项目1 初识C语言

上机指导1　C语言运行环境和运行C程序的方法

上机目标

(1)熟悉 Visual C++ 2010 学习版开发环境的使用方法。

(2)熟悉和掌握利用 Visual C++ 2010 学习版开发环境编辑、编译和运行一个简单的 C 程序的基本过程。

(3)掌握一个 C 程序的基本构成要素。

(4)掌握程序的基本调试方法。

上机练习

启动 Visual C++ 2010 学习版开发环境,编写如下程序代码并运行:

```
#include "stdio.h"
main()
{
  printf("My first C program.");
}
```

具体实现步骤如下:

步骤 1:打开 Visual C++ 2010 学习版开发环境,执行“文件”→“新建”→“项目”命令,弹出“新建项目”对话框,在“新建项目”对话框中选择“Visual C++”

→“空项目”选项，在下面的“名称”文本框中输入项目名称，如“text_01”，选择项目路径，单击“确定”按钮，进入新建项目界面，如图 1-1 所示。

图 1-1 “新建项目”对话框

步骤 2：在新建项目界面中的右侧找到新建的项目“text_01”。右击“text_01”下的“源文件”，在弹出的快捷菜单中执行“添加”→“新建项”命令，弹出“添加新项-text_01”对话框，在中间界面中选择“C + + 文件”选项，在下面的“名称”文本框中输入文件名，如“1. c”，并选择文件的路径。单击“添加”按钮，进入代码编辑窗口，如图 1-2 所示。

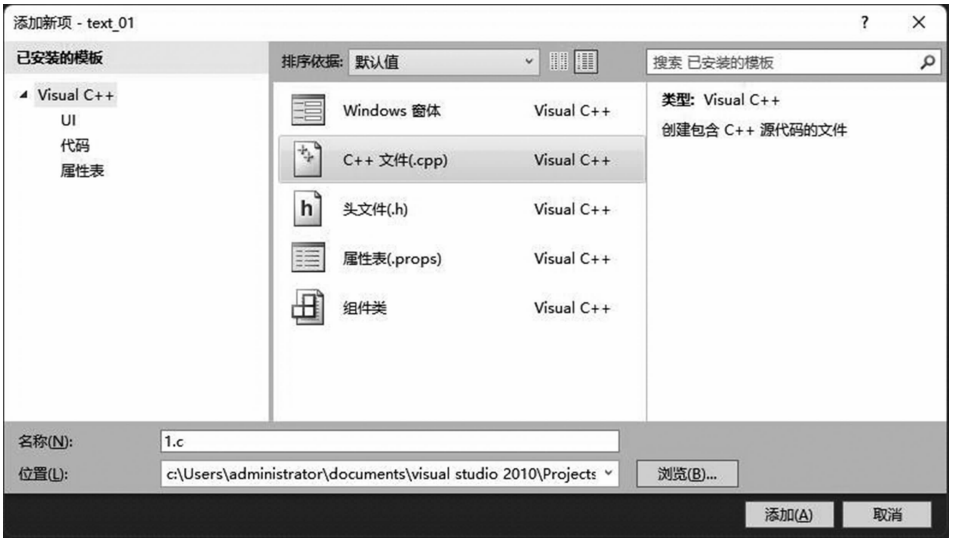

图 1-2 “添加新项-text_01”对话框

步骤 3:在代码编辑窗口中输入如下代码:

```
#include "stdio.h"
  main()
{
  printf("My first C program.");
}
```

步骤 4:将 C 源程序输入结束后,按 Ctrl+F5 快捷键,弹出提示对话框,单击“是”按钮,若程序没有错误,则显示图 1-3 所示的结果;若程序有错误,则在代码编辑窗口下边的输出窗口中显示错误提示,在代码编辑窗口中根据错误提示修改源代码,按 Ctrl+F5 组合键重新生成运行。

图 1-3 程序运行结果

上机指导2　在Visual C++ 2010下调试C语言程序

上机目标

(1)熟悉Visual C++ 2010在Windows系统下的调试方法。

(2)熟悉使用光标或断点在Visual C++ 2010下的调试方法。

(3)使用Shift+F11快捷键调试代码.

