第员章摇悦语言概述

员援员摇悦语言的发展历史及其基本特征

悦语言是目前世界上最流行、使用最广泛、很有发展前途的一种优秀的高级程序设计语言。它的语言功能比较丰富，表达能力强，使用起来非常灵活方便。在对操作系统和系统使用程序以及需要对硬件进行操作的场合，用 悦语言明显优于其他高级语言，许多大型应用软件都是用 悦语言编写的。除此之外，悦语言还具有绘图能力强、可移植性高、数据处理能力强大的优点，因此是一门适于编写系统软件，三维、二维图形和动画的高级语言。可见，悦语言适合于作为系统描述语言，既可用来写系统软件，也可用来写应用软件。悦语言的应用如此广泛，那么它又是如何发展至今呢？

员援员援员摇悦语言的发展历史

悦语言的发展颇为有趣。它的原型是早期的 粤[illegible]语言。粤[illegible]是在 员怨远园年出现的，它是一种面向问题的高级语言，但是它离计算机硬件比较远，不适合用来编写系统程序。员怨缘园年，剑桥大学将 粤[illegible]语言发展成为 悦[illegible]（悦[illegible]语言）。悦[illegible]语言在 粤[illegible]语言的基础上更接近硬件，但是它的规模比较大，难以实现。员怨苑园年英国剑桥大学的 [illegible]对 悦[illegible]语言做了简化，并推出了 月[illegible]（月[illegible]悦[illegible]语言，员怨园年，美国贝尔实验室的 [illegible]将 月[illegible]进行了修改，并为它起了一个有趣的名字“月语言”。意思是将 悦[illegible]语言煮干，做了进一步的简化，提炼出了它的精华。并且他用 月语言写了第一个 [illegible]操作系统，在 [illegible]上实现。但 月语言过于简单、功能也极其有限。而在 员怨缘园年至 员怨苑园年间，月语言也被人给“煮”了一下，美国贝尔实验室的 [illegible]在 月语言的基础上最终设计出了一种新的语言，他取了 月[illegible]的第二个字母作为这种语言的名字，这就是早期的 悦语言。悦语言既保持了 月[illegible]和 月语言的优点（精练、接近硬件），又克服了他们的缺点（过于简单、数据无类型等）。

继 悦语言诞生后，为了使 [illegible]操作系统推广，员怨苑园年 [illegible]发表了不依赖于具体机器系统的 悦语言编译文本《可移植的 悦语言编译程序》，使 悦移植到其他机器时所需做的工作大大简化了。员怨苑员年 月[illegible]和 [illegible]出版了名著《[illegible]悦[illegible]》，从而使 悦语言成为目前世界上流行最广泛的高级程序设计语言。员怨苑员年以后，悦语言已先后移植到大、中、小、微型机上，并独立于 [illegible]和 [illegible]的发展。员怨愿园年，随着微型计算机的日益普及，出现了许多 悦语言版本。由于没有统一的标准，这

些 悦语言之间出现了一些不一致的地方。为了改变这种情况，美国国家标准研究所（粤晕灾阅）为 悦语言制定了一套 粤晕灾阅标准，成为现行的 悦语言标准。员怨怨园年，国际标准化组织 陨灶贼（陨灶贼藻则灶葬贼蚤燥灶葬造 韵则早葬灶蚤扎葬贼蚤燥灶 枣燥则 杂贼葬灶凿葬则凿蚤扎葬贼蚤燥灶）接受 员怨苑年 粤晕灾阅悦为 陨灶贼 悦的标准（陨灶贼 怨怨怨－员怨怨园）。目前所广泛使用的 悦编译系统大致都是以它为基础的，但也有些不同。在微型机上常见使用的有 酝蚤糟则燥 杂燥枣贼 悦 栽怎则遭燥 悦 匝燥则造凿 悦 月造葬糟噪 悦等版本。

对 悦语言和其他传统的高级语言作比较后发现，从掌握语言的难易程度来看，悦语言比其他语言难一些。月葬杂蚤糟 悦是初学者入门的较好的语言，云蚤扎耘则 悦也比较好掌握。对科学计算多用 云蚤扎耘则 悦或 孕葬糟噪 悦对商业和管理等数据处理领域，用 悟糖月葬泽 悦为宜。悦语言虽然也可以用于科学计算和管理领域，但 悦语言的特长并不在于此而是在于 悦语言能对操作系统和系统实用程序以及需要对硬件进行操作的场合。从教学角度看，由于 孕葬糟噪 悦是世界上第一个结构化语言，所以曾一度被认为是计算机专业的比较理想的教学语言，但 悦语言也是理想的结构化语言，且描述能力强，也适于教学。因此，悦语言已经成为被广泛使用的教学语言。

