进制转换工具

摘　　要

随着科技的进步，社会的发展，人们在日常生活中对进制转换的要求也相应提高。而市面上的进制转换工具经常存在收费高昂与运行不稳定等问题。针对生活中进制转换这一环节，传统的人工笔算已适应不了日益增长的转换需求带来的压力，开发新型、开源、免费的进制转换工具已是迫在眉睫。本次程序设计采用比较流行的10进制桥梁转换2、8、16进制的开发模式，在Visual Studio 2019开发工具下，使用C++作为开发语言来开发进制转换工具。该工具与传统的人工笔算相比避免了冗杂的数字运算，提高了工作效率。经过细致的测试，该进制转换工具运行良好，达到了设计所预期的目标。

关键词：进制转换；Visual Studio 2019;C++

**Number System Conversion Tool**

**ABSTRACT**

With the progress of science and technology, the development of society, the people’s conversion requirements are also increased accordingly in their daily lives. However, the system conversion tools on the internet market sometimes have some problems, such as high charges and unstable operation. Traditional manual computing can not adapt to the increasing demand of conversion, so it is urgent to develop a new, open source and free tool. The tool uses the more popular convertion way which makes 10 system a bridge to convert 2, 8, 16 systems in the Visual Studio 2019 development tools. C ++ is the language to develop the tool. Compared with the traditional manual calculation way, this tool avoids the redundant numeral operation and improves the work efficiency. Through careful test, the tool runs well and achieves the design goal.

**Key words**: Number System Conversion; Visual Studio 2019;C++

目 录

1 功能需求 …………………………………………………………1

1.1 功能任务 ………………………………………………………1

1.2 开发环境 ………………………………………………………1

1.3 界面设计 ………………………………………………………1

1.4 预期效果 ………………………………………………………2

2 功能需求 …………………………………………………………3

2.1 头文件、函数定义、全局变量设置 ……………………………3

2.2 主要操作算法描述 ………………………………………………4

3 程序实现 …………………………………………………………9

3.1 主函数 ……………………………………………………………9

3.2 主菜单 …………………………………………………………10

3.3 次级菜单 ………………………………………………………11

3.4 10进制中转 ……………………………………………………12

3.5 对应进制转换 …………………………………………………18

3.6 输出函数 ………………………………………………………20

4 程序测试 ………………………………………………………21

4.1正常情况 …………………………………………………………21

4.2异常情况 …………………………………………………………23

5 心得体会 ………………………………………………………23

5.1心得体会 …………………………………………………………23

参考文献 ……………………………………………………………25

1 功能需求

1.1 功能任务

利用C++程序设计语言实现二、八、十、十六进制数之间相互转换，设计相应算法。具体步骤包括：将各进制数字符串转换为十进制正数值；分离该十进制数值的整数和小数部分；对分离出的整数、小数部分分别做转换，将其转换为目标进制；将转换好的整数、小数部分以及符号位拼接为字符串输出。