上机练习

按照上机指导1的步骤在Visual C++ 2010学习版开发环境中编写如下程序代码。

```
#include "stdio.h"
main()
{
  int n,s;
  n=1;
  s=0;
  while(n<=100)
  {
      s+=n;
      n++;
  }
  printf("the result is %d\n",s);
  return 0;
}
```

调试上述代码的方式如下所示。

1.方法一

设置光标或断点,使程序执行到中途暂停,以便观察阶段性结果。

(1)使程序到光标所在行暂停。

①在代码“printf("the result is %d\n",s);”行右击,在弹出的快捷菜单中执行“断点”→“插入断点”命令。

②执行“调试(Debug)”→“启动调试(Start Debugging)”命令,或按 F5 键,程序将执行到光标所在行暂停,如图 1-4 所示。如果把光标移动到后面某个位置再按 F5 键,程序将从当前的暂停位置继续执行到新的光标位置第二次暂停。

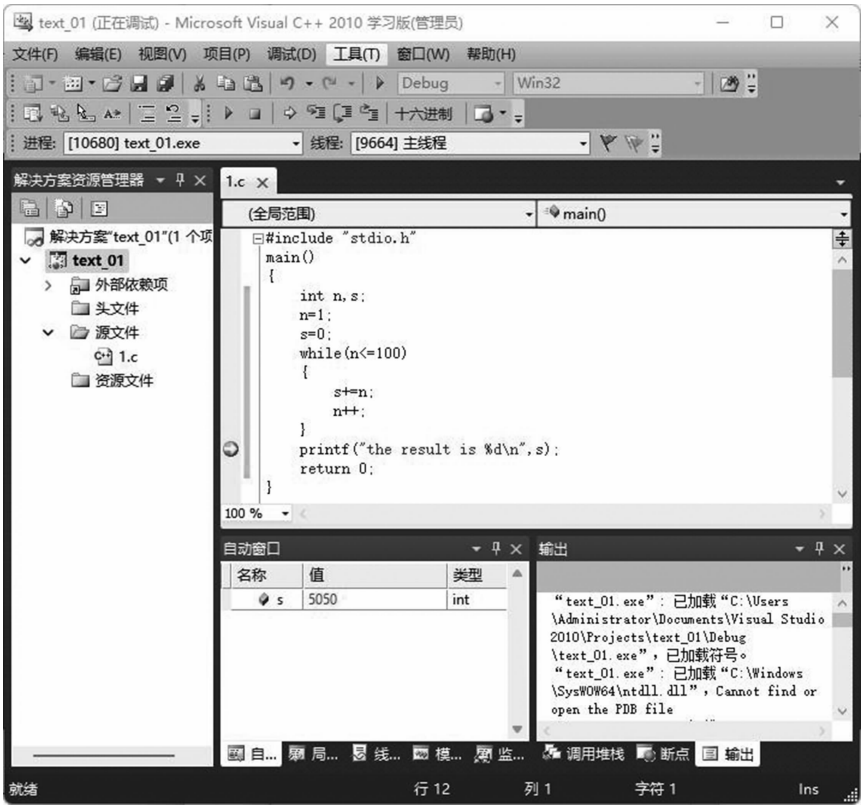

图 1-4　程序运行到光标处暂停

(2)在需要暂停的行上设置断点,断点通常用于调试较长的程序。

①执行“调试(Debug)”→“启动调试(Start Debugging)”命令,先对程序进行编译链接。

②将光标定位在断点所在行,按 F9 键,该操作是一个开关,按一次是设置断点,被设置了断点的行前面会有一个红色圆点标识,如图 1-5 所示,再按一次是取消设置断点,红色圆点标识消失。

