

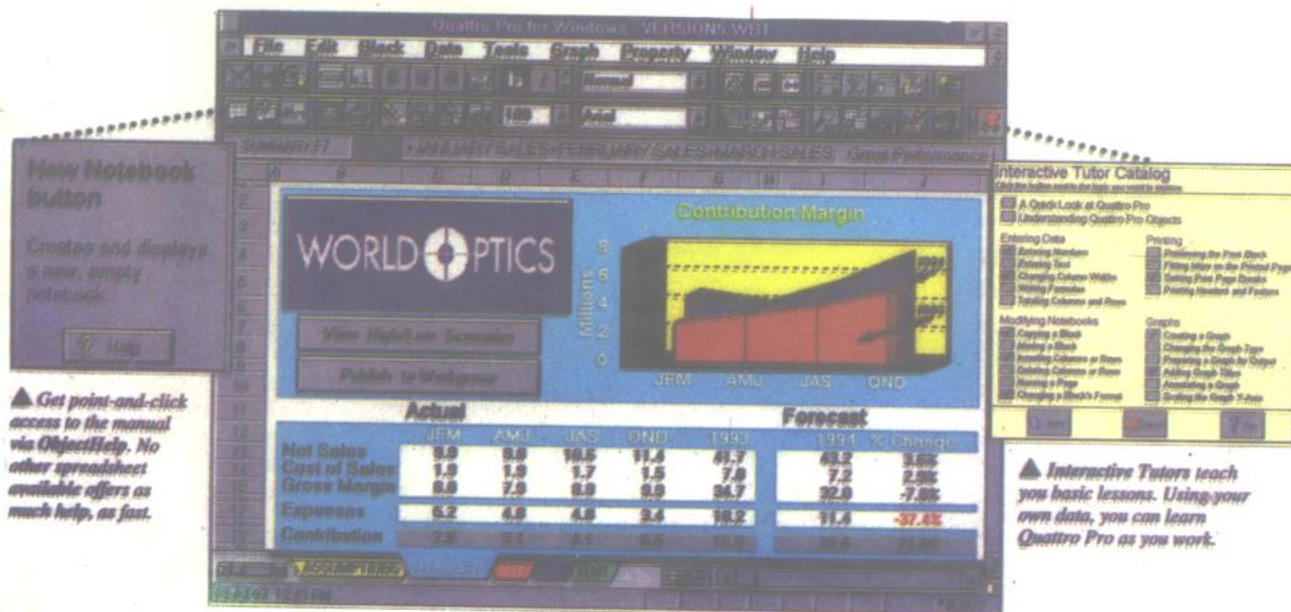
学苑出版社



程序语言总复习

(含程序设计)

黄梦熊 著
高升 张记录 改编
铁齐 刘军 审校



TP312
HM\1

计算机语言技术系列丛书

程序语言总复习

(含程序设计)

黄梦熊 著
高升 张记录 改编
铁齐 刘军 审校



025368

学苑出版社
1993.

(京)新登字 151 号

内 容 提 要

本书详细系统地介绍了程序语言、程序设计及系统分析的基本概念和最新理论,简明扼要地介绍了 BASIC, FORTRAN, PASCAL 语言,从而使读者能在短期内学会使用三种不同的语言。在此基础上还讨论了综合型问题、热门程序设计问题、重要程序设计问题等等内容。

本书内容全面,叙述清晰,通俗易懂,不仅适合于初学者,而且对于有程序设计语言知识的读者也是一本很好的参考读物。

欲购本书的用户,可直接与北京 8721 信箱联系,邮码 100080,电话 2562329。

版 权 声 明

本书繁体字中文版原名为《程式语言总复习》,由松岗电脑图书资料股份有限公司出版。版权归松岗公司所有。本书简体字中文版版权由松岗公司授予北京希望电脑公司,由北京希望电脑公司和学苑出版社独家出版、发行。未经出版者书面许可,本书的任何部分不得以任何形式或任何手段复制或传播。

计算机语言技术系列丛书

程序语言总复习(含程序设计)