员.员 悦语言的基本特征

悦语言发展如此迅速，而且成为最受欢迎的语言之一，主要因为它具有强大的不同于其他语言的功能。许多著名的系统软件，如 阅月葬泽 Ⅲ 孕葬糟噪 阅月葬泽Ⅳ 都是由 悦语言编写的。用 悦语言加上一些汇编语言子程序，就更能显示语言的优势了，像 孕则燥 耘 宰灶燥 郧燥造凿 杂贼葬则 等就是用这种方法编写的。归纳起来 悦语言具有下列特点：

员）悦语言是中级语言

悦语言允许直接访问物理地址，能进行位（遭蚤贼）操作，能实现汇编语言的大部分功能，可以直接对硬件进行操作。悦语言可以像汇编语言一样对位、字节和地址进行操作，而这三者是计算机最基本的工作单元。它把高级语言的基本结构和语句与低级语言的实用性结合起来，使得 悦语言既是成功的系统描述语言，又是通用的程序设计语言。

一般的将各类语言分类如下：

高级语言 月葬杂蚤糟 悦
云蚤扎耘则 悦
悟糖月葬泽 悦
孕葬糟噪 悦
粤凿葬
酝燥凿耘造 悦
中级语言 悦
云蚤扎耘则 悦
宏汇编
低级语言 汇编语言

圆）简洁紧凑、灵活方便

悦语言一共只有 猿缘个关键字，怨种控制语句。程序书写自由，主要用小写字母表示。下面将 悦语言与 孕葬糟噪 悦语言做一比较，如表 员.员所示。

表 员源摇悦语言与 孕窜糟蹋语言的对比

悦语言	孕窜糟蹋语言	含义
{摇摇}	月泽阴星… 耘晕阅	复合语句
蚤藻 杂;	陈云 (藻 栽泽晕杂	条件语句
蚤皑酝	灾窜砸 蚤阴星栽泽阴砸云;	定义 蚤为整型变量
蚤皑葬 [员园 ;	灾窜砸 葬 窜砸窜再 [员源园 韵云阴星栽泽阴砸;	定义 葬为整型一维数组
蚤皑枣() ;	云泽晕情躁星枣() 阴星栽泽阴砸;	定义 枣为返回整型值的函数
蚤皑* 责;	灾窜砸 责: ↑ 阴星栽泽阴砸;	定义 责为指向整型变量的指针变量
蚤=1;	蚤 越蚤+1	赋值语句,使 蚤+1→蚤
蚤++, 或++蚤	蚤 越蚤+1	蚤自增值 员 蚤+1→蚤

从比较中我们可以看到,悦程序比 孕窜糟蹋简练,源程序短,压缩了一切不必要的成分,使得输入程序的工作量更少。

猿爱运算符丰富

悦的运算符包含的范围很广泛,共有 猿源个运算符。悦语言把括号、赋值、强制类型转换等都作为运算符处理,从而使 悦的运算类型极其丰富,表达式类型多样化,灵活使用各种运算符可以实现在其他高级语言中难以实现的运算。

源爱数据结构丰富

悦的数据类型有 :整型、实型、字符型、数组类型、指针类型、结构体类型、共用体类型等。它具有现代化语言的各种数据结构,能用来实现各种复杂的数据类型 (如链表、树、栈等) 的运算。在 悦语言中引入了指针概念,使用起来比 孕窜糟蹋更为灵活、多样,使程序效率更高。另外 悦语言具有强大的图形功能,支持多种显示器和驱动器,且计算功能、逻辑判断功能强大。

缘爱悦语言是结构式语言

结构式语言的显著特点是代码及数据的分隔化,即程序的各个部分除了必要的信息交流外彼此独立。这种结构化方式可使程序层次清晰,便于使用、维护以及调试。悦语言是以函数形式提供给用户的,这些函数可方便地调用,并具有多种循环、条件语句控制程序流向,从而使程序完全结构化。它层次清晰,便于按模块化方式组织程序,易于调试和维护。

远爱悦语法限制不太严格、程序设计自由度大

一般的高级语言语法检查比较严,能够检查出几乎所有的语法错误,而 悦语言允许程序编写者有较大的自由度。例整型量与字符型数据以及逻辑型数据可以通用。

苑爱悦语言程序生成代码质量高,程序执行效率高

悦语言程序生成的代码一般只比汇编程序生成的目标代码效率低 员园豫 原园园豫。

愿爱悦语言适用范围大,可移植性好

悦语言有一个突出的优点就是适合于多种操作系统,如 阅晕陨载杂、载陨晕阅韵杂。同时,悦语言也适用于多种计算机机型。

由于 悦语言的这些优点,悦语言的应用领域越来越广,许多大型的软件都采用 悦语言编写,这主要是因为 悦语言良好的可移植性和较强的硬件控制能力。除此之外,悦语言已经

成为被广泛使用的教学语言。而且,悦语言强调程序设计的灵活性,放宽了对语法定义的限制,使得 悦语言的应用面更为宽广。

以上只是介绍了 悦语言的最容易理解的一般特点,至于 悦语言内部的其他特点,在深入了解了 悦语言以后,各位自己就会慢慢发现和领会。

员猿窑猿 简单的 悦程序介绍

本节主要介绍几个简单的 悦程序,通过从这几个程序的分析中感受 悦程序的一些特性。

【例 员猿窑员】

```
#include <stdio. h >
main (  )
 {
   printf ("\nHello World !\n") ;
}
```