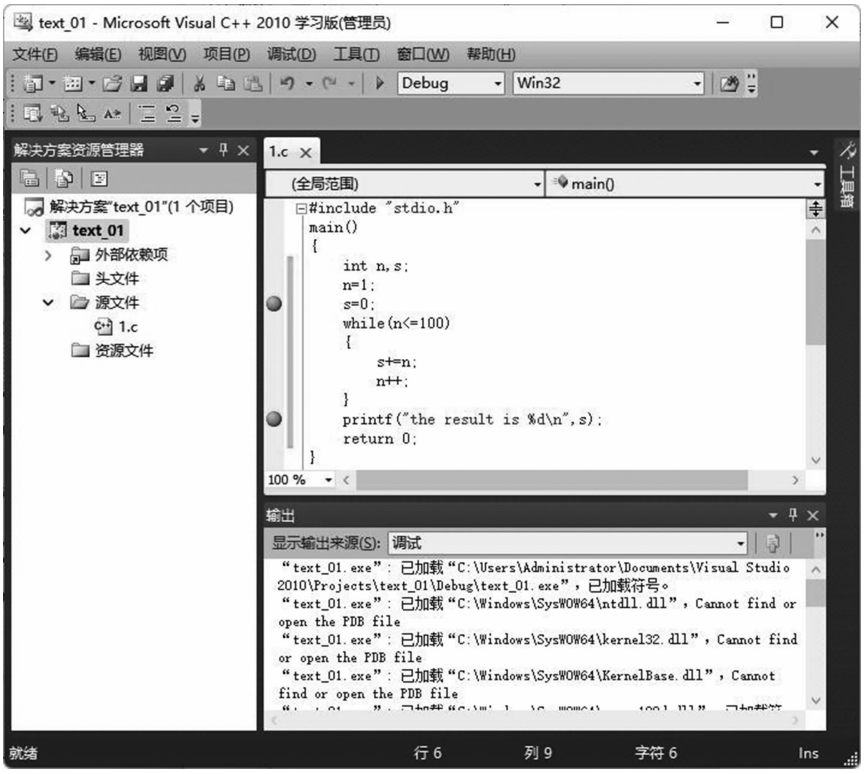

图 1-5　设 置 断 点

执行"调试(Debug)"→"删除所有断点(Toggle Breakpoint)"命令,或按 Ctrl+Shift+F9 快捷键,在弹出的"是否删除所有断点"对话框中单击"是"按钮,可以取消程序中的所有断点。

2. 方法二

当程序执行到某个位置时发现结果已经不正确了,说明在此之前肯定有错误码存在,如果能确定一小段程序可能有错,先按上面步骤暂停在该小段程序的头一行,再输入若干个查看变量,然后单步执行,即一次执行一行语句,逐行进行检查,查看到底是哪一行语句造成结果出现错误,从而确定错误的语句并予以纠正。单步执行的快捷键是 F11,如果要结束函数的单步执行,可按 Shift+F11 快捷键。

上机指导3　算法和算法的表示

上机目标

(1)理解算法的基本概念。

(2)掌握算法的5个基本特点,并能设计简单问题的算法描述。

(3)掌握程序流程图的画法。

上机练习

【例1-1】 有黑色和蓝色两个墨水瓶,但错把黑墨水装进了蓝色的墨水瓶中,而把蓝墨水装进了黑色的墨水瓶中,要求将其互换,请给出具体的算法描述。

算法分析:这是一个非数值运算问题。由于两个瓶子的墨水不能直接交换,解决这个问题的关键是引入第三个墨水瓶。设第三个墨水瓶为白色,其交换步骤如下:

(1)将黑瓶中的蓝墨水装进白瓶中。

(2)将蓝瓶中的黑墨水装进黑瓶中。

(3)将白瓶中的蓝墨水装进蓝瓶中。

(4)交换结束。

【例1-2】 计算函数M(x)的值,函数M(x)为

$$M(x)=\begin{cases} bx+a^2 & x\leqslant a \\ a(c-x)+c^2 & x>a \end{cases}$$

其中,a、b、c均为常数。

算法分析:本题是一个数值运算问题。其中,M代表要计算的函数值,有两个不同的表达式,根据x的取值决定采用哪一种计算方法。根据计算机具有逻辑判断的功能,用计算机解题的算法如下:

(1)将a、b、c和x的值输入计算机中。

(2)判断x和a的大小关系,如果x不大于a,执行步骤(3),否则执行步骤(4)。

(3)按表达式 $bx+a^2$ 计算 M(x)的值,然后执行步骤(5)。

(4)按表达式 $a(c-x)+c^2$ 计算 M(x)的值,然后执行步骤(5)。

(5)输出 M(x)的值。

(6)算法结束。

【例 1-3】 给定两个正整数 m 和 n(m≥n),求它们的最大公约数。

算法分析:求最大公约数的问题一般用辗转相除法(也称欧几里得算法)求解。例如,设 m=35,n=15,余数用 r 表示,它们的最大公约数的求解过程如下:35/15 商为 2,余数为 5,以 n 代替 m,以 r 代替 n,继续相除;15/5 商为 3,余数为 0,余数为 0 时,所得的 n 即两数的最大公约数,即 35 和 15 的最大公约数为 5。

用这种方法求最大公约数的算法描述如下:

(1)将两个正整数分别存放到变量 m 和 n 中。

(2)求余数:计算 m 除以 n,将所得的余数存放到变量 r 中。

(3)判断余数是否为 0:若余数为 0,则执行步骤(5),否则执行步骤(4)。

(4)更改被除数和除数:将 n 的值存放到 m 中,将 r 的值存放到 n 中,转向步骤(2)。

(5)输出 n 的当前值,算法结束。

如此循环,直到得到结果。

【例 1-4】 鸡翁一,值钱五;鸡母一,值钱三;鸡雏三,值钱一。百钱买百鸡,翁、母、雏各几何?