著 者:黄梦熊
编 者:高 升 张记录
审 校:铁 齐 刘 军
责任编辑:徐建军
出版发行:学苑出版社 邮政编码:100032
社 址:北京市西城区成方街 33 号
印 刷:双青印刷厂
开 本:787×1092 1/16
印 张:30.125 字 数:705 千字
印 数:1—5000 册
版 次:1993 年 12 月北京第 1 版第 1 次
ISBN7-5077-0776-8/TP·8
本册定价:39.00 元

学苑版图书印、装错误可随时退换

前 言

计算机与人类的日常生活相当密切，利用计算机来增加工作效率是现代人所应具备的技能，而计算机是由程序来指挥的，因此通过练习程序语言和程序设计来理解计算机的基本知识，进而驾驭计算机解决问题是精通计算机最有效的途径之一。

本书详细系统地介绍了程序语言、程序设计及系统分析的基本概念和最新理论，简单扼要地介绍了 BASIC, FORTRAN, PASCAL 语言，从而使读者能在短期内学会使用三种不同的应用广泛的语言，在此基础上讨论了综合型问题、热门程序设计问题、重要程序设计问题等等内容。

本书的特点是：文字简练，图形及程序并举，希望能在有限的篇幅内覆盖丰富的内容，书中含有大量范例，并附有经过调试运行的程序，故使读者能直接运用这些程序。本书讨论了各类热门和重要程序设计问题，为对于要复习程序设计语言的读者无疑提供了一本内容全面的书籍。

本书是根据黄梦熊先生（台湾）编著的《程式语言总复习》一书改编而成的。本书在改编过程中力求在忠实原著的基础上，对原著中不正确之处做了适当的修改。

在本书的改编过程中得到了北京希望电脑公司秦人华老师的支持和指导，在此谨表谢意。

限于改编者的水平，书中错误和不妥之处，敬请读者批评指正。

1993 年 12 月

目 录

第一章 基本概念(一)	1
1.1 程序语言的定义及分级	1
1.2 第四代语言(4GL)	10
1.3 各种常见的程序语言的特征	11
1.4 流程图的介绍	19
第二章 基本概念(二)	28
2.1 系统的生命周期及开发阶段	28
2.2 结构化分析	35
2.3 由上而下的程序设计	36
2.4 模块化程序设计	40
2.5 结构化程序设计	41
2.6 主子程序之间常见参数传输方式—— 传址调用、传值调用、传值兼传结果调用及传名调用	55
2.7 PASCAL 程序语言参数的讨论	62
2.8 FORTRAN 程序语言参数的讨论	70
2.9 数据的安全	73
2.10 存储体的分配与释放的分类	74
第三章 BASIC 程序语言速成	78
3.1 BASIC 程序的组成及一般规定	83
3.2 常量、变量及表达式的讨论	85
3.3 REM、LET、END 及 STOP 语句	97
3.4 与输出有关的语句——PRINT、LPRINT、PRINT USING、 LPRINT USING 及输出的位置控制函数 TAB、SPC	99
3.5 与输入有关的语句——INPUT、LINEINPUT 及 READ / DATA	107
3.6 基本的控制语句——GOTO、IF-THEN-ELSE 及 ON-GOTO	112
3.7 循环语句——FOR / NEXT 及 WHILE / WEND	115
3.8 数组	122
3.9 函数	123
3.10 子程序	135
第四章 FORTRAN 程序语言速成	140
4.1 FORTRAN 语句的格式及符号名称的命名原则	140
4.2 常量及变量的讨论	142
4.3 表达式、赋值语句、DATA 语句、STOP 及 END 语句	148
4.4 输入与输出语句	153

