继续攻读研究生，这一路上，人不断汲取各种知识，丰富自己的头脑，提升自己的能力和素质，获得多样的技能本领，成长为一个在某一领域具有相当工作能力的人才。这一路径，也如同那台计算机被陆续安装了一些软件，有常规的文档处理及文本展示软件，也有专业方向的数据处理软件——当然，这里的数据含义很广，可以是数值数据、图形图像数据、文本数据、音频及视频数据等。这样，此台计算机就可以在相关方向上完成一些专业操作，成为一个有力的工具。

正如计算机硬件应该装入正版软件，执行精准的指令，正确完成预定的功能一般，人需要秉承积极向上的思想，才能实施正确稳妥的行为，完成人的学习工作任务。在这个纷繁复杂的世界中，充斥着各种诱惑，各种思潮，各种生活方式，有些就像盗版软件会给计算机带来病毒而导致崩溃一样，使人无知、糊涂甚至犯错。此时，需要让自己装入杀毒软件，主动汲取正确的思想观念，建立真善美的价值观，以有效改正错误并提升自己抵御诱惑的能力。

真，可以作为学习科研的指导原则，追求真理，证明事实，找到问题的真正答案。这个过程中，以“真”为指导思想，将可以极大地锻炼人的探索能力、钻研能力和克服困难的能力。善，则是为人处世的准则，劝解人从良好的心态出发，乐观向上地对待周围的人和事，做好事，多助人。善的举动，能给人带来内心的丰盈满足，能让自己的身体和心理处于上佳的循环之中，给人带来收获和回报。这也正好符合“善”字的另一个意思——吉祥美好。而“美”，其意义则随着时代的更迭而不断丰富着。美，可以指符合自身实际的目标，为这个目标而持续努力的过程，在前进路途上披荆斩棘的步伐，以及在遇到坎坷时坚定的目光……实际上，你只要珍惜时光，努力奋斗，你的每个瞬间，都是美妙的，都给度过的每一时刻镶嵌了有价值的内容。

人的成长和计算机硬件不断装入软件的过程有如此的类似之处，这大概就是自然科学妙不可言之处吧。

所以，美好的你是否在风华正茂的年月里把时间用在读书学习上？是否坚持锻炼身体，练就了强健体魄？你是否畅想了未来，为自己毕业后的工作进行了规划？你是否勤奋努力，完成学习任务的同时积极参加学校实践活动来丰富自己的经历，提高自己的综合素质呢？还有很多有意义的事情，你是否去做过？这一切，都是在为你的大脑安装正版软件啊，都是在赋予你一项一项的能力啊！

青春时光里，你精神焕发，朝气蓬勃，思路敏锐，目光如炬。此时的你专心于学习，是再好不过的事情！更不用说，当你的父母看到你认真努力，坚持不懈，积极向上，乐观开朗的模样时，他们将是多么欣慰，多么愉快，多么喜上眉梢啊！你必定也因为他们的自豪笑容而由衷开心吧！

亲爱的你，到知识中去吧，和书本对话吧，你一定知道，出色地完成学习任务是多么酷的一件事！