算法分析:这是一个典型的枚举问题。如果用 x、y、z 分别代表公鸡、母鸡和小鸡的数量,根据题意列方程组:

$$\begin{cases} x+y+z=100 \\ 5x+3y=z/3=100 \end{cases}$$

据题意可知,x、y、z 的范围一定是 0 到 100 的正整数,那么,最简单的解题方法为:假设一组 x、y、z 的值,直接代入方程求解,即在各个变量的取值范围内不断变化 x、y、z 的值,穷举 x、y、z 全部可能的组合,若满足方程组,则是一组解,这样即可得到问题的全部解。

可见利用枚举法解题的主要步骤如下:

(1)分析问题,确定答案的大致范围。

(2)确定列举方法。常用的列举方法有顺序列举、排列列举和组合列举。

(3)做试验,直到遍历所有情况。

(4)试验完后可能找到与题目要求完全一致的一组或多组答案,也可能没找到答案,即证明题目无答案。

注意:枚举法的特点是算法简单、容易理解,但运算量较大。对于可确定取值范围但又找不到其他更好的算法的问题,可以采用枚举法。通常枚举法用来解决“有几种组合”“是否存在”“不定方程”等类型的问题。

【例 1-5】 计算 s=1+2+3+……+100。

算法分析:

首先确定 s 的初始值为 0。

其次确定迭代公式 s+i→s。

当 i 分别取值 1、2、3、……、100 时,重复计算迭代公式 s+i→s,迭代 100 次后,即可求出 s 的值。其中,i 的取值是一个有序数列,所以可由计数器产生,即使 i 的初始值为 1,然后每迭代一次 i 的值加 1,图 1-6 所示为迭代法求和流程图。

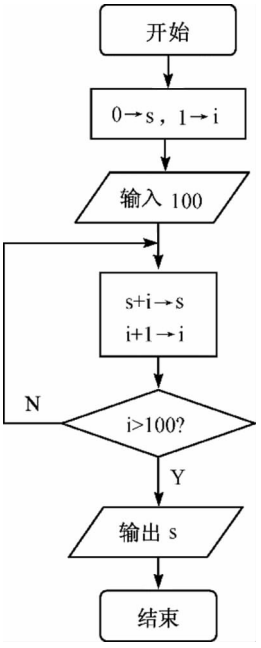

图 1-6 迭代法求和流程图

【例 1-6】 求 n!,设 n=15。

算法分析:求解 n!,其算法存在如下递推过程:

f(0)=0!=1

f(1)＝1!＝1×0!＝1

f(2)＝2!＝2×1!＝2

f(3)＝3!＝3×2!＝6

……

f(n)＝n!＝n×(n－1)!＝n×f(n－1)

要计算 15!,可以从递推初始条件 f(0)＝1 出发,应用递推通项公式f(n)＝n×f(n－1)逐步求出 f(1)、f(2)、f(3)、……、f(14),即由简到繁逐次迭代,直到求出 f(15)的值。

对算法的说明如下:递推法的关键是找到通项公式,求一个数的阶乘是简单的问题,该通项公式 f(n)＝ n×f(n－1)可以直接看出来。但事实上,有些问题要找到通项公式是相当困难的,并且即使找到了,计算也不一定简便,所以在设计算法时应从多种角度出发,找出最简便的解法。