1.2 开发环境

Visual Studio 2019，以C++为编程语言。

1.3 界面设计

进制转换工具是根据人们日常的转换需要而进行设计的，工具从功能上划分为三个主模块，需求确认模块，进制转换模块，信息输出模块。

具体应实现的功能如下：

* 需求确认模块

功能：根据用户需求的不同，分别跳转到不同的次级菜单，调用不同的函数。

* 进制转换模块

功能：通过进制转换流程的数学方法将用户需要的数据进行转换，并调用不同的信息输出模块。

* 信息输出模块

功能：输出转换结果，并引导用户是否继续转换。

进制转换工具的总工作流程如图1-1所示：

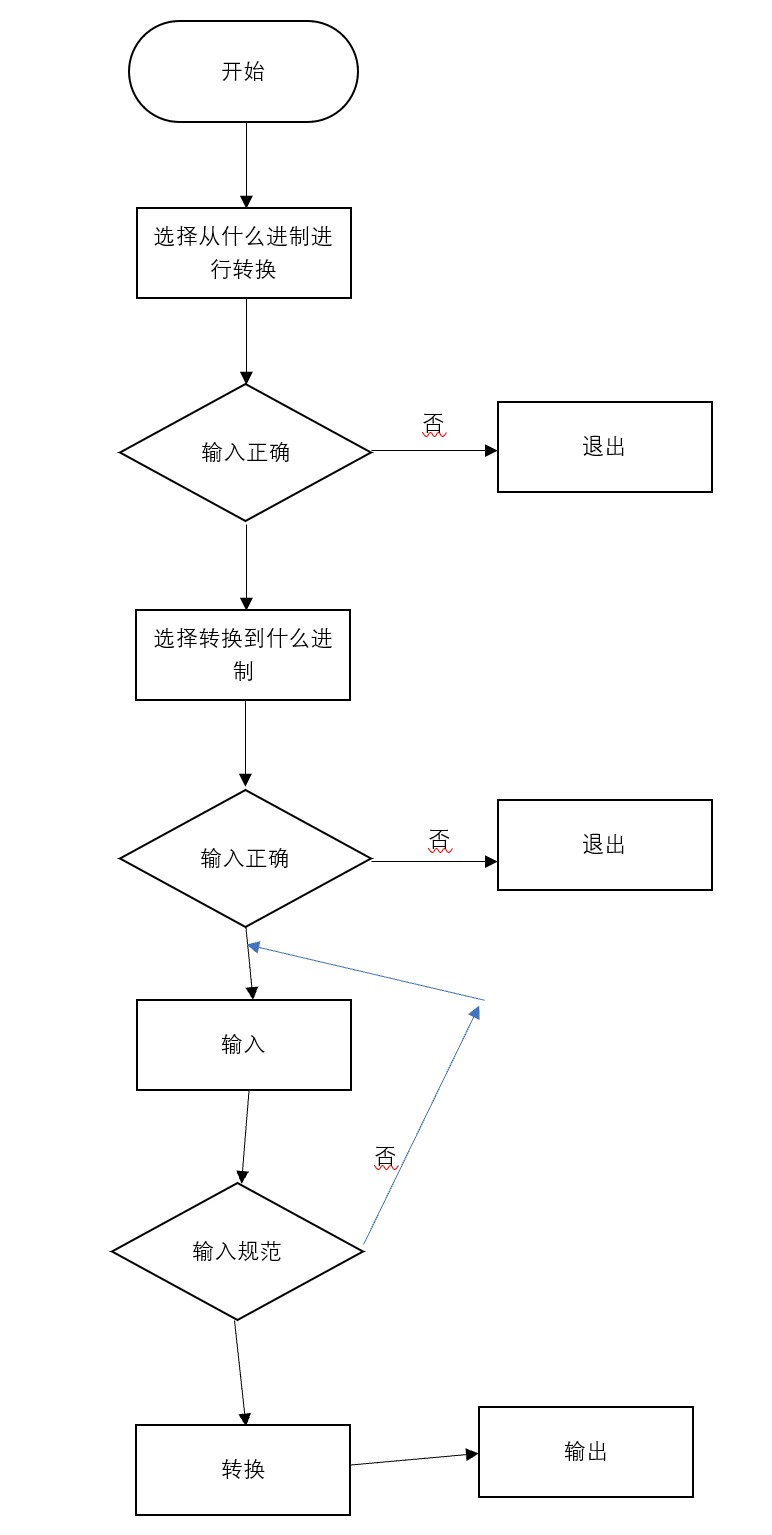


图1-1　　进制转换工具的总工作流程

1.4 预期效果

在灵活引导用户需求之后，检测用户输入的原进制数字符串合法性（包括小数点位置，‘0’的位置以及进制数码合法性），并对其是否为负数给出标记。能够兼顾界面美观与稳定性，并给予人性化提示与连续转换功能。

2 功能需求

2.1 头文件、函数定义、全局变量设置

#include<iostream>

#include<cmath>

#include<cstdio>

#include<stdlib.h>

#include<iomanip>

#include <stdio.h>

#include <tchar.h>

#include<string>

#include <sstream>

using namespace std;