程序运行结果为:

```
Hello World !
```

虽说这个程序只有 缘行,但是这是一个 悦语言的经典的程序。它牵涉到了 悦语言的函数调用、文件的预先声明处理,以及一个完整的“ 皂葬增()" 主函数。在这个程序中,第一行的“ 粹增糟燥蚤灶贼.澡" 是一个对 澡糟糟燥蚤文件的预先声明和调用,因为以后的“ 责则蚤灶贼枣()" 函数是在这个文件中描述和定义的。在 悦语言中,为了程序的顺利执行,所有的变量和函数在使用之前都必须声明。第二行的“ 皂葬增()" 是主函数名,在用 悦语言编写程序时,每一个程序都必须有一个“ 皂葬增()" 函数 (主函数) 且只能有一个主函数。在对程序进行编译时,编译程序会找到“ 皂葬增()" 函数作为程序的入口来编译程序。第三行和第五行的大括号“ {" “} " 是主函数范围限定符,它们规定了主函数的范围。“ {" 和“} " 必须成对出现,否则会出错。第四行的“ 责则蚤灶贼枣(" 悦语言是区分大小写的!撰以 ;" 是一个对输出函数的调用,双引号内的字符按原样输出,“ 撰以" 是换行符,即在输出“ 匀藻造造燥 宰燥则造凿!" 后回车换行,语句最后有一分号。在 悦语言里没有专用的输入、输出语句,而是调用函数“ 责则蚤灶贼枣()" 来实现输出的。撰以 是传给这个函数的参数。

【例 员猿窑圆】

```
# include <stdio. h >
main (  )                  / * 主函数 */
 {
    int x,y,sum ;          / * 定义变量 */
    x =46 ;y =100 ;
    sum =x +y ;            / * 求两数之和 */
    printf ("sum is %d\n", sum)  ;
}
```

此程序的功能是求出两个已给出的整数的和。为了便于理解,在程序中用 辕 …… *辕

加注汉字表示解释，但在实际的程序运行中，注释是不起作用的。

该程序首先定义 曾赠为整型变量，然后通过赋值运算符“ 越”分别将 曾赠赋值为 源元员圆 在程序的第五行中，将 曾回赠的值赋给变量 泽皂，在第六行中“豫凿”是输入输出的“格式字符串”，它主要用来指定输入输出时的数据类型和格式，在以后的章节中将详细介绍与学习。本程序中，“豫凿”表示输出的是“十进制整数类型”的数，而 责藻藻函数中的 泽皂是要输出的变量。

最终程序运行后的结果为

```
sum is 146
```

【例 员.圆】

```
# include <stdio. h >
main ( )                                  / * 主函数 * /
 {  int x,y,z ;                           / * 变量说明 * /
    scanf ("%d%d",&x,&y) ;                / * 输入变量 x 和 y 的值 * /
    z = max (x,y) ;                       / * 调用 max 函数 * /
    printf ("max = %d",z) ;               / * 输出 = max (x,y) * /
}

int max (int a,int b)                     / * 定义 max 函数 * /
 {  int c ;                               / * 声明部分，定义变量 * /
    if  (a > b)  c = a ;
    else c = b ;
    return  (c) ;                         / * 返回 c 的值，把结果返回主调函数 * /
}
```

此函数的功能是输入两个整数，输出其中的最大数。上面例中程序的功能是由用户输入两个整数，程序执行后输出其中较大的数。本程序由两个函数组成，主函数和 皂藻藻函数。函数之间是并列关系，可从主函数中调用其他函数。皂藻藻函数的功能是比较两个数，然后把较大的数返回给主函数。皂藻藻函数是一个用户自定义函数，因此在主函数中要给出说明。可见，在程序的说明部分中，不仅可以有变量说明，还可以有函数说明。关于函数的详细内容将在以后的学习中详细介绍。在程序的每行后用 辕 和 * 辕括起来的内容为注释部分，程序不执行注释部分。

例 员.圆中程序的执行过程是，首先请用户输入两个数，回车后由 泽藻藻函数语句接收这两个数送入变量 曾赠中，然后调用 皂藻藻函数，并把 曾赠的值传送给 皂藻藻函数的参数 葬遭 在 皂藻藻函数中比较 葬遭的大小，把大者返回给主函数的变量 扎，最后在屏幕上输出 扎的值。

通过例子，我们得出 悦源程序的结构特点：

（员）一个 悦语言源程序可以由一个或多个源文件组成。每个源文件可由一个或多个函数组成。一个源程序不论由多少个文件组成，都有一个且只能有一个 皂藻藻函数，即主函数。也可以包含一个 皂藻藻函数和若干个其他函数，函数是 悦语言的基本组成单位。被调用的函数可以是系统提供的库函数（例如 责藻藻增和 泽藻藻函数），也可以是用户根据需要自己编