4.5 基本的控制语句——GOTO 及 IF 语句	163
4.6 循环的控制语句——DO 语句	170
4.7 数组	175
4.8 函数及子程序	178
4.9 声明语句	188
第五章 PASCAL 程序语言速成	202
5.1 PASCAL 程序的结构及输出语句	202
5.2 数据类型及 CONST、TYPE、VAR 语句	205
5.3 输入语句、表达式及格式化输出	214
5.4 基本的控制语句——GOTO、IF 及 CASE	221
5.5 循环语句——FOR、WHILE 及 REPEAT	228
5.6 函数与过程	236
5.7 标准函数与过程	241
5.8 数组	251
5.9 记录	256
5.10 文件	261
5.11 集合	271
5.12 动态的数据结构——指针	275
第六章 综合型问题集锦	282
6.1 循环语句的讨论	282
6.2 表格或数组的建立	285
6.3 BASIC 程序语言问题集锦	286
6.4 FORTRAN 程序语言问题集锦	287
6.5 PASCAL 程序语言问题集锦	290
6.6 求执行的结果问题集	291
6.7 算术式的通式的推导	292
6.8 用限定的几种运算操作	297
第七章 超热门程序设计问题集	301
7.1 大小比较问题集 (一)	302
7.2 大小的比较问题集 (二)	304
7.3 阶乘计算问题集	313
7.4 FIBONACCI 数列及类似数列	319
7.5 函数子程序问题集	326
7.6 最大公因数(GCD)与最小公倍数(LCM)问题集	329
7.7 质数、因数及质因数问题集	335
7.8 数组的运算问题集	343
7.9 数组的处理问题集	350
7.10 数组的应用问题集	353
7.11 递归范例——汉诺塔	359

7.12	分类问题集	365
7.13	合并及更新操作问题集	371
7.14	搜索问题集	377
7.15	排列问题集	382
7.16	解一元二次方程的根	385
第八章	热门的程序设计问题集	390
8.1	数列的计算问题集	391
8.2	数字系统的转换问题集	396
8.3	各种底的乘法表的输出问题集	404
8.4	三角形的应用问题集	409
8.5	字符串的操作问题集	411
8.6	图形的打印问题集 (一)	412
8.7	成绩的计算问题集	418
8.8	求平方根	423
8.9	应用问题集锦	424
8.10	与日期相关的问题集	426
8.11	决定一系列数中任何一个数的前面三个数	428
8.12	随机数问题集	431
第九章	热门的程序设计进阶	439
9.1	无限制的数值表示法	439
9.2	将二进制数转换成葛雷码的研讨	442
9.3	典型的问题集锦	444
第十章	重要的程序设计问题集	448
10.1	解联立方程序	448
10.2	阿姆斯壮数	455
10.3	图形的打印问题集 (二)	456
10.4	求一元多次方程式的根	459
10.5	明码与密码的互换问题集	466
10.6	函数图形的打印	469
10.7	将阿拉伯数目转换为各种表示法	472

第一章 基本概念(一)

1.1 程序语言的定义及分级

一、程序语言(Programming Language)的定义

凡是能描述算法(algorithm)和数据结构(data structure)的概念者称为程序语言,它是用来指挥计算机执行赋值动作的一套规则。

二、程序语言的分级

按照程序语言比较接近计算机本身用的语言或人类自己用的语言分成低级或高级;依据人类使用程序语言的方便程度而分,程序语言可划分为五代(如图 1-1-1 所示):第一代是机器语言(Machine Language),第二代是汇编语言(Assembly Language),第三代是高级语言(High-level Language) (包括面向过程语言(Procedure-oriented Language)及面向问题语言(Problem oriented Language)),第四代是超高级语言(Very High-level Language),第五代是自然语言(Natural Language);其中机器语言及汇编语言又称为低级语言(Low-level Language),超高阶通常被称为第四代语言(Fourth-Generation Language, 4GL)(注:有些文献称汇编语言为中级语言(Medium-level Language))。

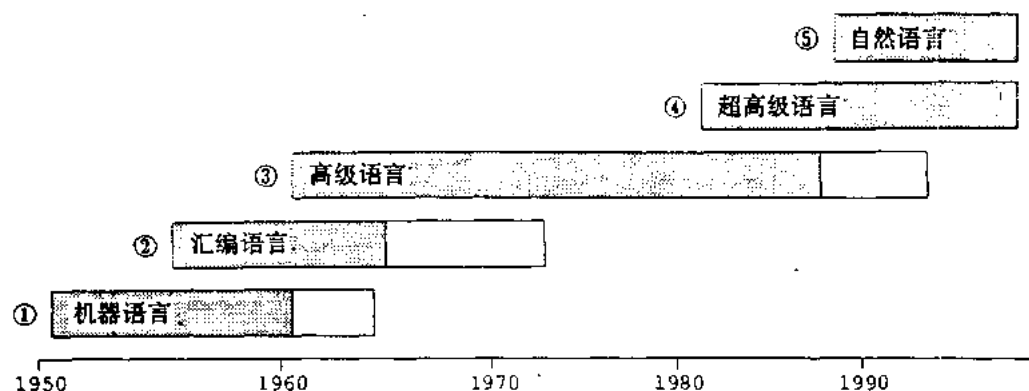


图 1-1-1 程序语言依使用时期划分级别(其中较深的部分表示大量使用的时期,白色的部分表示正逐渐减少使用中)

