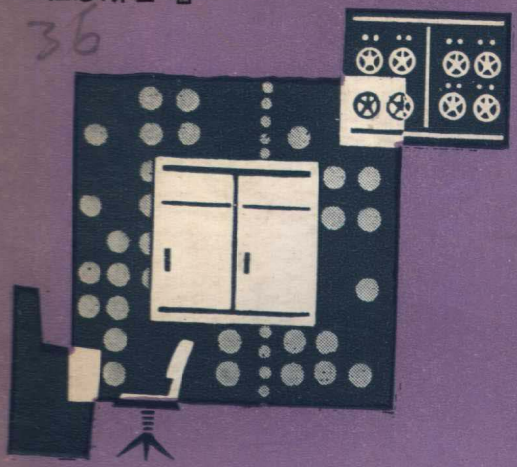
J. 伯恩斯坦 著



电子计算机

过去、现在和未来

科学出版社

电子计算机

——过去、现在和未来——

〔美〕J. 伯恩斯坦 著

丁元煦 译

科学出版社

1978

内 容 简 介

本书通过著者本人学会怎样和电子计算机相对话的亲身经历对电子计算机作了简明的介绍。书中介绍了计算机和计算机语言的发展历史以及在这方面作出了卓越贡献的重要人物。书中谈到了电子计算机的现状和未来。这是一本普及电子计算机知识的通俗读物。

Jeremy Bernstein

THE ANALYTICAL ENGINE

Computers—Past, Present and Future

Random House, New York

电 子 计 算 机

——过去、现在和未来——

〔美〕J. 伯恩斯坦 著

丁元煦 译

*

科学出版社出版

北京朝阳门内大街137号

中国科学院印刷厂印刷

新华书店北京发行所发行　各地新华书店经售

*

1978年12月第 一 版　开本：787×1092　1/32

1978年12月第一次印刷　印张：2

印数：0001—326,700　字数：44,000

统一书号：15031·205

本社书号：1263·15—8

定价：0.16 元

前　言

尼尔·玻尔非常擅长于用幽默的语言，(或者说以“一些小小的笑话”)来表达自己的想法和批评别人的想法。最近无意中发现玻尔晚年的助手爱基·彼得逊 (Aage Peterson) 写的一篇有关玻尔的哲学思想的文章中，谈到玻尔所喜爱的一个故事。彼得逊写道：“一个偏僻的村子里有个小小的犹太人集居地。一次，一位有名的犹太传教士来到附近的城市讲道。村民们很想知道这位大学者到底讲些什么，就派了一个年青人去听听。年青人回村后说：‘教士讲了三次。第一次讲话，非常精彩，讲得简洁扼要，每个字我都能听懂。第二次讲得更好，深奥难解。我听不懂多少，但他自己则懂得所讲的一切。第三次讲得最好，对我是一次难忘的巨大体验。我一点儿也听不懂，连教士自己也懂不了多少。’”

我对计算机这个主题的感受，多少有点象那个年青人对那个教士的感受。一部分，我自己慢慢就懂了，另一部分需要计算机专家给我讲解（读者有兴趣，可找一本列有在这方面曾经对我特别有用的参考文献的书目），最后一部分则涉及人们尚未完全了解的最新研究领域。当事情成为纯理论推理时(我往往也最热衷于学习和描述这类事物)，我就试着在适当章节加上“或者”及“可能”等字眼，以提醒读者。

本书以及它所根据的《纽约人》杂志里的文章，都是在许多人协助下写成的。首先，书中谈到的大部分人，目前仍积极从事该领域的研究。他们阅读了部分原稿。在这里，我感谢他们花了不少时间向我讲解他们的研究工作并精心审阅稿

件，使之尽可能不出差错。除了这些计算机专家外，还要感谢国际商业机器公司的托马斯·狄根（Thomas J. Deegan）。他对于为广大外行严肃认真地撰写有关计算机的文章曾经给以热心的赞助，并用了几乎一年的时间帮助我广泛搜集材料。还要感谢《纽约人》杂志编辑部的罗伯托·格第（Robert Gerdy）。他经常耐心地指导、批评和帮助我搞清了文章中最难理解的材料，而当初，这对我几乎是件不可能的事。最后要感谢《纽约人》杂志的编辑威廉·肖恩（William Shawn），是他鼓励我为这本阅读对象基本上为非科学家的杂志撰写这样一个复杂的科学主题。