习题

一、选择题

1. 下列叙述中不正确的是(　　)。

A. 程序就是软件,但软件不仅仅是程序

B. 程序是指令的集合,计算机语言是编写程序的工具

C. 计算机语言都是形式化语言,它有严格的语法规则和定义

D. 计算机语言只能编写程序而不能编写算法

2. 下列叙述中正确的是(　　)。

A. 结构化程序设计方法是面向过程程序设计的主流

B. 算法就是计算方法

C. 一个正确的程序就是指程序书写正确

D. 计算机语言是编写程序的工具而不是表示算法的工具

3. 在 C 程序中,下列叙述不正确的是(　　)。

A. main 函数可以位于程序的任意位置

B. 书写 C 程序时可以一行多句,也可以一句多行

C. C 语言本身没有输入/输出语句

D. 编译 C 程序时，可以发现注释行中的错误

4. 下列叙述中错误的是(　　)。

A. C 语言源程序由主函数组成

B. C 语言源程序由自定义函数组成

C. C 语言源程序由函数和过程组成

D. C 语言源程序中的注释是以“/*”开始，以“*/”结束

二、填空题

1. 在流程图符号中，判断框中应该填写的是_____。

2. 算法的_____特性是指一个算法必须在执行有限步骤后终止。

3. 在程序设计中，把解决问题确定的方法和有限步骤称为_____。

4. 组成 C 语言源程序的基本单位是_____，一个 C 语言源程序中有_____个主函数，还可包含若干个_____函数。

三、应用题

1. 用任意一种熟悉的方法描述求 n 个数中最小数的算法。

2. 已知 f(1)＝f(0)＝1，用两种算法计算斐波那契数列 f(n＋2)＝ f(n＋1)＋f(n)的前 30 项，并用流程图表示。

3. 对输入的任意 3 个数 a、b、c，要求按从小到大的顺序将它们打印出来，用流程图表示该算法。

4. 判断一个整数 n 能否同时被 3 和 7 整除，用流程图表示该算法。

5. 求全班某课程的平均分，用流程图表示该算法。

项目2 数据类型、常量和变量

上机指导1　C语言运行环境和运行C程序的方法

上机目标

(1)掌握不同类型常量的表示方法。

(2)掌握C语言数据类型的概念,熟悉如何定义一个整型、字符型、实型变量,以及如何对它们进行赋值。

(3)掌握整型、字符型、实型变量数据输出时所用的格式转换符。

上机练习

【例2-1】 整型常量的不同进制表示方法。

参考程序如下:

```
#include <stdio.h>
main()
{
  printf("十六进制整数30代表十进制整数%d\n",0x30);
  printf("八进制整数40代表十进制整数%d\n",040);
  printf("十进制整数50代表十进制整数%d\n",50);
}
```

程序运行结果如图2-1所示。

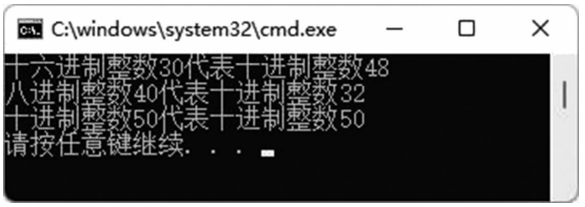

图 2-1 【例 2-1】程序运行结果

【例 2-2】 实型常量的两种表示方法。

参考程序如下：

```
#include <stdio.h>
main()
{
  printf("小数表示形式:%6f\n",3.141593);      /*实型常量的小数表示形式*/
  printf("指数表示形式:%6f\n",0.3141593e1);    /*实型常量的指数表示形式*/
  printf("指数表示形式:%6f\n",31.41593e1);     /*实型常量的另一种指数表示形式*/
}
```

程序运行结果如图 2-2 所示。

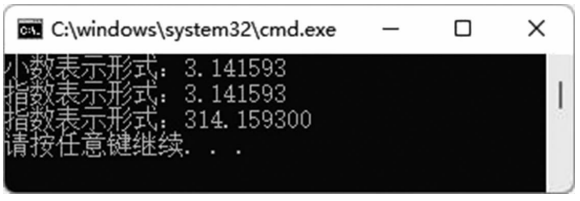

图 2-2 【例 2-2】程序运行结果

【例 2-3】 字符型常量的应用。

参考程序如下：

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

```
    char c1,c2;
    c1='e';
    c2='f';
    printf("字符变量 c1 所代表的字符是:%c\n 字符变量 c1 的 ASCII 码是:%d\n",
    c1,c1);
    printf("字符变量 c2 所代表的大写字符是:%c\n",c2-32);
    printf("字符变量 c2 的大写字符的 ASCII 码是:%d\n",c2-32);
}
```

程序运行结果如图 2-3 所示。

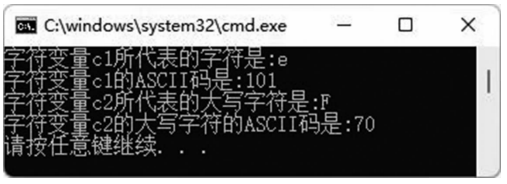

图 2-3 【例 2-3】程序运行结果

程序中先定义变量 c1 和 c2 为字符型变量。通过赋值语句给 c1 和 c2 变量分别赋予字符常量 'e' 和 'f'。第一个 printf()语句用于输出字符型变量 c1 的字符(%c);其后用控制字符(%d)输出 c1 变量对应的十进制整型数值——ASCII 码值。第二个 printf()语句用 c2-32 输出 c2 字符变量所对应的大写字符 F 及其 ASCII 码值。由上可知,字符常量可以像数值常量一样,在程序中进行相应的各种运算。