程序语言所隶属的级别愈新,指示计算机完成工作所需的命令就愈少,提供人(程序设计员)与(计算机)交谈的能力也愈强,其中第一代至第三代程序语言属于过程性程序语言(Procedure Language),亦即必须指示计算机“做什么”及“如何去做”,而第四代

及第五代程序语言属于非过程性程序语言(Nonprocedure Language), 亦即只须指示计算机“做什么”, 而可以不用指示“如何做”。

尽管后面几代程序语言指示计算机工作比较简易, 但前面几代程序语言仍有许多优点, 因此以上各代程序语言当前都仍在使用中。

(一) 第一代程序语言——机器语言

机器语言(Machine Language)是电子计算机能够直接执行的语言, 它是由直接代表一系列计算机命令的一串 0 与 1 连续排列而成; 不同类型的计算机拥有不同的命令集, 因此所使用的机器语言也不一样, 亦即机器语言是一种与机器有关(machine dependent)的语言, 必须先精通所使用的计算机的结构, 才能直接用机器语言写程序, 程式的编写及维护都很困难。

(二) 第二代程序语言——汇编语言

直接读或编写机器语言程序有困难, 于是将每一条机器语言基本命令用一条助记符号(mnemonic symbol)取代, 这些助记符号统称为汇编语言(Assembly Language)又称为低级语言(low-level language), 因此汇编语言是一种符号化程序语言, 也是最接近机器语言的程序语言, 除了宏(MACRO)命令以外, 大都可将逐条命令转化为机器语言; 不同型的计算机提供不同的汇编语言而且不能通用, 所以汇编语言是与机器有关的语言, 用汇编语言编写程序方便多了, 但计算机却看不懂, 因此计算机厂商提供一套系统程序用来将汇编语言程序转换成机器语言; 此套程序称为汇编程序(Assembler)。

(三) 第三代程序语言——高级语言(包括面向过程语言及面向问题语言)

汇编语言写的程序和人们日常所使用的语言相差甚多, 而且设计汇编语言程序除了要熟悉汇编语言的用法以外, 尚须对于计算机的软硬件特性有所了解, 学习及使用都不方便; 因此专家们依据科学、商业或统计等需要而开发一些比较接近人类日常语言的程序语言称为高级语言(High-level Language), 例如: FORTRAN、COBOL、BASIC、PASCAL、ALGOL 及 PL/1 等都属于高级语言, 这些程序语言都只注重程序的编写而与计算机本身无关, 因此又称为面向过程语言 Procedure-oriented Language)。

高级语言程序必须通过计算机厂商提供的系统程序(例如: 编译程序(Compiler)或解释程序(Interpreter))转换成机器语言(或先转换成汇编语言, 再转换成机器语言)后才能执行; 用高级语言设计程序不需要熟悉计算机的结构, 而且只要经各种计算机的编译程序(或解释程序)转换后, 就可以在该种计算机上执行, 因此高级语言是与机器无关(machine independent)的语言。

面向问题语言(Problem-oriented Language)乃是为解决某些特殊问题而设计的, 它使得程序的编写者能全力地进行问题本身的规划, 而尽可能不必去管计算机本身及处理程序的细节, 因此只要对该类问题有相当了解的人, 就能很快地学会使用这类面向问题语言, 有效地解决问题; 例如: GPSS(General Purpose System Simulator)及 SLAM 为适用于系统模拟的面向问题语言, STRESS(Structural Engineering System Solver)为适用于解决结构工程问题的面向问题语言。

(四) 第四代程序语言——超高级语言

超高级语言(Very High-level Language)通常被称为第四代语言(Fourth-Generation Language, 4GL), 系由一系列集成性模块软件构成的高生产力语言(High-Productively Language), 而不是单一种程序语言, 这些模块软件必须让用户能够很容易的使用而且具有很高的生产力。