一

电子计算机，虽然还处于新兴时期，但在现代生活中所起的作用已经同当年希腊、罗马神谕所起的作用颇为相似。人们普遍相信，如果向通用电子计算机（UNIVAC）提出一个问题，它就会进行一大堆奇妙而熟练的作业，然后以神谕的形式给出答案。把电子计算机看作迅速成长的运算机器，恐怕更为现实而不至令人望而生畏。它能进行加减乘除，能解一长列包括加减乘除运算的复杂式子。它还能在计算进行中改变进程，即按照前一步运算所得结果，在几种步骤中选择其中某一步骤续运算。虽然，所有这些基本上都可归结为算术问题，但是从算术上来说，小学生用一分钟作一个乘法，同计算机用一秒钟作十万个乘法之间的区别只在于数量上的差别。当然，两者之间实际上还是有很大差别的。因为人们交给计算机的问题可以比想要交给一组熟练的计算人员（更不用说小学生了）去运算的问题复杂得多。例如，当人们打算预测一次选举的结果时，他可以向计算机输入有关上一次选举的大量数据，这样，计算机几乎马上就可以将它同这次选举的相应数据进行对比。这是个纯数字过程；只要有足够时间，人也能作出完全一样的对比，并精确地得到相同的预测结果。显然，任何这类预测结果，其可靠程度不可能超过输入计算机的统计数据的可靠性。计算机本身并不创造数据，因此对计算机在不正确的数据基础上得出错误预测时加以嘲笑，同在正确的数据基础上作出正确预测时把计算机说成先知先觉，都是同样不恰当的。在这两种情况下，计算机仅仅是个工具，既值不得称

赞，也值不得责怪。

我是以理论物理学者的身份同计算机打交道的。我经常碰到一些数字问题，用手工进行这些枯燥的计算时，假如我能坚持到底的话，就需要用上好几个星期的时间，而用计算机只要几分钟就解决了。在向计算机提出几次问题后，我终于对它感到无比钦佩。但一直到最近，我才对计算机有了足够的接触和了解，也就是说，我自己开始同它打交道。但是，我们之间曾经存在过语言上的障碍。这种障碍只有通过熟练的媒介才能克服。因为不能简单地用英语向计算机提出问题并要求它给出答案。计算机有它自己的语言——机器语言。我们必须把吩咐它做的事译成这种语言。最初同计算机打交道时，我见过一些这种翻译（即所谓"程序"）。由于它们看起来像是一些难懂的字母与数字的阵列，而且其排列形式同普通英语、算术或代数上所用的毫无共同之处，于是我便认为编程序乃是一种魔术，最好让专职人员去做。在许多年当中，我曾以我熟悉的语言准备了自己的问题并提出解这些题的指令，然后全部交给程序员接着去做。程序员和计算机解除了我大量的苦工，但也给我一种我所不喜欢的感觉，一种远离自己所提问题的漠然之感。

但我的一些同事当时告诉我，程序设计已经出现了一次大革命。由于创造了几种新语言，这种工作已经变得容易多了。这些新语言很接近传统数学语言中的词和符号，计算机在一定程度上能自己进行翻译。而且，他们还告诉我，任何一个人哪怕他对现代计算机工作原理只有一些模糊的概念，也能够学会至少在某种程度上不必通过翻译人员而同它对话。人们还告诉我，适于用来进行科学计算的这种语言叫做 FORTRAN—Formula Translation。不少大学都经常举办有些象意大利语或法语速成课那样的短期速成班。我任教的纽约大学

有一个计算中心，它是古朗数学科学研究所的一部分。该中心每年举办三期 FORTRAN 短期课程，我决定参加其中一期的学习。结果我发现，学一学 FORTRAN，对一般了解计算机是个很好的入门。

上第一堂课之前，我以为所有同学一定都是职业科学家或工程师，但我们的教师霍华德·瓦洛维茨（Howard L. Walowitz）经过短时间调查后发现，班上除了有一个图书馆馆员以外，还有几个该校护理教育系正在写有关如何详细分析医学统计数据的论文的女同学。虽然有少数同学曾在别处学过 FORTRAN 课程，到这里来是重新学习，但大部分同学对这种语言是一无所知的。然而，用 FORTRAN 语言写程序是很容易学会的，不到几天工夫，我们大多数人都在一定程度上掌握了用 FORTRAN 语言写程序的要领并能用计算机处理问题并取得答案。