#define MAX 100 设置“最大值”为100。

bool running = true; 设置持续转换标签为“真”

bool positive = true; 正负值判定为“正数”

int PR = 6; 设置精度为小数点后6位

菜单函数定义：

void firstmenu(); 主菜单

void secondmenu\_2(); 2进制转换二级菜单

void secondmenu\_8(); 8进制二级菜单

void secondmenu\_10(); 10进制二级菜单

void secondmenu\_16(); 16进制二级菜单

功能函数定义

void fun2(int object); 2进制转10进制

void fun8(int object); 8进制转10进制

void fun10(int object); 10进制转换

void fun16(int object); 16进制转10进制

;

void make2(double ten); 10进制转2进制

void make8(double ten); 10进制转8进制

void make16(double ten); 10进制转16进制

输出函数定义

void display(string out); 2、8、16进制数码输出

void fastout(double ten); 10进制数码输出

2.2 主要操作算法描述

2.2.1 需求确认模块

进制转换工具通过两级菜单确认用户需求。

主菜单界面如图2-2-1所示：

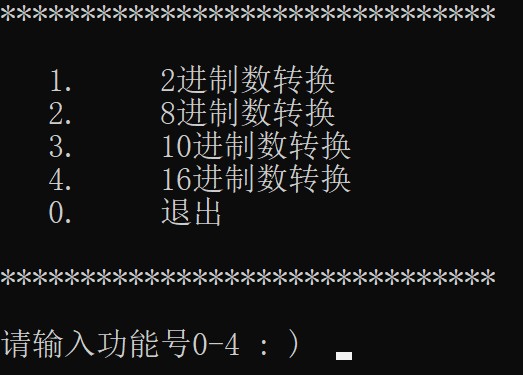


图2-2-1-1　　主菜单界面

通过用户输入的数字，对应选项的ID，进而找到对应的次级菜单，列出次级菜单的选项。

次级菜单界面如图2-2-2所示（以二进制转换到其他进制为例）：

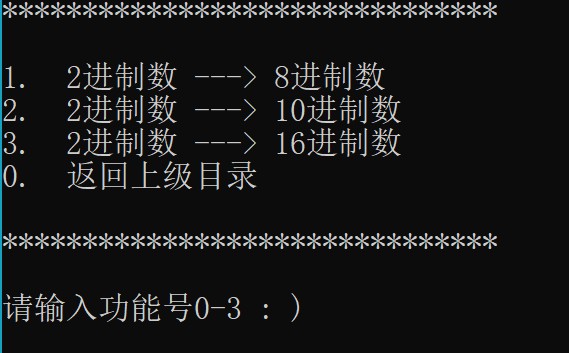


图2-2-1-2　　子菜单界面

通过用户输入的数字，对应选项的ID，进而找到对应的转换方式，调用对应转换函数开始转换。

主次菜单核心代码示例：

* 主菜单：

void firstmenu()

{

system("cls");

int button = 0;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n";

cout << " 1. 2进制数转换\n";

cout << " 2. 8进制数转换\n";

cout << " 3. 10进制数转换\n";

cout << " 4. 16进制数转换\n";

cout << " 0. 退出\n\n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n";

cout << "请输入功能号0-4 : ) ";

cin >> button;

if (button < 0 || button>4)

{

cout << "请输入功能号0-5 : (\n\n";

cin >> button;

system("cls");

}

else

{

switch (button)

{

case(1):

{

system("cls");

secondmenu\_2();

break;

}

……

}

}

}

* 次级菜单：

void secondmenu\_2()

{

int button = 0;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n";

cout << "1. 2进制数 ---> 8进制数\n";

cout << "2. 2进制数 ---> 10进制数\n";

cout << "3. 2进制数 ---> 16进制数\n";

cout << "0. 返回上级目录\n\n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n";

cout << "请输入功能号0-3 : ) ";

cin >> button;

if (button < 0 || button>3)

{

cout << "请输入功能号0-3 : (\n\n";

cin >> button;

system("cls");

}

else

{

switch (button)

{

case(1):

{

system("cls");

fun2(8);

break;

}

……

}

}

}

2.2.2 进制转换模块

（1）合法性验证、正负性判断

通过if条件句分别判断输入的字符串中‘0’，‘-’，‘.’的数量、进制数码和位置的合法性：

例如：if (original[0] == '-')