【例 2-4】 转义字符的应用。

参考程序如下:

```
#include <stdio.h>
void main()
{
    int x,y,z;
    x=7; y=8; z=9;
    printf("%d\n\t%d%d\n%d%d\t%d\n",x,y,z,z,y,x);
}
```

程序运行结果如图 2-4 所示。

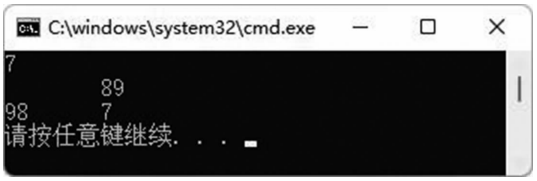

图 2-4 【例 2-4】程序运行结果

【例 2-4】中第一行第一列输出 x 的值 7；接着是转义字符 '\\n'，则执行回车换行操作，移到第二行；接着是转义字符 '\\t'，于是移到下一制表位置，再输出 y 的值 8；接着输出 z 的值 9；再执行回车换行 '\\n'，移到第三行；又输出 z 的值 9 和 y 的值 8；接着跳到下一制表位置 '\\t'，与上一行的 9 对齐，接着是转义字符 '\\b'，则退回一格，输出 x 的值 7。

【例 2-5】 字符串常量的应用。

参考程序如下：

```
#include <stdio.h>
void main()
{
  char *c;
  c="He is my boss.";                    /*为c赋值*/
  printf("%s\n",c);
  printf("%s\n","He is my boss.");       /*输出字符串常量*/
}
```

程序运行结果如图 2-5 所示。

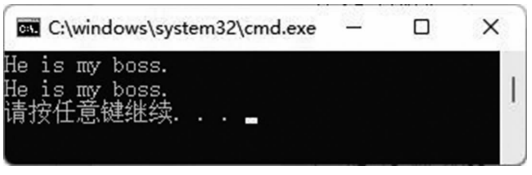

图 2-5 【例 2-5】程序运行结果

注意：本程序中的“char *c”不同于字符变量定义“char c”，可以把一个字符常量赋予一个字符变量，但不能把一个字符串常量"He is my boss."赋予一个字符变量。在 C 语言中没有相应的字符串变量，必须用一个字符数组来存放一个字符串常量。

【例 2-6】 符号常量的使用。

参考程序如下：

```
#include "stdio.h"
#define NUM 35
int main()
{
  float p,total;
  scanf("%f",&p);
  total=p*NUM;
  printf("%f\n",total);
  return 0;
}
```

程序运行结果如图 2-6 所示。

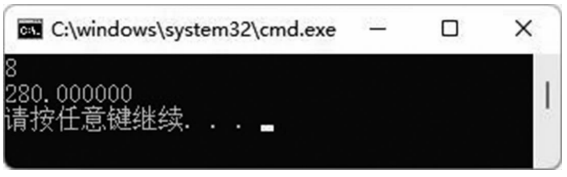

图 2-6 【例 2-6】程序运行结果

程序中用#define 命令行定义 NUM 代表常量 35。此后凡在本程序中出现的 NUM 都代表 35，可以和常量一样使用。

【例 2-7】 求一个圆柱体的体积，用符号常量代替 π。

参考程序如下：

```
#include "stdio.h"
#define PI 3.14159
main()
{
  float r,h,v;
  printf("输入圆柱体的底面半径和高:");
  scanf("%f,%f",&r,&h);
  v=PI*r*r*h;
```

```
    printf("圆柱体的体积是%f\n",v);
}
```

程序运行结果如图 2-7 所示。

图 2-7 【例 2-7】程序运行结果

上机指导 2　变量的定义和初始化

上机目标

(1)理解变量的含义。

(2)掌握变量定义和初始化的方法。

(3)能够根据题目要求正确使用变量。

上机练习

【例 2-8】 整型数据的溢出。

参考程序如下：

```
#include <stdio.h>
void main()
{
   int a,b;
   a=2147483647;
   b=a+1;
   printf("%d,%d\n",a,b);
}
```

程序运行结果如图 2-8 所示。

图 2-8 【例 2-8】程序运行结果

输出 a 为 2 147 483 647，而 b 为－2 147 483 647。这是因为一个整型变量只能容纳－2 147 483 647～2 147 483 647 范围内的数，无法表示大于 2 147 483 647 或小于－2 147 483 647 的数。遇此情况就发生溢出。