程序设计的第一步是对一些事物写下一些什么东西。瓦洛维茨首先让我们看一种写程序用的带格子的纸，即一张空白的写 FORTRAN 程序的纸。这是一张淡绿色的纸，有打字纸那么大小，划成约 1500 个长方格，每个格内可以填入一位数、一个字母或一个标点符号。然后，他给我们一个容易的"示范"问题和一张程序纸。这个问题显然用不着利用一台价值数百万美元的计算机。这是一个简单的所得税计算。其中，所得税按以下的公式计算：

TAX = 22%(GROSS INCOME − $600.00 × NUMBER OF EXEMPTIONS)
[所得税] [总收入] [免税数]

乍一看，FORTRAN 程序似乎有点可怕，但一旦懂得各种短语所起的作用，我们就会懂得它们是何等精确，甚至是何等雅致！将计算所得税的 FORTRAN 程序稍加简化，就成为：

```
    1   READ INPUT TAPE 5,
        读   输入   带
            GROSS,EXMP
            总   免税
    TAX = 0.22*(GROSS - EXMP*600.0)
所得税          总      免税
        IF (TAX) 2,3,3
        假定 所得税
    2   TAX = 0.0
        所得税
    3   WRITE OUTPUT TAPE 6,
        写    输出    带
        GROSS, EXMP, TAX
        总     免税  所得税
        GO TO 1
        进  入
```

看到象第一个那样的指令时，很容易得出下述印象，即计算机将对任何英语指令作出反应。实际上，FORTRAN 只包括有限的几个短语，把它们组织在一起就能表达解各种复杂数学问题所需的一系列步骤。大多数大型计算机在磁带上"存贮"数据。在本例中，由"读出"指令通知计算机在 5 号磁带上找到合适的数据——一组总收入及免税数的数字。

其次，用接近通常的书写方式写出该式，但有所不同的是需要改变一些符号以免含混不清。于是写出：

```
TAX = 0.22*(GROSS - EXMP*600.0)
[所得税]      [总]     [免税]
```

接着写出功能最强的一个 FORTRAN 语句(这个语句表明了现代计算机作出判断的能力)：

```
IF   (TAX) 2,3,3
[假定] [所得税]
```

该语句告诉计算机在算出所得税后，当答案分别为负数、零、或正数时应该如何继续进行工作。如果答案为负数，计算机就转到指令 2，或

```
TAX = 0.0
(所得税)
```

由于不可能有负税这样的事，于是指令 2 指出，如果"免税数

＊600”大于“总数”，计算机就把所得税写成零。在这种情况下，负数答案与无税的意思实际上相同，因此程序员对这个负数的值是不感兴趣的。

如果得出的所得税为零或正数，计算机就进到指令 3，后者命令在另一磁带上写出所得税计算的实际结果。这个磁带将被用来启动一个高速打印机，这样，打印机就会在纸上印出整个算式。

“转到 1 ”这个指令是让计算机返回到程序的开头，就是说，用 5 号磁带上的下一组数字重新开始。

程序员一旦为变数“总”及“免税”提出许多组数值后，该指令序列便能命令计算机计算所得税了。上述数值写在紧接在指令序列下面的“数据（DATA）”标题下。这些数据在适当时候就被记在 5 号磁带上。程序与数据是相互分开的实体。程序员一旦编出程序后，任何时候他提供了数据，就能再使用这个程序。

上完第一课后，我们被带去参观纽约大学计算中心。该中心有两台配合使用的计算机，其中较小的一台为 I.B.M. 1401 计算机，较大的一台为 I.B.M. 7090计算机，前者用来为后者准备数据。在这样的组织安排下，向 7090 机提供数据可以分成三步。首先，把绿色表格纸上的程序转移到穿孔卡片上，卡片上打成的穿孔图案代表 FORTRAN 符号。（这是个机械作业，很象打字，通常由穿孔员进行，但一般人也很容易学会。）其次，将卡片叠送入 1401 机，由后者“阅读”，即以电子方法来感应识别穿孔图案。第三，1401 机把卡片上的信息记到磁带上，磁带上的信息则由 7090 机直接读出。在大多数情况下，1401 机还控制着输出 7090 机所得结果的打印机，以每分钟约六百行的速度运行。计算机工作时，运行情况完全在一个纯灰色金属箱内进行。但磁带盘则安置在有玻璃窗的若干