{

positive = false;

}就判断了正负。

又例如if (original[i] == '-' && i != 0)

{

ok = false;

}就判断了负号的位置合法性。

又例如if (original[i] != '0' && original[i] != '1' && original[i] != '.' && original[i] != '-')

{

ok = false;

}就判断了进制数码的合法性

若输入不合法，则提示用户重新输入。

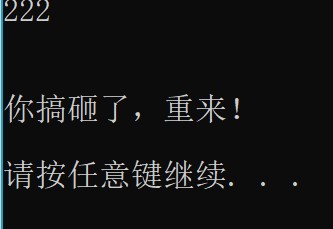


图2-2-2-1　　“重来”界面

全部合法性判断完毕后，布尔变量ok为真值，则进行下一步操作。

（2）小数存在与小数位置的判断

bool xiaoshu = false;

for (int o = 0; o < le; o++)

{

if (original[o] == '.')

{

xiaoshu = true;

sit = o;

}

}

判断完毕后，根据是否为小数的if语句进入不同分支进行字符到数值的转换。

（3）从字符到数值再到十进制（以有小数位的情况为例）

if (original[i] == '1')

{

upor[i] = 1;

}

当有小数位时，在小数点位置前使用“up-original”数值数组通过for循环遍历与if条件句判断的方法进行承接，在小数点后使用“down-original”数值数组通过for循环遍历与if条件句判断的方法进行承接。

ten += ((pow(2, upmi) \* upor[i]));

upmi--;

根具进制转换的要求，将每个位置的数值乘上对应进制的对应位置的幂次方，并将其累加到ten里，传递给下一个函数进行进制转换。

（4）从十进制到对应进制（以有小数位的情况为例）

for (i = 0;; i++)

{

left = temp % 2;

first[i] = char(left) + '0';

temp = temp / 2;

if (temp == 0)

{

break;

}

}

for (int j = 0; j <= i; j++)

{

end[j] = first[i - j];

}

end[i + 1] = '.';

for (j = i + 2;; j++)

{

b = b \* 2;

left = int(b);

end[j] = char(left) + '0';

b = b - left;

pr++;

if (b == 0)

{

break;

}

if (pr > PR)

{

break;

}

}

}

通过将上个函数传递过来的数值辗转除以对应进制数值，并倒序拼凑成为新的字符串。

通过PR作为小数点的精度控制（默认为6，即到小数位后6位停止转换）。

2.2.3 信息输出模块

（1）结果输出

分为两类，十进制直接输出与其他进制输出：

十进制：

if (positive == false)

{

cout << "-";

}

cout << std::to\_string(ten) << "\n是转换得到的结果 这里使用了std里的to\_string（）来将十进制数值按数字一个个转换成字符，并拼接成字符串输出。需要头文件： <sstream>

其他进制：

cout << out;

cout << "\n是转换得到的结果 因为其他进制的输出比十进制输出需要多一步，已经转换为字符串了，故直接输出即可。

（2）输出引导

输出完后进行询问，是否需要继续转换，若否，则另全局变量running为假，则停止主函数的调用。

cout<<"继续转换？ YES or NOT\n\n1 = yes, 2 = no.";

cin >> button;

if (button == 1)

{

cout << "我们的工作马上开始\n（为了运行的稳定，请一定使用英文输入法）\n\n\n";

system("pause");

system("cls");

return ;

}

else

{

running = false;

}

3 程序实现

3.1主函数

int main()

{

while (running) 当running为真，循环调用主菜单

{

firstmenu();

}

cout << "\n\n欢迎您下次使用!!\n"; 跳出循环代表程序结束，清屏

system("pause");

system("cls");

return 0;

}

3.2主菜单

void firstmenu()

{

system("cls"); 先清屏保证美观性

int button = 0; 对应每一项

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n";

cout << " 1. 2进制数转换\n";

cout << " 2. 8进制数转换\n";

cout << " 3. 10进制数转换\n";

cout << " 4. 16进制数转换\n";

cout << " 0. 退出\n\n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n";

cout << "请输入功能号0-4 : ) ";

cin >> button;

if (button < 0 || button>4) 输入错误容错检查

{

cout << "请输入功能号0-5 : (\n\n";

cin >> button;

system("cls");