写设计的函数（例如例 员源中的 皂葬糟函数）。

（圆）源程序中可以有预处理命令（蛋凿糟藻命令仅为其中的一种），预处理命令通常应放在源文件或源程序的最前面。

（猿）悦程序的执行总是先从 皂葬蚤函数开始，而不论 皂葬蚤函数在整个程序中的确切位置（皂葬蚤函数可以放在程序的最前面，也可以放在程序的最后面，或者是放在其他的一些函数的前面或后面）。

（源）每一个说明、每一个语句都必须以分号结尾。例如：澡藻造造漏 越燥则造凿;分号是悦语句的必要组成部分，即使是程序中最后一个语句也应该包含分号，但注意预处理命令、函数头和花括号“｝”之后不能加分号。

（缘）悦语言本身并没有输入输出的语句，有关输入输出的操作都是通过调用库函数澡藻蚤造凿和 责则蚤灶贼枣等函数来完成的。由于输入输出操作牵涉到具体的计算机硬件设备，把输入输出操作放在函数中处理，使悦对输入输出实行“函数化”，就可以使悦语言本身的规模较小，编译程序更为简单，程序的可移植性增强，从而能够方便、容易地在各种机器上实现。

（远）标识符、关键字之间必须至少加一个空格以示间隔。若已有明显的间隔符，也可不必再加空格来间隔。

（苑）在程序语句旁边可以用 辕* …… *辕来对悦程序中的语句作必要的注释，以增强程序本身的可读性。

另外，初学者从书写清晰，便于阅读、理解、维护的角度出发，在书写程序时应遵循以下规则：

（员）一个说明或一个语句占一行。

（圆）用｛摇｝括起来的部分，通常表示了程序的某一层次结构。｛摇｝一般与该结构语句的第一个字母对齐，并单独占一行。

（猿）低一层次的语句或说明可比高一层次的语句或说明缩进若干格后书写。以便看起来更加清晰，增加程序的可读性。

在编程时应力求遵循这些规则，以养成良好的编程风格。

本章小结

在本章中介绍了当今被广泛使用的优秀高级程序设计语言悦语言的发展历程。

（员）悦语言由早期的粤悦灶郧语言发展而来，先后经历了悦蕴藻语言、月悦蕴藻语言、月语言的发展。员怨苑远年美国贝尔实验室的 陨则葬糟 运则藻藻凿皂葬灶在月语言的基础上最终设计出了一种新的语言，这就是早期的悦语言。

（圆）悦语言自诞生以来一直被广泛地使用至今，主要是因为悦语言具有强大的生命力和不同于或优于其他的语言的特点。在本章的员援猿节中详细阐述了悦语言的愿个特点。

（猿）在本章的员援源节中介绍了猿个简单的悦程序，简要说明了悦语言构成特点和书写习惯等知识。

总的来说，本章从总体的角度介绍了悦语言的发展历史和悦语言基本特征，但有关悦

语言的具体特性还需读者在进一步的学习过程中慢慢领会。

习摇摇题

一、问答题

员遥摇请根据你的理解,说说 悦语言具备的主要特点。

员猿摇悦语言的主要用途是什么?它和其他高级语言相比有什么异同?

员源摇写出一个 悦程序的基本构成。

员缘摇上机调试本章的 猿个例题,熟悉 悦语言的上机环境。

二、编程题

员远摇请参照本章的例题,编写一个 悦程序,输出以下的信息;

```
**************
  Welcome to you!
**************
```

员苑摇根据例题编写一个 悦程序,输入 葬遭糟藻个值,输出其中最大者。

第 圆章 摇基本数据类型、运算符和表达式

前面我们已经对 悦语言有了初步的认识，相信你一定迫不及待地想要见识 悦语言的风采，那我们就从 悦语言的基本知识开始吧。本章主要介绍 悦语言中的基本数据类型、运算符和表达式，通过这些内容的学习来掌握 悦语言程序设计的基础知识。

圆.员 摇标识符

圆.员.员 摇标识符

在 悦程序中，有大量的函数、数组、变量，为了能够区别它们，我们是不是应该给它们取上名字用以区分它们呢？在 悦程序中，能够标识函数、语句标号、变量的一串字符叫标识符，悦程序中使用标识符的具体规定如下：

(员) 标识符必须由英文字母、数字或下划线组成，且首字符只能是英文字母或下划线。

正确的标识符，如：

b123　pust　_456　vfp_123

错误的标识符，如：

3d　#gfd　@ffff　a＊＊＊＊

(圆) 悦程序中区别大、小写字母。如 ：粤，葬阅等代表不同的标识符，如果在程序中用以表示变量，那么它们分别代表不同的变量。

(猿) 禁止使用关键字作为标识符。

圆.员.圆 摇关键字

关键字又叫保留字，是 悦编译系统预先定义的一些具有特定含义的标识符。猿圆个关键字如表 圆.员所示。标准 悦的关键字有存储类型符、数据类型符和语句定义符 猿类。它们只能按定义的方法加以使用，而不能作常规的标识符使用。因此，在程序中使用变量名、语句标号、函数名等标识符时，不能与关键字相同。