个小柜内，每个小柜装有一个磁带盘。观看计算机运算时，看到许多磁带盘轮流运行的情景，是非常引人入胜的。当某一磁带机的灯光一亮，柜内的磁带盘便旋转几分之一秒。然后，另一台磁带机的灯光亮时，它柜内的磁带盘又转动起来，如此不断地进行下去。

上第二堂课时，教师给我们布置了头一回课外作业。这时我才知道，尽管 FORTRAN 看来简单，但细微的差异却很多。所布置的作业题是有关偿还抵押借款的。同计算所得税一样，最后要列出一个公式，亦即求出相当于抽象变数的具体数值。但这个公式比所得税公式更加复杂。在正式写程序之前，老师建议我们先画个"框图"。框图中标明解题时须采取的各步骤间的逻辑关系。画框图时，必须井然有序地拟出自己独立解该题时须进行的各项数学运算及逻辑运算。当我们自己解题时，往往是想当然地那样进行运算，几乎不用思索。但不能要求计算机也能想当然地那样去进行运算。尽管分析各个步骤的逻辑关系有些单调，但在几次试着不画框图而写出程序后，才知道把程序要点整理得井然有序是十分重要的。当然，程序本身必须用组成 FORTRAN 语言的各种成分及文法结构来写出。因此，在决定了合适的运算序列并将运算序列画出框图之后，我们接着就应当找出一些可将自己的命令传达给计算机的词句。寻找正好合适的词句，就象拚对益智图一样难，但一旦找到了窍门，就会觉得饶有兴味。但是 FORTRAN 的文法非常严格，如果在规定必须有逗点的地方漏掉这个逗点，计算机就会干脆不接受这个程序。只是在三番五次失败之后，我才写成了一份文法细节全都正确的程序。在拿到第一次打印好的计算结果时，我感到十分满意，因为我终于同计算机对上了话。这使我回想起当年在巴黎学了一星期法文之后，发现自己能够同看门人进行简短但有效的会话时的那种

心情。

第二次课外作业题比第一次难。这回我了解到向计算机输入错误的 FORTRAN 程序会发生什么情况。我编了一个自认为似乎很简单的程序，把它交给了穿孔员。过了一会儿，我拿到了打印出来的计算结果。一开头是日期和下列标题："709/7090 FORTRAN 诊断程序结果"，下面印出了我所写的公式，接着是一则简要的说明："多了个左括弧"(同其他文法一样，FORTRAN 文法要求左右括弧对应)。我看了一下公式，发现穿孔员把代表除的符号误认为左括弧了。到了计算机上，该公式便成为：

```
STDV = (SUM(GN)**.5
        和
```

该式本来应该是：

```
STDV  = (SUM/GN)**.5
标准除法    总数
```

当我问起究竟出了什么问题时，人们把计算机接受 FORTRAN 程序后所采取的一些步骤告诉了我。计算机接受 FORTRAN 程序后，首先对它进行扫描，检查有无简单的文法错误，例如，有没有漏掉逗点。如果一切看起来都很正常，计算机就会按常规继续工作，将 FORTRAN 译成机器语言，(即一般所说的翻译过程)，并接着执行指令。这就同人们接到外语的命令后，先把它译成本国语言然后再遵照执行是一样的。如果计算机在第一次扫描时查出差错，它就不再继续翻译而打印出一张有诊断语的纸，以提请注意这些错误。在我那份打印的计算结果上，出现了如下一些有点象发牢骚的语句："标点符号使用得不合规则……原始程序出了差错……停止编译……停止执行命令。"

我把这些指给瓦洛维茨看，他说他一般要求计算机只要纠正差错，接着干它的活就行了，其他可不必过问。这张纸的末尾有一段记录，说明计算机查出差错的时间不到 36 秒。现

代计算机的运算速度十分惊人。我们做的大部分课外作业题(肯定是比较简单的机器运算),计算机只花了不到一分钟。在这一分钟内,它就完成了以下这些工作:一、把 FORTRAN 程序译成了机器语言;二、用人们提供的数据执行机器语言程序;三、将计算结果记在磁带上。