}

else 根据button（按钮）数值选择功能

{

switch (button)

{

case(1):

{

system("cls");

secondmenu\_2(); 先清屏再调用对应进制子菜单

break;

}

…………以下重复故省略

}

}

}

3.3 次级菜单

以二进制转换为例：

void secondmenu\_2()

{

int button = 0;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n";

cout << "1. 2进制数 ---> 8进制数\n";

cout << "2. 2进制数 ---> 10进制数\n";

cout << "3. 2进制数 ---> 16进制数\n";

cout << "0. 返回上级目录\n\n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n";

cout << "请输入功能号0-3 : ) ";

cin >> button;

if (button < 0 || button>3)

{

cout << "请输入功能号0-3 : (\n\n";

cin >> button;

system("cls");

}

else

{

switch (button)

{

case(1):

{

system("cls");

fun2(8);

break;

}

…………

}

}

}

“fun2(8)”中的参数8代表着由二进制转换为8进制

3.4 10进制中转

以二进制转换为例：

前半段（容错部分）

void fun2(int object)

{

string original;

bool ok = false;

while (ok == false)

{

int i; 循环变量

int xiaoshu = 0; 小数点的个数

cin >> original; 输入需要转换的对应进制的字符串

int le;

le = original.length(); 测定长度

ok = true; 初始化正确标志为“正确”

for (i = 0; i < le; i++) 遍历

{

bool okpoint = false; 设置小数判定为“不是小数”

if (original[0] == '-')

{

positive = false; 第一个是‘-’则负数标记激活

}

if (original[i] == '-' && i != 0)

{

ok = false; 不在第一位出现‘-’则出错

}

if (original[0] == '.' || original[le - 1] == '.')

{

ok = false; 开头结尾出现小数点错

}

if (original[i] == '.')

{

xiaoshu++; 记录小数点个数

}

if (xiaoshu > 1)

{

ok = false; 多了错

}

for (int i = 0; i < (original.length()); i++)

{

if (original[i] == '.')

okpoint = true;

}

if (okpoint == true) 如果有小数点

{

int sit = 0;

for (int i = 0; i <( original.length()); i++)

{

if (original[i] == '.')

{

sit = i; 记录小数点位置

}

}

}

else

{

if (original[0] == '0') 开头没有0；

ok = false;

}

if (original[i] != '0' && original[i] != '1' && original[i] != '.' && original[i] != '-')

{

ok = false; 判定数码是否正确

}

}

if (ok == false) 错误就重来

{

cout << "\n\n你搞砸了，重来！\n\n";

system("pause");

system("cls");

}

else

{

cout << "\n\n转换已开始，请稍候......\n\n"; 正确就继续

system("pause");

system("cls");

}

}

后半段（改变进制部分）

double ten = 0.0; 传递给下一个函数的十进制数值

if (positive == true)

{

int le;

le = original.length(); 测长度

int sit = 0;

bool xiaoshu = false; 判定小数（同上）

for (int o = 0; o < le; o++)

{

if (original[o] == '.')

{

xiaoshu = true;

sit = o;

}

}

if (xiaoshu == true) 是小数：

{

int upmi = sit - 1; 上半段最高幂次

int downmi = -1; 下半段最高幂次

int i; 循环变量i，j

int j;

int upor[MAX] = { 0,0,0,0,0,0 }; 上半段数组

int downor[MAX] = { 0,0,0,0,0,0 }; 下半段数组

for (i = 0; i < sit; i++)

{

if (original[i] == '1')

{

upor[i] = 1;

}

if (original[i] == '0')

{

upor[i] = 0;

} 字符到数字转变

ten += ((pow(2, upmi) \* upor[i])); 根据幂与为此关系累加

upmi--;

}

for (j = sit + 1; j < le; j++) 同理。

{

int u = 0;

if (original[j] == '1')

{

downor[u] = 1;

}

if (original[j] == '0')

{

downor[u] = 0;