第四代语言属于非过程性程序语言(Nonprocedure Language), 亦即只须指示计算机“做什么”, 而可以不用指示“如何去做”; 第四代语言是一种高生产力的语言, 生产力大约是第三代语言的十倍, 例如: RAMIS 是 1970 年 Mathematica 公司的 Jerry Cohen 开发出的第一个第四代语言, FOCUS 是能在 IBM PC 执行的第四代语言。

注: 本书将另开专节介绍第四代语言(4GL)。

(五) 第五代程序语言——自然语言

自然语言(Natural Language)是人类日常生活所用的语言, 例如: 中文、英文、日文、德文等都属于自然语言。

自然语言能将人类的指示(包括俚语及不符文法的句子等)转换成机器能处理的码, 若指示不明确则机器会礼貌地请求进一步解释。

自然语言又称为知识库语言(Knowledge-Based Language), 系以处理某特定领域的信息为基础的语言; 利用自然语言进出知识库是人工智能(Artificial Intelligence)的基础, 此外专家系统(Expert System)是知识库系统的一种。

自然语言的特点是存取数据很容易, 但现阶段的自然语言软件通常一次只能处理一个句子, 而且处理较复杂的逻辑还不成熟; 当前已有专家正在开发人类与计算机之间交谈不受限制的纯自然语言。

注: 有些文献将程序语言划分为第六代: 第一代是机器语言, 第二代是汇编语言, 第三代是面向过程语言及面向问题语言, 第四代是查询语言(Query), 第五代是应用产生器(Application Generator), 第六代是自然语言; 仅介绍查询语言及应用产生器如下:

(一) 第四代语言——查询语言

查询语言(Query)是采用类似英语的高级语言命令存取及编排数据, 而且在限定的范围内提供算术运算处理, 使用户能在短时间内完成繁杂的查询作业及产生所需求的报表; 查询语言的用户只需指示计算机“做什么”, 而不必指示“如何去作”, 由计算机的系统软件自动产生对应的程序, 因此查询语言将能大量地提升整个系统的生产力, 例如: NO-MAD2 及 EASYTRIEVE Plus 等都属于查询语言。

(二) 第五代程序语言——应用产生器

应用产生器(Application Generator)又称为代码产生器(Code Generator)是为协助计算机专业人员开发整体咨询系统而设计的, 系由程序设计师以填空的方式输入欲建立的应用系统的详细规格(包括数据库的描述、数据的建立文件及文件维护、查询书面的配置及有关报表的格式等),

然后应用产生器软件自动选取有关的模块(Module)集成成为应用系统。

程序语言所隶属的级别愈新,指示计算机完成工作所需的命令就愈少,提供人(程序设计员)机(计算机)交谈的能力也愈强,其中第一代至第三代程序语必须指示电脑“做什么”及“如何去做”,但第四代至第六代程序语言只须指示计算机“做什么”而不用指示“如何去做”。

三、高级语言(High-Language)的优点

(一) 学习容易,学习的时间较短

高级语言所用的语句很接近人们日常所用的数学式或英文句子,因此学习容易而且能在短时期内学会使用。

(二) 对于人力及管理的需求都比较少

许多报告指出:程序设计员在单位时间内所能完成调试过的(debug)语句的数目与使用的程序语言无关,而平均一条高级语言语句能达成十条汇编语言语句的功能,因此用高级语言编写程序能减少许多对于人力及管理的要求。

(三) 程序的调试(debug)容易

程序设计员不必考虑计算机的结构,而能够全力查找高级语言程序中算法及设计上的错误,而且高级语言的语句简洁,因此很容易找出错误及进行修改。

(四) 具有与机器无关(machine independent)的优越性

用高级语言设计程序不需要熟悉计算机的结构就可进行,而且只要经各种计算机的编译程序(或解释程序)转换后,就能在该种计算机上执行,因此高级语言具有与机器无关的优越性。

(五) 编写较大的应用程序时,不至于为了琐碎的程序细节而影响设计的进行。

(六) 卓越的文件(documentation)能力

许多高级语言的结构本身就拥有卓越的文件记载能力。

四、编译程序(Compiler)和解释程序(Interpreter)的讨论

(一) 定义:

1. 编译程序(Compiler)是计算机厂商提供的系统程序之一,功能是将高级语言(例如: FORTRAN、COBOL、PASCAL 等程序语言)所写的程序的全部语句转换成能被机器接受的等效目标程序(Object Program)(即机器语言)。
2. 解释程序(Interpreter)是计算机厂商提供的系统程序之一,功能是按照高级语言(例如: BASIC 程序语言)所写的程序执行时语句的逻辑顺序,逐命令转换为机器语言命令并且执行。

(二) 区别:

1. 编译程序及解释程序的输入都是高级语言所写的程序,但编译程序只产生等效的目标程序(object program)(即机器语言程序)却不执行,而解释程序

按照输入程序执行时语句(statement)的逻辑顺序逐一地编译并且执行。

2. 编译程序按照程序实体上语句的输入顺序进行处理, 而解释程序则依照程序执行时语句的逻辑顺序进行处理。
3. 编译程序对于程序中每个语句都只处理一次, 而解释程序依照执行时语句的逻辑顺序进行处理, 因此可能发生某些语句被重复处理许多次(当这些语句没有被执行到时, 例如当错误发生时才执行的程序段可能就不会被执行)。

(三) 优缺点的比较:

1. 编译程序的优缺点:

(1) 优点:

- ① 节省执行的时间——每次执行时只要把已编译完成的目标程序加载存储器直接执行即可, 而不必再进行原始程序的编译程序, 因此将节省可观的执行时间。
- ② 节省编译的时间——编译程序按照原始程序实体的输入顺序进行编译, 每个语句都只编译一次, 执行时则直接执行产生的目标程序, 因此即使某些语句重复执行许多次也不须要再编译, 所以节省了许多的编译时间。

(2) 缺点:

- ① 执行时需要比较大的存储器空间——原始程序中单一个语句经过编译程序编译转换后可能产生成千上百个机器语言命令, 编译程序将原始程序内所有的指令都给予转换产生目标的程序存放到协存储体(例如: 磁盘)中; 欲执行该程式时, 由加载程序将目标程序加载存储器后执行, 因此执行时需要比较大的存储器空间。
- ② 存储时需要比较大的协存储体空间——将原始程序及目标程序一并存储, 因此需要比较大的协存储体空间。
- ③ 执行阶段发生错误的处理比较费时及麻烦——先将原始程序修改, 然后重新编译产生目标程序, 再重新执行。

2. 解释程序的优缺点:

(1) 优点:

- ① 执行时所需要的存储器空间比较小——解释程序按照原始程序执行时语句的逻辑顺序逐一地编译转换并且执行, 而不是等到原始程序中所有命令都转换后才执行, 因此需要的存储器空间比较小。
- ② 存储时需要的协存储体空间比较小——只要将原始程序存储即可。
- ③ 执行阶段发生错误的处理比较简单而且节省时间——可直接修改语句或数据后继续转换及执行。

(2) 缺点:

- ① 执行所需要的时间比较长——每执行一次解释程序将原始程序直

接输入，然后依照执行时语句的逻辑顺序逐一编译转换并且执行，因而花费许多时间在叙述的编译上，因此执行所需要的时间比较长。

- ② 编译所需要的时间比较长——解释程序将原始程序直接输入，然后依照执行时语句的逻辑顺序逐一地编译转换并且执行，若某些语句执行许多次，则每次都需要进行编译转换才执行，因此需要比较长的编译时间。

(四) 适用场合：

1. 编译程序适用于要求能快速恢复、存储器够大及作业环境变动不大的场合。
2. 解释程序适用于不须快速恢复、存储器较小及作业环境经常变动的场合。

范例 1

何谓“程序语言”？依使用方便程度而言可分为几个层次？请分项说明。
计算机程序语言的类有那些？试比较其间差异及优劣。

解：请参阅本节一、及二。

范例 2

语言大致可分为“自然语言”与“人工语言”，其主要区别何在？又“程序语言”属于何者？

解：自然语言(natural language)是人类日常生活所用的语言，例如：中文、英文、日文等都属于自然语言，而人工语言(artificial language)是人类所制造的语言，例如：BASIC、FORTRAN、COBOL 等都属于人工语言；程序语言属于人工语言。