悦语言预处理中经常用到一些专用命令，如：

(员) 宏定义预处理命令 #

(圆) 文件包含预处理命令 #

(猿 条件编译预处理命令 裕[illegible]

这些都是 悦语言特有的编译预处理命令,其主要特征是它们均以“裕”开头,不同于前面的关键字,编译预处理命令虽然不属于关键字,但为了避免混乱,建议也把他们看成关键字,不要在程序中作为函数名或变量名使用。

表 圆[illegible]个关键字

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	

圆[illegible]悦语言的基本数据类型

与其他高级语言相比,悦语言的数据类型最多,运算符最多。因此,相应的运算规则,运算结果类型也多。有些运算过程或运算符的使用容易出错,我们在学习本节时要充分重视。悦程序输入的是数据,输出的是数据,处理的也是数据。

悦的数据类型如下:

- 数据类型
  - 基本类型
    - 整型
    - 字符型
    - 实型
    - 枚举类
  - 构造类型
    - 数组类型
    - 结构体类型
    - 共用体类型
  - 指针类型
  - 空类型

所以我们可以根据数据的属性将数据分为常量和变量两大类。

圆[illegible]常量与变量

员[illegible]常量

在程序运行过程中,其值不能被改变的量称为常量。常量的类型众多具体可分为以下几类,如表 圆[illegible]所示。

如 缘[illegible] 原[illegible]为整型常量 圆[illegible]为实型常量 ;“造,‘灶’”为字符型常量 ;“[illegible]”,“[illegible]

糟蹋糟为字符串常量。

圆援符号常量

符号常量是在程序（或程序的一部分）中指定用一个符号（标识符）代表一个常量。

【例 圆援猿】

```
# include <stdio. h >
# define PI 3. 14      / * 定义符号常量 */
main ()
{
   float r,a;
   r =2. 0;
   a = PI * r * r;
   printf ("a = %f",a);
}
```

表 圆援猿 悦语言的常量数据类型

常量数据类型	短整型常量	取值范围在 原藻则到 垣藻则之间的整数
	长整型常量	取值范围在 原圆缘缘园园园园园到 垣圆缘缘园园园园的整数
	实型常量	取值范围在 员园原猿缘到 员园垣猿缘，有效数字是 苑位
	字符常量	用单引号引起来的单个字符
	字符串常量	用双引号引起来的一串字符
	符号常量	须经宏定义才能使用
	转义字符常量	用于输入输出函数的专用常量

程序中用糟蹋糟命令行定义 孕员代表常量 猿援员源员缘，程序的运算结果是 糟藻员援园缘猿园园园。由此我们可以总结出，用符号代替一个常量，使程序更易理解，可读性好。当需要修改时只需要修改一处即可，方便、不易出错。

定义符号常量的一般格式是：

#define 符号常量 常量

这是一种预编译命令，它不同于变量，符号常量的值在其作用域内不能改变，也不能再被赋值。如再用以下赋值语句给 孕员赋值：

孕员越援藻猿园园;

是错误的；

习惯上，符号常量名用大写，变量名用小写，以示区别。

猿援变量

变量也分为若干类型，如表 圆援猿所示。

（员）变量及变量名

在程序运行过程中，其值可以改变的量称为变量。实质上，变量是程序中数据连同其存

储空间的抽象。一个变量应该有一个名字,称为变量名。一个变量在内存中占据一定的存储单元,在该存储单元中存放该变量的值。

表 圆猿.缘 悦语言的变量数据类型

变量数据类型	基本类型	整数类型	整型
			短整型
			长整型
			无符号整型
			无符号短整型
			无符号长整型
		实数类型(浮点类型)	单精度浮点型
			双精度浮点型
			长双精度浮点型
		字符类型	
		枚举类型	
	构造类型	数组类型	
		结构体类型	
		共享体类型	
	指针类型		
	空类型		

注意:

悦语言中的大写字母和小写字母被认为是两个不同的字符。如 [illegible]和 [illegible]是两个不同的变量名。

悦语言中的标识符长度无统一规定,随系统而不同。

一般地,标识符的选择通常是“常用取简,专用取繁”,尽量“见名知意”,以增加程序的可读性。

圆援圆 变量说明与变量赋值

(员) 变量说明

在 悦语言中,程序中引用一个变量,实际上是对指定存储空间的引用。因此,必须先开辟(分配)存储空间才能引用它。那么,在引用变量之前就必须先对变量进行说明,所谓变量说明即定义变量的类型。这样,在编译时就会根据指定的类型分配给该变量一定长度的存储空间,并决定数据的存储方式和允许的操作方式。这就是“先定义,后使用”。其格式如下:

类型标识符　　变量表列

例如:

```
int x,y;
```

此时定义 曾赠为整型变量,为 葬遭各分配一定的存储空间(即两个字节),并按整数形式存储和运算。

(圆 变量赋值

从运算器向变量所代表的存储单元传送数据的操作称为赋值。简单地说,就是将一个数据赋给一个变量。

在 悦语言中,赋值操作用“ 越”表示,其格式如下:

变量 =表达式

【例 圆猿圆】

```
x =1 ;        / * 把 1 赋值给 x  * /
a =2 ;        / * 把 2 赋值给 a * /
x =x +1 ;   / * 把 x 的值加 1 后在赋给 x,这时 x 的值为 2 (x 的初值为 1) * /
a =x +a ;   / * 把 x,a 相加后,赋给 a,这时 a 的值为 4 * /
```

说明:

- “ 越”是赋值符号,不是等号。
- 赋值运算的作用是将“ 越”右侧表达式的值赋给“ 越”左侧的变量。步骤是先计算表达式的值再向变量赋值。

猿援变量初始化

悦语言允许在定义变量时对变量赋初值,即变量的初始化。

例如:

```
int i =20 ;  / * 定义 i 为 int 型变量,并且初值为 20 * /
```

等价于

```
int i ;
i =20 ;
```

访问内存中的变量是通过变量名来引用变量的值。实际上,系统在编译时将每一个变量名对应一个地方,在内存中不再出现变量名而只有地址。因此,要访问变量,必须通过地址找到该变量的存储单元,再从存储单元中取出其值。比如说:

[illegible]

编译时,系统分配给 葬一个起始地址 (假设为 猿远园。这样,程序中若引用变量 葬系统找到其对应的地址 猿远园 然后从 猿远园和 猿远员这 圆个字节中取出其值 猿 然后赋予 遭变量。摇摇

圆援猿援猿摇整型数据

员援整型常量

整型常量即整常数。悦语言整常数的书写形式可用以下 猿种形式表示:

- 十进制整数。
- 八进制整数。以 园开头的数是八进制数。
- 十六进制整数。以 园曾开头的数是十六进制数。

【例 圆猿员】

```
# include < stdio. h >
main ()
{
```

```
{
    int a,b,c;
    a =50;
    b =-032;
    c =0x5b;
    printf ("a = %d,b = %d,c = %d\n",a,b,c);
}
```

程序执行结果为

a =50,b =-26,c =91

圆援整型变量

(员) 整型变量的分类

整型变量可以分猿类：

- 基本型摇摇蜜贼
- 短整型　　澡燥则贼蜜贼或 澡燥则贼
- 长整型　　造燥灶早蜜贼或 造燥灶早

它们又区分为有符号 (泽蚤早灶藻凿) 和无符号 (怎灶泽蚤早灶藻凿) 两大类。对于有符号数，存储单元的最高位用来存储符号，圆表示“ 垣”，员表示“ 原” ；对于无符号数，存储单元中全部用二进制位 (遭蚤贼) 存放数的本身，而不包括符号。无符号变量只能存放不带符号的整数，如 缘，员圆等，而不能存放负数，如 原圆，原猿等。一个无符号整型变量中可以存放的数的范围比一般整型变量中数的范围扩大一倍。如果整型量占 圆个字节，范围为 原猿员怨圆愿至猿员怨圆愿，而无符号型变量的范围为 园至远缘缘猿缘。类型标识符如表 圆愿援员所示。

表 圆愿援员类型标识符

类型标识符	所占位数	数的范围
[泽蚤早灶藻凿] 蜜贼	员远	原猿圆愿愿至猿员怨圆愿
[泽蚤早灶藻凿] 泽澡燥则贼蜜贼	员远	原猿圆愿愿至猿员怨圆愿
[泽蚤早灶藻凿] 造燥灶早[蜜贼]	猿圆	原圆员源苑源源愿猿缸至圆员源苑源源愿猿缸
怎灶泽蚤早灶藻凿 [蜜贼]	员远	园至远缘缘缘
怎灶泽蚤早灶藻凿 泽澡燥则贼[蜜贼]	员远	园至远缘缘缘
怎灶泽蚤早灶藻凿 造燥灶早 [蜜贼]	猿圆	园至源圆怨源缸苑圆缘

(圆) 整型变量的定义

悦规定在程序中所有用到的变量都必须在程序中说明其类型，即“定义”。例如：

```
int a,b;(指定变量 a,b 为整型)
long e,f;(指定变量 e,f 为长整型)
unsigned short c,d;(指定变量 c,d 为无符号短整型)
```

对变量的定义，一般是放在一个函数体的开头部分 (也可以放在程序中间，但作用域只

限于该程序块)。

(猿 整型常量的类型

整型变量可分为 基本型、短整型、长整型和 无符号型、无符号短整型、无符号长整型等类型,那么常量是否也有这些类型?在将一个整型常量赋给上述几种类型的整型变量时如何做到类型匹配?请注意以下几点:

- 一个整型常量,如果其值在 -32768~32767范围内,认为它是 基本型,它可以赋值给 基本型和 长整型变量。
- 一个整型常量,如果其值超过上述范围,而在 -2147483648~2147483647范围内,则认为它是 长整型,可以将它赋值给一个 长整型变量。
- 常量中无 无符号型。但一个非负值的整型常量可以赋值给 无符号型变量,只要它的范围不超过变量的表示数值范围即可。例如,将 65535赋给一个 无符号基本型变量是可以的,而将 65536赋给它是不行的(溢出)。
- 在一个整常量后加一个字母 远或 L则认为是 长整型常量,例如 358L、358l、0x358L等。这往往用于函数调用中。如果函数的形参为 长整型,则要求实参也为 长整型,此时用 358作实参是不行的,而要用 358L做实参。

猿愿整型常量向整型变量赋值

(员 将带符号的整型数据(基本型)赋给 长整型变量时,要进行符号扩展。方法是:如果 基本型数据为正值(符号位为 园),则 长整型变量的高 员远位补 园;如果 基本型数据为负值(符号位为 员),则 长整型变量的高位补 员;低 员远位接受整型数据的 员远位。

(圆 将一个 长整型数据赋给一个 基本型变量,要进行数值位截取。方法是:将 长整型数据中低 员远位原封不动地搬到 基本型变量中。这时应注意由于数据范围的变化出现的数据错误。

(猿 将 无符号基本型数据赋给 长整型变量,要进行数值位扩展。方法是:将高位补 园。

(源 将一个 无符号型数据赋给一个占字节数相同的整型变量(如 无符号基本型赋给 基本型或 无符号长整型赋给 长整型)就是将 无符号型数据原样送到非 无符号型变量中。这时,应注意由于数据范围变化出现的数据错误。

(缘 将非 无符号型数据赋给长度相同的 无符号型变量,就是将非 无符号型数据原样赋给 无符号型变量。请结合上述规则读下面的例子:

【例 圆猿源】

```
# include <stdio. h >
main ()
{
   int a,b,c,d;
   long x,y;
   unsigned u;
   a =53;
   b =-3;
```

— 员源 —

```
    c =65533；
    d =0xABCDE；
    u =-2；
    x =a；
    printf ("a =%d,a =%u\n",a,a)；/* %d,以带符号十进制形式输出整数 */
    printf ("b =%d,b =%u\n",b,b)；/* %u,以无符号十进制形式输出整数 */
    printf ("c =%d,c =%u\n",c,c)；
    printf ("d =%d,d =%x\n",d,d)；/* %x,以无符号十六进制形式输出整数 */
    printf ("u =%d,u =%u\n",u,u)；
    printf ("x =%ld,x =%lx\n",x,x)；/* %ld,以带符号十进制形式输出长整数 */
}
```

运行结果如下：

```
a =53,a =53
b =-3,b =65533
c =-3,c =65533
d =-17186,d =bcde
u =-2,u =65534
x =53,x =35
```

圆援援援遥实型数据

员援实型常量

实型数据就是带小数点的数，由于小数点的位置飘忽不定所以实数在 悦语言中又称浮点数。实数有两种表示形式：

（员 十进制数形式。如 员援猿猿猿，园援园园园员，园援园园园，园援园等。

（圆 指数形式。如 猿援猿猿猿藻园，猿援猿猿猿藻原园都代表 猿援猿猿猿伊员园原。但注意字母 藻或 耘之前必须有数字，且 藻后面指数必须为整数，如 藻员，藻援猿都是不合法的指数形式。

圆援实型变量

悦实型变量分为单精度（则燥葬贼型）、双精度（凿燥怎遭造藻型）和长双精度（造燥灶早凿燥怎遭造藻型）三类，对每一个实型变量都应在使用前加以定义。

如：

```
float x,y；
double z；
```

则燥葬贼型数据占据 源个字节（猿圆位），有效数字是 苑位。凿燥怎遭造藻型数据占 愿个字节（远源位），提供 员缘位有效数字。造燥灶早凿燥怎遭造藻型数据占 员园个字节（愿园位），提供 员怨位有效数字。单精度实数的数值范围约为 员园原猿愿耘原猿愿，双精度实数范围为 员园原猿员耘原猿园园，长双精度实数范围为 员园原猿怨耘原猿怨猿。应当说明，实型常量不分 则燥葬贼型和 凿燥怎遭造藻型。一个实型常量可以赋给一个 则燥葬贼型或 凿燥怎遭造藻型变量。根据变量的类型截取实型常量中相应的有效位数字。

例如，葬指定为单精度实型变量

float a；

a =222222.222；

由于 [illegible] 型变量只能接收 [illegible] 位有效数字（即 [illegible]），最后两位小数不起作用，若 [illegible] 改为 [illegible] 型，则能全部接收上述 [illegible] 位数字并存储在变量 [illegible] 中。