注意：在 C 语言程序中，当进行数据运算时，要考虑数据溢出问题。为了防止数据溢出，需要定义合适的变量类型，如可将【例 2-8】中的数据 a、b 定义为长整型。

【例 2-9】 整型变量的定义与使用。

参考程序如下：

```
#include <stdio.h>
int main()
{
  int a,b,c,d;
  unsigned u;
  a=12;
  b=-36;
  u=10;
  c=a+u;
  d=b+u;
  printf("a+u= %d,b+u= %d\n",c,d);
  return 0;
}
```

程序运行结果如图 2-9 所示。

图 2-9 【例 2-9】程序运行结果

注意：不同种类的整型数据可以进行算术运算。

【例 2-10】 变量的初始化与使用。

参考程序如下：

```
#include <stdio.h>
int main()
{
  int a,b,c=10;
  char ch1='A',ch2='a';
  printf("%d\n",c);
  printf("%c,%c\n",ch1,ch2);
  return 0;
}
```

程序运行结果如图 2-10 所示。

图 2-10 【例 2-10】程序运行结果

本程序中，“int a,b,c=10;”表示定义了 3 个整型变量，但只给变量 c 赋初值 10。如果要同时给 3 个变量赋值，可以采用如下形式：

```
int a,b,c;
a=b=c=10;
```

【例 2-11】 求 80 的三次方。

参考程序如下：

```
#include <stdio.h>
void main()
{
  short int x;
  x=80*80*80;
```

```
    printf("80 的三次方是 %d\n",x);
}
```

程序运行结果如图 2-11 所示。

图 2-11 【例 2-11】程序运行结果 1

这个结果显然是不正确的,问题出在哪里? 如果将程序改为

```
#include <stdio.h>
void main()
{
  long int x;
  x=80 * 80 * 80;
  printf("80 的三次方是 %d\n",x);
}
```

那么程序运行结果如图 2-12 所示。

图 2-12 【例 2-11】程序运行结果 2

80 * 80 * 80 的结果 512 000 超过了整型变量的表示范围,因此用短整型变量 x 来保存数值 512 000 产生了错误的输出结果。将变量 x 定义为长整型即可解决这一问题。可见在定义变量时要充分考虑其取值范围,以免发生错误。

【例 2-12】 实型变量的声明。

参考程序如下:

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

```
{
  float f1,f2;          /*声明变量 f1、f2 为单精度实型变量*/
  double d1,d2;         /*声明变量 d1、d2 为双精度实型变量*/
}
```

【例 2-13】 实型变量的使用。

参考程序如下：

```
#include <stdio.h>
void main()
{
  float f;
  double d;
  f=77777.77777;                /*为单精度实型变量赋值*/
  d=77777.77777777777777;       /*为双精度实型变量赋值*/
  printf("f=%f\nd=%f\n",f,d);
}
```

程序运行结果如图 2-13 所示。

图 2-13 【例 2-13】程序运行结果

注意：由于 f 是单精度型，有效位数有 7 位，可保留 6 位小数。变量 f 的数值的整数已占 5 位，故小数两位后均为无效数字。d 是双精度型，有效位为 16 位，可保留 14 位小数。但系统规定小数后最多保留 6 位，故其余部分四舍五入。

【例 2-14】 字符变量的使用。

参考程序如下：

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

```
    char c1,c2;
    c1=101;c2=104;          /*为字符变量赋值*/
    printf("%c, %c\n",c1,c2);
    printf("%d, %d\n",c1,c2);
}
```

程序运行结果如图 2-14 所示。

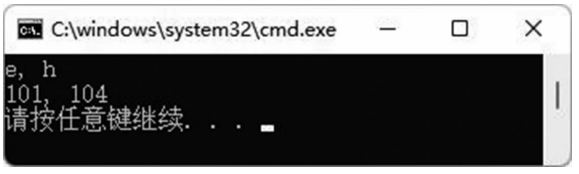

图 2-14 【例 2-14】程序运行结果

程序中 c1、c2 被声明为字符型变量，但在下一行却将整数 101、104 分别赋给 c1、c2。它们的作用就相当于下面两个赋值语句：

```
c1='e';c2='h';
```

因为 'e' 和 'h' 的 ASCII 码分别为 101 和 104，函数体中第三行输出两个字符，“%c”为输出字符格式，所以程序输出：

```
e,h
```