}

ten += ((pow(2, downmi) \* downor[u]));

downmi--;

}

}

else

{

int mi = le - 1;

int i;

int ori[MAX] = { 0,0,0,0,0,0 };

for (i = 0; i < le; i++)

{

if (original[i] == '1')

{

ori[i] = 1;

}

if (original[i] == '0')

{

ori[i] = 0;

}

ten += ((pow(2, mi) \* ori[i]));

mi--;

}

}

}

else

负数的时候：改变循环开始位次即可，不再赘述

…………

if (object == 10)

根据传来的菜单控制符，判断下一步转换为什么进制（调用什么函数）

{

fastout(ten); ——快速输出，因为已经是十进制

}

else switch (object)

{

case 16:

{

make16(ten);

break;

}

case 8:

{

make8(ten);

break;

}

}

若为十进制，则有

double ten;

if (positive == true) 正数

{

stringstream Oss;

Oss << original;

Oss >> ten; 将original转换为ten

}

else 负数，字符位子前移一格再转换（删去1位子）

{

int n = original.length();

for (int i = 0; i < n-1; i++)

{

original[i] = original[i + 1];

}

stringstream Oss;

Oss << original;

Oss >> ten;

}

3.5 对应进制转换

void make2(double ten)

{

int i;

int j;

string first = { " " };

string end = { " " };

int temp;

int left; 初始化

int a = 0;

float b = 0; 整数小数部分

判断整数还是有小数。

a = int(ten);

b = ten - floor(ten); 整数小数部分分离

if (b == 0) 如果是整数

{

temp = a;

for (i = 0;; i++)

{

left = temp % 2;

first[i] = char(left) + '0'; 根据转换方法交替

temp = temp / 2;

if (temp == 0) 余数为零停止

{

break;

}

}

for (int j = 0; j <= i; j++)

{

end[j] = first[i - j]; 倒叙转移

}

}

else

{

int pr = 0;

temp = a;

for (i = 0;; i++)

{

left = temp % 2;

first[i] = char(left) + '0';

temp = temp / 2;

if (temp == 0)

{

break;

}

}

for (int j = 0; j <= i; j++) 前半段是整数，同上

{

end[j] = first[i - j];

}

end[i + 1] = '.'; 给一个点

for (j = i + 2;; j++)

{

b = b \* 2;

left = int(b);

end[j] = char(left) + '0'; 类型转换+‘0’

b = b - left;

pr++; 精度位数加一

if (b == 0)

{

break;

}

if (pr > PR) 另一种停止：根据精度<6

{

break;

}

}

}

display(end); 输出

}

3.6 输出函数

fastout函数与此除了输出使用cout << std::to\_string(ten)并无其他不同，不叙

void display(string out)

{

int button = 2; 转换继续控制器

system("cls"); 清屏，换行

cout << "\n\n";

if (positive==false)

{

cout << "-"; 负号加‘-’

}

cout << out;

cout << "\n是转换得到的结果 ：） 单击以继续...\n\n";

system("pause");

cout<<"继续转换？ YES or NOT\n\n1 = yes, 2 = no.";

cin >> button;

if (button == 1)

{

cout << "我们的工作马上开始\n（为了运行的稳定，请一定使用英文输入法）\n\n\n";

system("pause");

system("cls");

return ;