在我错用标点符号的这一情况下,要修改程序是颇为简单的,只要穿孔员准备一张新卡片就行了。事情很容易,只需要改一个符号。亦即穿孔员用穿孔机复制这张卡片时,只须在出错的括弧处暂停一下,打上一个分数号并继续复制原卡片的其余部分就行了。

FORTRAN 诊断程序查出的差错,大部分与标点符号有关。当然也很可能(甚至经常)出现其他类型的错误,例如逻辑上的错误。但计算机一般不检查这类错误,只检查那些介乎标点与逻辑之间的错误。有一次,我写了一个程序,其中有一个标号为 9070 的语句,但后来又把它写成标号为 9060 的语句。计算机作出反应,印出下列文字:"下列格式语句,虽曾被引用,但已从源程序中删去…9060。"

我问人家,为什么计算机不去诊断范围更广的差错呢?人们告诉我说,那样就会浪费计算机的时间。让计算机去检查标点符号的差错而由程序员去寻找更细致的差错,这样做要经济得多。

以后两周的课外作业越来越难了。我们最感兴趣的一项作业是编一个这样的程序:使计算机能将几百个数字排成递降顺序。这个问题要多次使用 FORTRAN 中的"条件"(IF)语句。这种"条件"语句早在开课第一天我们就学过。这次要用"条件"语句使计算机在各数相减求出结果为正、负或零之后进行排列和重新排列。这次练习还给我们详细介绍了 FORTRAN 中的"循环"(DO)语句(有时叫"循环"回路)。在计算

过程中，我们往往要求连续多次重复一个过程。一个简单而不太引人注意的例子是用 3 逐个乘从 1 到10各数。当然也可以写出一个分别包括十个乘法指令的 FORTRAN 程序，但这过于繁琐。假如不是十个乘法而是一万个乘法的话，那甚至就不只是繁琐的问题了。全套指令可以压缩成两个简单指令：

```
DO 5,I = 1,10
(循环)
5M(I) = 3*I
(乘)
```

这里，循环语句令计算机对十个数值执行指令 5，而指令 5 则是让计算机不管在什么情况下都要用 3 去乘整数 1 。

我们班还学习了如何使用“子程序”。算术中的某些特殊运算(例如，求平方根或对数)往往会多次用到子程序。虽然完全可以在 FORTRAN 内为求平方根而写出详细的步骤，但更经济的是一次写出一个求平方根的程序并存贮在磁带上，以备需要时马上使用。例如需要计算 F 的平方根时，只须写：

```
SQRT(F)
(平方根)
```

就行了。一般说来在用 FORTRAN 语言写程序时大约有三十个这样的子程序可用。另外还可以利用其他许多专门程序及子程序(这应归功于国际商业机器公司在怀特普莱恩斯的一个程序库)。这些程序可免费送给计算机用户，但用户必须向该程序库提供自己新编的程序。该程序库目前收藏有几千个程序而且还在不断增加。

学完两周课程后，我还不够资格算是个熟练的程序员，但我确实对 FORTRAN 能协助解决哪类问题有所了解。我们的课外作业并非直接来自物理学研究的需要，而且计算类型都是相似的。但我仍然确信自己已经有能力用计算机来解我想解决的问题了。我很清楚计算机能够做些什么工作，但不了解它到底是如何进行这些工作的。课程学完后，我决定对计算机作更多的理解并了解它的历史和未来发展的前景。

二

在今天，大多数受过教育的人都能进行算术计算。计算时，记忆肯定起着重要作用。如果记不住乘法表，最简单的算术也会变成一件可怕的事情。如今我们认为这是理所当然、人人皆知之事，但这一知识的广泛流传，为时并不太久。英国费兰提计算机制造公司的B.V. 鲍登（Bowden）在1953年写的《比思想还快》一书中题为"计算简史"的一篇光辉文章中写道：

> 几百年前，算术既不为人所知，也未得到广泛应用。1662年，当时主管英国海军部合同处的裴庇斯（Pepys）深感自己需要学习。为了学习乘法表，他黎明四时就起床，依灯勤读。他曾在剑桥大学念过书，按当时标准，应当算得上是个很有学问的人，晚年任皇家学会会长，成为牛顿的挚友。但在他任法令秘书时，却连为英皇置办木材时需要用的简单算术都不太懂，而且当时小学生的算术知识很少超过"2乘2"的水平。