范例 3

汇编语言与高级语言有何基本上的差异？

解：汇编语言与高级语言基本的差异是汇编语言是与机器有关(machine dependent)的语言，不具移植性(portable)，而高级语言是与机器无关(machine independent)的语言，具有高度的移植性；换言之，不同型的计算机提供不同的汇编语言而且不能通用，而高级语言程序只要经各种计算机的编译程序(或解释程序)转换就能在该种计算机执行。

类似例题

试简单说明(50 字以内)以下类似名词的主要差别：

低级计算机语言(Low-Level Language)、高级计算机语言(High-Level Language)

解：低级计算机语言是与机器有关(machine dependent)的语言，不具移植性(portable)，而高级计算机语言是与机器无关(machine independent)的语言，具有高度的移植性。

范例 4

什么是机器语言(Machine Language)? 什么是汇编语言(Assembly language)? 什么是高级语言(High Level Language)? 这三者关系如何? 有什么不同?

解：机器语言、汇编语言及高级语言的定义请参阅本节二、(一)(二)(三)；汇编语言及高级语言都必须转换成机器语言才能执行，不同是机器语言及组合语言是与机器有关的语言，不具移植性，而高级语言是与机器无关的语言，具有高度的移植性。

范例 5

试述低级语言、高级语言和第四代语言的优缺点。

解：(一) 低级语言的优点：

1. 执行的速度快。
2. 能充分利用硬件的特性，因此弹性比较大。

低级语言的缺点：

1. 学习、书写及维护都很困难。
2. 属于与机器有关(machine dependent)的语言，不具移植性(portable)，亦即低级语言程序无法在不同的计算机上执行。

(二) 高级语言的优点：

1. 学习容易而且能够短期内速成。
2. 书写、调试(debug)及维护都很容易，而且具有卓越的文件能力。
3. 对于人力及管理的需求都比较少。
4. 属于与机器无关(machine independent)的语言，具有高度的移植性。

高级语言的缺点：

1. 执行的速度比较慢。
2. 无法充分利用硬件的特性，因此弹性比较小。

(三) 第四代语言的优点：

1. 比高级语言容易学习而速成。
2. 具有很高的生产力——属于非过程性(nonprocedure)程序语言，只须指示电脑“做什么”，而可以不用指示“如何做”。

第四代语言的缺点：

1. 处理比较复杂的逻辑还不成熟，仅能解决一般性的问题。
2. 无法充分利用硬件的特性，因此弹性比较小。

范例 6

低级程序语言包括机器语言和_____语言两种。
能把原始程序转换成目标程序的系统软件是_____。

解: 汇编、编译程序(Compiler)。

范例 7

单选或多选题:

An interpreter

- (A) does not produce object program
- (B) translate the language statement and then execute it immediately
- (C) does not produce object program and execute the source program directly
- (D) execute program faster than compiler does.

解: (A) (B) (C)

范例 8

Suppose we already have the following three programs

Program A—which is a BASIC interpreter written in FORTRAN language

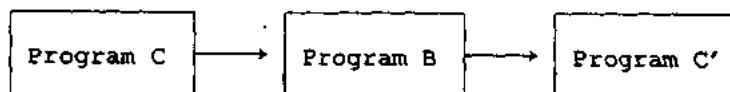
Program B—which is an Assembler written in machine language

Program C—which is a FORTRAN compiler written in Assembly language

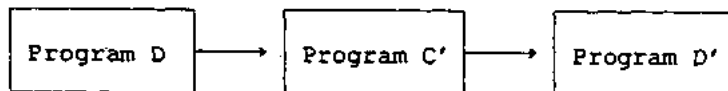
Try to solve the following individual problem:

- (1) To compile a FORTRAN application program(Called program D).
- (2) To assembly an Assembly application program(Called program E).
- (3) To interpret a BASIC application program(Called program F).