字符型数据也可以用整数形式输出，如第四行，“%d”为输出整数格式，所以输出：

```
101,104
```

【例 2-15】 将小写字母转换为大写字母。

参考程序如下：

```
#include <stdio.h>
void main()
{
  char c1,c2;
  c1='m';
  c2='n';
```

```
    printf("转换前的小写字母是%c,%c\n",c1,c2);
    c1=c1-32;              /*将c1转换成对应的大写字母*/
    c2=c2-32;              /*将c2转换成对应的大写字母*/
    printf("转换后的大写字母是%c,%c\n",c1,c2);
}
```

程序运行结果如图 2-15 所示。

图 2-15 【例 2-15】程序运行结果

上机指导 3　各种数据类型的直接转换

上机目标

(1)理解自动类型转换和强制类型转换的含义。

(2)能够编程实现数据类型之间的正确转换。

上机练习

【例 2-16】 强制类型转换运算符的使用。

参考程序如下：

```
#include <stdio.h>
void main()
{
   float x=7.6,y=3.2;
   printf("x+y=%f\n",x+y);
   printf("强制类型转换后\n");
```

```
    printf("x+y= %d\n",(int)x+(int)y); /* 将 x 和 y 强制转换为整型后再相加 */
}
```

程序运行结果如图 2-16 所示。

图 2-16 【例 2-16】程序运行结果

第 5 行代码:以浮点格式输出表达式 x+y 的值,结果为 10.800000(C 语言默认浮点格式输出时,小数部分占 6 位)。

第 7 行代码:将 x 和 y 的类型强制转换为整型后再相加,并以整型格式输出,结果为 10。在将实数类型强制转换为整型时,采用去除小数部分的办法。

【例 2-17】 数据类型的转换。

参考程序如下:

```
#include <stdio.h>
int main()
{
   int i;
   float x=69.87;
   float y;
   i=(int)x;                    /* 数据类型的显式转换 */
   printf("i= %d\n",i);
   printf("x= %f\n",x);
   y=56;                        /* 数据类型的隐式转换 */
   printf("y= %f\n",y);
   return 0;
}
```

程序运行结果如图 2-17 所示。

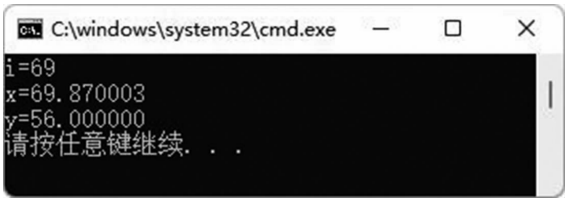

图 2-17 【例 2-17】程序运行结果

习题

一、选择题

1.（　　）是程序执行过程中其值不发生变化的数据。

A. 常量　　B. 变量　　C. 数值　　D. 字母

2. 在 C 语言中，字符常量是用（　　）括起来的单个字符。

A. 圆括号　　B. 方括号　　C. 单引号　　D. 双引号

二、填空题

1. 在 C 语言中，数据类型可分为_____、_____、_____、_____四大类。

2. 转义字符以_____开头，后跟一个或几个字符。

三、简答题

1. C 语言规定对所用到的变量要“先定义后使用”，这样做有什么好处？

2. 字符常量与字符串常量有什么区别？

四、应用题

1. 请将下面各数用八进制数和十六进制数（补码）表示。

（1）12。

（2）－1。

（3）65。

（4）－10。

2. 写出以下程序的运行结果。

（1）程序一。

```
#include <stdio.h>
void main()
{
  char c1=97,c2=98;
  int a=97,b=98;
  printf("%3c,%3c\n",c1,c2);
  printf("%d,%d\n",c1,c2);
  printf("\n%c %c\n",a,b);
}
```

(2)程序二。

```
#include <stdio.h>
void main()
{
  int i,j;
  i=3; j=4;
  printf("%d%d\n",i++,++j);
  printf("%d,%d\n",i,j);
  printf("%d,%d\n",-i++,-++j);
}
```

(3)程序三。

```
#include <stdio.h>
int main()
{
  int x=1,y=2;
  printf("%d+%d=%d\n",x,y,x+y);
  printf("10 Squared is: %d\n",10*10);
  return 0;
}
```

(4)程序四。

```
#include <stdio.h>
int main()
{
  char c='A';
   printf("c:dec=%d,oct=%o,hex=%x,ASCII=%c\n",c,c,c,c);
   return 0;
 }
```

3.编写一个C程序,要求输出本学期的课程表。

参考格式如下:

	星期一	星期二	星期三	星期四	星期五
1—2	英语	高数	C语言	英语	C语言
3—4	上机	英语	高数	美术	高数
5—6	体育	美术	音乐	自习	自习