他的努力显然没有白费。正如鲍登博士1663年12月所写的，裴庇斯在日记中写道："我妻子立即从床上起来，整个下午我们一起做算术习题。她已能熟练地演算加、减、乘法，因而我就不打算再拿除法去困扰她了，只是教她怎样使用地球仪。"

速算能力的差别很大并且与掌握高等数学抽象概念的能力毫无关系。一般来说，数学家都不是杰出的计算能手。计算技巧对于从事数学上的创造性工作的人来说，也不是必需

的。1961 年夏，我有机会在日内瓦同欧洲核子研究中心的一位程序设计和数值分析专家克莱因（William Klein）一道工作。他肯定是从未有过的最快的计算能手之一。当时我正在该中心从事物理研究，同一位朋友研究某一问题。约一星期后，我们研究出一个代数公式。对于这个公式的许多方面，我们都觉得蛮不错，打算对它作出评价。欧洲核子研究中心有一台大型费兰提厂出的水银牌计算机。由于当时我们对程序设计都一无所知，所以只能请求支援。于是，该中心派了克莱因参加我们的工作。他个子矮小，外表和气，看来精力充沛，是个四十来岁的荷兰人。他把我们的公式看了几秒钟，先用荷兰话喃喃自语，然后对公式的一些较复杂的部分作出数值估算。他说这样做，有助于最有效地编制出计算机程序。以前听说过，克莱因具有几乎惊人的才能，于是我问他是否考虑用心算来估算整个公式。他说，这样做，工作量太大，最好交给计算机去做。观察他进行工作，给了我很深的印象。为了了解有关他的情况，我欣然翻阅了鲍登博士的另一篇文章“思想与机器过程”。文中写道：

> （克莱因先生）能够记住 100 × 100 以下的乘法表、所有 1000 × 1000 以下的平方根以及大量的奇事（例如，3937 × 127 = 499999）。这些对他很有用，一旦需要这些数字，它们好像立刻就能在他的脑海中呈现出来。此外，由于他记得住 150 以下数字的对数值而且能记到小数第 14 位，所以必要时他把所用数字分解为因子后，再“查一下”脑中的对数值就可以算出像复利之类的总和了。他对历法也很有研究，能说出历史上任何一天是星期几。

鲍登博士继续写道：

> 克莱因心算六位数以下的乘法，比一般人用台式计算器算得更快。例如，算出六对三位数的乘积，他只需要

九秒钟；而一个有经验的人用计算器计算则需要用一分钟之久。

克莱因可以完全用心算得出

1388978361 × 5645418496 = 7841364129733165056。

这个运算包括25对二位数相乘和24次四位数的相加，共计49次计算，全部运算只用了六十四秒钟。我们好多人也都试算了一下，花的时间为六至十六分钟不等，并且除一人外，答数全都错了。

我曾就他的这种技艺问过他本人。他说他现在已经依靠计算机了，这门技艺已经有些生疏了，如今再计算鲍登博士的乘法题恐怕要花费整整两分钟。不管怎样，克莱因家族显然都具有惊人的计算才能。鲍登博士继续写道：

威廉·克莱因的哥哥利奥（战时死于盖世太保之手）同威廉几乎一样，也是个优秀的计算能手和数学家。阿姆斯特丹的斯托克维斯（Stockvis）博士曾对他们兄弟俩做过心理学测验，发现尽管他们的运算成绩非常相近，但算法完全不同。例如，威廉是利用声音来记忆数字，计算时自言自语，高声会干扰他的计算。假如他算错了，那是由于搞混两个发音相似的数字。利奥则利用视觉记忆数字。如果他发生了错误，那是由于把形象相似的数字搞混了。在童年的早期，他们兄弟俩就对数字入了迷。威廉学医，得过医学学位。在决定以计算为职业之前，他曾在医院实习过。

克莱因告诉我，战后他以数学家巴斯噶的名字作为假名，在欧洲的舞台上搞了几年心算表演。近来，他不愿意在枯燥的心算上再无谓地花费过多的时间了。这种合理的心愿同导致人们为什么要发明计算机器的这种感情是相似的。正如莱布尼兹在1671年所写："让一些杰出人才像奴隶般地把时间