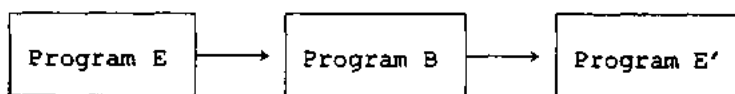
解: (1) ① 将FORTRAN compiler的原始程序C通过Assembler(程序B)转换成可执行的机器语言程序 C'。



- ② 将FORTRAN应用程序D通过FORTRAN compiler(程序C')编译生成目标的程序 D'。

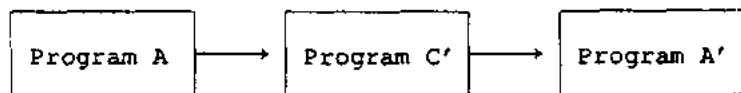


- (2) 直接将Assembly应用程序E通过Assembler(程序B)转换成可执行的机器语言程序 E'即可。

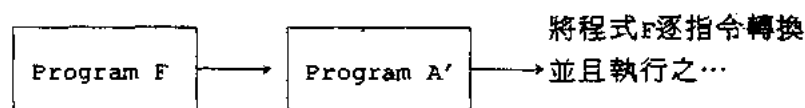


(3) ① 与(1)①相同。

② 将BASIC interpreter 的原始程序A通过FORTRAN compiler(程序C)编译生成目的程序 A'。



③ 通过BASIC interpreter(程序A'), 按照BASIC应用程序F执行时语句的逻辑顺序逐命令转换并且执行。



范例 9

解释名词:

(一)Assembler

(二)Compiler

(三)interpreter

Describe the differences between a compiler and an interpreter.

Compare their advantages and disadvantages.

解释 Compiler 与 Interpreter 的不同及其优缺点的比较。

试说明 COMPILER 的功能及其可能产生的输出为何?

编译(compiler)与解释(interpreter)有何不同?

Compiler 是什么? 它与 interpreter 有何不同?

解: Assembler(汇编程序)是计算机厂商提供的系统程序之一, 功能是将汇编语言(Assembly Language)写的程序转换成能被机器接受的等效机器语言。

Compiler及Interpreter的定义、区别及优缺点比较请参阅本节四。

范例 10

欲使高级语言程序能在计算机内执行, 须经过哪些步骤? 每一步骤的作用为何?

解: 欲使高级语言程序能在计算机内执行, 须经过以下步骤:

(1) 编写原始程序(Coding Source Program)

作用: 依照高级语言的语法编写程序, 以期满足目标需求。

(2) 输入或加载原始程序(Input or Load Source Program)

作用: 将原始程序从键盘或读卡机输入存储器或将已存储在磁盘的原始程

序载入存储器准备编译。

- (3) 编译(Compile)原始程序: 如编译结果发现错误, 则修改原始程序, 再重新编译, 直到完全正确为止。

作用: 产生机器语言(machine language)。

- (4) 连接(Linkage Editor)成可执行程序

作用: 将主程序引用到的输入/输出程序、函数或子程序的地址给予处理, 产生可执行的程序。

高级语言程序经过连接后即可在计算机内执行。

1.2 第四代语言(4GL)

一、定义

第四代语言(Fourth-Generation Language, 4GL)是由一系列集成性模块软件构成的高生产力语言(High-Productivity Language), 而不是单一种程序语言, 这些模块软件必须让用户能够很容易地直接使用而且具有很高的生产力。

第四代语言属于非过程性程序语言(Nonprocedure Language), 亦即只须指示电脑“做什么”, 而可以不用指示“如何去做”。

第四代语言是一种高生产力的语言, 其生产力大约是第三代语言的十倍, 例如: RAMIS 是 1970 年 Mathematica 公司的 jerry Cohen 开发出的第一个第四代语言, FOCUS 是能在 IBM PC 执行的第四代语言。

二、第四代语言的设计原则及目标

- (一) 尽量使用易记的语句, 避免使用更深的语法。
- (二) 只须最少的操作(Minimum Work) 即能使用 4GL。
- (三) 只须最低的技巧(Minimum Skill) 即能使用 4GL。
- (四) 使开发系统所需的时间最短(Minimum Time)。
- (五) 使错误减至最少(Minimum Errors)。
- (六) 使维护所需成本及时间最少(Minimum Maintenance)。
- (七) 使成果最大(Maximum Results)。

三、第四代语言的内容

第四代语言提供了集成性的应用系统开发环境, 几乎包含各种应用软件, 其内容是以数据库为核心, 外围包括建立、更新及管理数据库的软件、查询语言、报表产生器、屏幕设计、计算的能力、…等, 同时提供决策分析的工具、文书处理、与其它计算机连线及数据通讯等功能。

范例 1

- (一)解释名词:

Fourth generation language