**实验八 进制转换**

**一、实验目的和要求**

输入两个十进制正整数n和base(2<=base<=16), 将n转换为base进制后输出。要求定义并调用printInt(n, base),它的功能是输出n的base进制。

**二、实验内容**

1. 实验准备
   1. 理论知识介绍
      1. 数据的输入：

要从用户依次读入int类型的变量a,n，可以使用如下语句:

cin >> a>>n;

* + 1. 数据的输出

要将变量的内容显示在显示器上，可以使用cout和流插入运算符。Cout还可以输出表达式的执行结果，比如:

cout << a << ‘+’ << b << ‘=’ << a+b << endl;

* + 1. 算术表达式

C++中，算术运算里有+（加法），-（减法），/（除法），%（取模），但是没有取平方算数运算。

* + 1. 数据类型 （整型）：

C++中一个整型变量可以储存一个整数，可以直接通过cin和cout输入和输出。

* + 1. 关系表达式

关系表达式用于比较两个值的大小。C++提供了6个关系运算符: <, <=, >, >=,

==, !=。关系运算符左结合。

* + 1. for语句

**for (i = 0; i < n; ++i)**

**需要重复执行的语句**

* + 1. if 语句

**if （条件） 语句**

**if （条件） 语句1 else 语句2**

* + 1. 函数

编写一个实现某个功能的函数称为函数定义。一旦定义了一个函数，在程序中就可以反复调用这个函数。函数的定义形式如下：

**类型名 函数名 (形式参数表)**

**{**

**变量定义部分**

**语句部分**

**}**

* + 1. 递归函数

递归函数就是将规模较大的问题分解成同样形式的小问题的函数，递归函数通常通过调用自身来实现递归。

1. 实验项目
   1. 分析
      1. 实验要求定义printInt(n, base)函数来输出n的base进制，这个问题稍微有点复杂，我们可以将其分解，只要将n对base进行整数的除法，再利用printInt(n, base)函数对商进行商的base进制的输出，最后再将余数进行base进制的输出
      2. 在使用printInt(n, base)函数时要注意输出信息的位置，因为我们要将余数放在进制数的后面，所以输出的位置应该在进制位的后面。
      3. 由于存在超过10进制的数，所以要考虑将10,11,12,13,14,15分别转换为A,B,C,D,E,F。
   2. 方案
      1. 利用cin读取n,base的数值
      2. 利用cout, printf, printInt(n, base)来共同进行想要格式的输出
      3. 在printInt(n, base)函数中进行递归来输出n的base进制
   3. 源程序

#include<iostream>

using namespace std;

void printInt(int n, int base) {

int quotient = n / base;

int modulo = n % base;

if (n != 0) {

printInt(quotient, base);

printf("%X", modulo);

}

return;

}

int main() {

int i, n, base;

cout << "Please input n,base: ";

cin >> n >> base;

cout << n << "(10) = ";

printInt(n, base);

cout << "(" << base << ")" << endl;

return 0;

}

* 1. 测试数据
  2. 255 16
  3. 15 2
  4. 77 3
  5. 1234 13
  6. 运行结果

1. 255(10) = FF(16)
2. 15(10) = 1111(2)
3. 77(10) = 2212(3)
4. 1234(10) = 73C(13)
   1. 异常现象

最初的程序在输入 255 16 输出的结果为 255(10) = ff(10)

* 1. 分析

最初的程序使用的输出的方法是这样的：

cout << hex << modulo;

由于cout是一个流，所以hex这个参数会影响之后所有的内容，所以括号内的base本来为十进制的16也被转换为了16进制的10，而且这种方法也不能产生大写的F。

* 1. 总结

cout接受的是一个流, 所以前面的内容会对后面的内容产生影响，这时候我们就可以利用printf函数来避免这种影响 。

**三、实验小结**

1. 完成情况

将cout的输出用printf的输出方法替换之后得到正确结果。

1. 重点

递归函数函数的定义和调用。

1. 难点

如何将进制转换处理后的结果正确地输出并得到想要的格式的结果。

1. 解决方法

将cout << hex << modulo; 语句替换为printf("%X", modulo);。

1. 有待改进之处

printf用来输出并不是一个最佳的选择，很可能利用cout也能得到想要的结果

1. 收获、体会

合理地利用递归函数可以有效地简化问题，而且递归函数中的递归语句的位置要特别注意。C++中有许多不同的输出方法,比如cout 和printf,它们有各自的特点，我们可以利用这些特点来输出我们想要的格式。