[illegible] 字符型数据和字符串常量

[illegible] 字符常量

字符常量的特征是只有一个字符，该字符用两个单引号括住，如‘ [illegible]、‘ [illegible] ＊ [illegible] 字符常量，而‘ [illegible] ’不是字符常量。但由于 [illegible] 语言规定，字符常量是区分大小写的，所以‘ [illegible] ’和‘ [illegible] ’两个不同的字符。

[illegible] 中有些控制字符（也称非显示字符）是无法直接用字符常量去表示的。例如，前面已经遇到过，在 [illegible] 函数中的“ [illegible] ”，它代表一个换行符，就是一个难以用字符常量表示的控制字符。这时，[illegible] 语言规定用一种特殊形式表示控制字符：就是以一个“ [illegible] ”开头的字符序列。因为“ [illegible] ”后面的字符已不再是原来该字符的作用而转为新的含义，因而称为转义字符，如表 [illegible] 所示。如‘ [illegible] ’的“灶”不代表字母 灶而作为换行符。

表 [illegible] 转义字符

字符	功能	字符	功能
[illegible]	换行	[illegible]	反斜杠字符（[illegible]
[illegible]	横向跳格	[illegible]	单引号字符
[illegible]	退格	[illegible]	双引号字符
[illegible]	回车	[illegible]	[illegible] 到 [illegible] 位 [illegible] 进制数所代表的字符
[illegible]	走纸换页	[illegible]	[illegible] 到 [illegible] 位 [illegible] 进制数所代表的字符

例如：[illegible] 使用 [illegible] 码（八进制数）表示一个字符，例如，‘ [illegible] ’表字符‘ [illegible] ’表换行，用‘ [illegible] ’表图形字符‘ ■ ’

【例 [illegible]

```
# include <stdio. h >
main ()
{
    printf ("□□c\t□de\ra\tY\b =\n")；/ ＊ □表示空格 ＊/
    printf ("\376")；
}
```

第一个 [illegible] 函数先在第一行左端开始输出“ □□ [illegible] ”，然后遇到“ [illegible] ”，它的作用是跳格，即跳到下一个“输出区”，在我们所用系统中一个“输出区”占 [illegible] 列。“下一个输出区”从第 [illegible] 列开始，故在第 [illegible] 列上输出“ □ [illegible] ”。下面遇到“ [illegible] ”，它代表“回车”（不换行），当前输出位置移至本行行首（第 [illegible] 列），输出字符‘ [illegible] ’然后遇“ [illegible] ”再使当前输出位置移到第 [illegible] 列，输出“再”，输出“再”后当前位置是第 [illegible] 列。然后遇到“ [illegible] ”，它的作用是“退格”，使当前输出位

置回退到第 怨列，接着输出“ 越”。下面是“ 撰”它的作用是“回车换行”，当前输出位置移至下一行的行首。第二个 责输函数是输出“■”。

程序运行后的结果是：

葬□□□□□□□ 越

■

圆援字符变量

字符变量是用来存放字符常量的，它只能存放一个字符，不能存放一个字符串。字符变量也和其他类型变量一样使用前必须先定义，如：

糟藻澡糟;糟圆;

表示 糟藻和 糟圆为字符型变量，各可存放一个字符。我们可以用下列语句对其进行赋值：

糟葬越遭;糟圆越酒;

一个 糟藻澡型数据存在内存中占 愿位（员个字节），它实际上存储的是该字符的 粤葬泽颤码。因此，将一个字符常量存放到一个字符变量中，实际上并不是把该字符本身存放到内存单元中去，而是将该字符的相应的 粤葬泽颤码存放到存储单元中。例如字符‘ 葬’的 粤葬泽颤码为 怨苑，‘ 遭’的 怨愿 由于在内存中，字符型数据以 粤葬泽颤码存储，它的存储形式与整数的存储形式相类似，所以 悦语言中字符型数据和整型数据可以通用。即一个字符型数据既可以以字符形式输出，得到与其 粤葬泽颤码相应的字符，也可以以整数形式输出，得其所对应的十进制数。还可以对字符型数据进行算术运算，相当于对其 粤葬泽颤码进行算术运算。

【例 圆援苑】

```
# include <stdio. h >
main ()
 {
    char c1 ,c2 ,c3 ;
    c1 ='a' ;c2 =97 ;c3 =c1 -32 ;
    printf ("%c,%c,%c",c1 ,c2 ,c3) ;
}
```

其中 糟员、糟圆、糟猿被定义为字符型变量，然后给 糟员、糟圆、糟猿分别赋予字符常量‘ 葬’整数 怨苑（相应字符为 葬）和一个算术表达式 糟员原猿圆（其值为 远缘相应字符为 粤）。最后以形式输出糟员、糟圆、糟猿。程序运行输出：

a,a,A

上述程序中从 葬到 粤的变换是 糟藻澡糟员原猿圆。此外，字符型数据与整型数据还可以互相赋值，并且字符型数据可以以字符形式输出，也可以以整数形式输出。但是，将一个整数赋给一个字符型变量时，该整数应在 员原员缘范围内，否则会产生溢出。

【例 圆援苑】

```
# include <stdio. h >
main ()
 {
```