}

else

{

running = false; 终止变量running为假

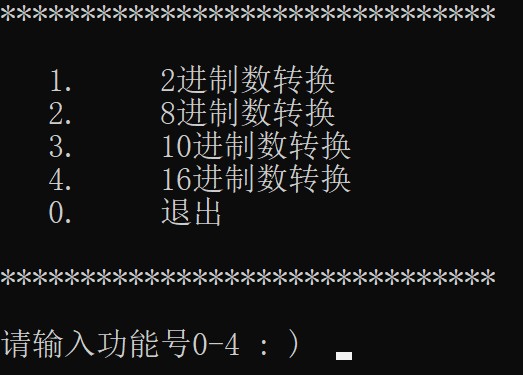
}

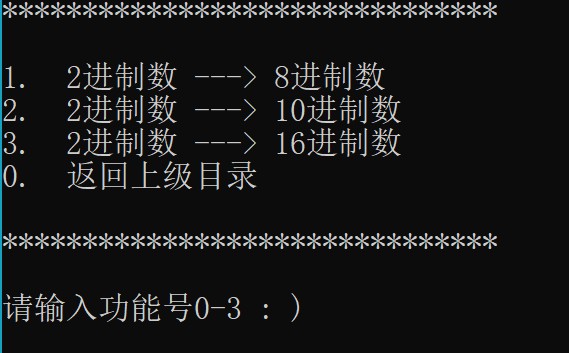
}

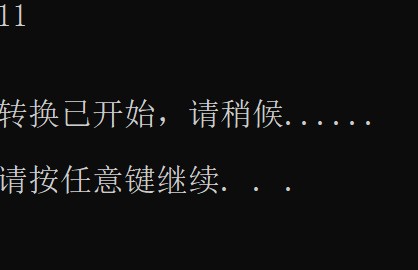
4 程序测试

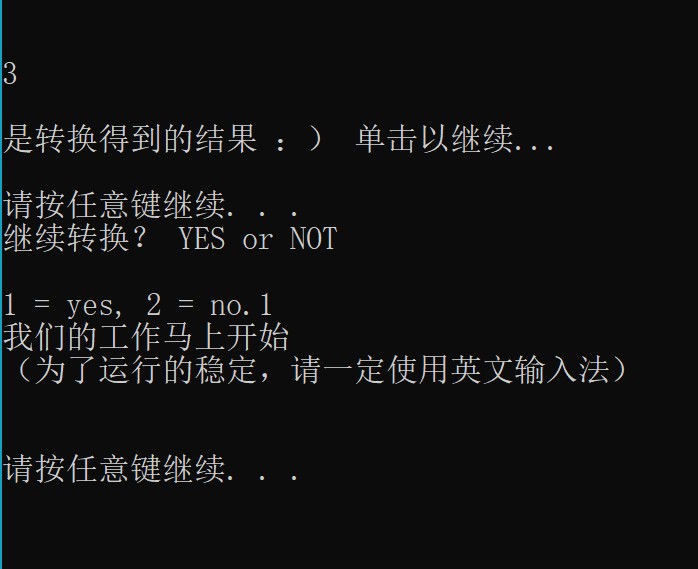
4.1正常情况

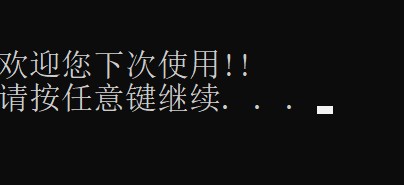
流程如下图所示





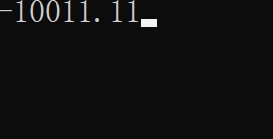
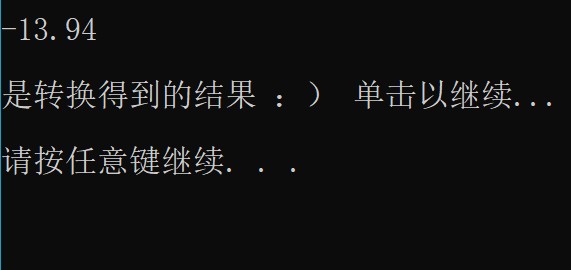




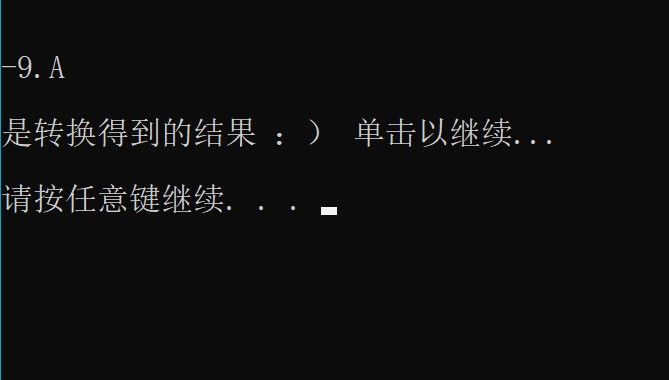
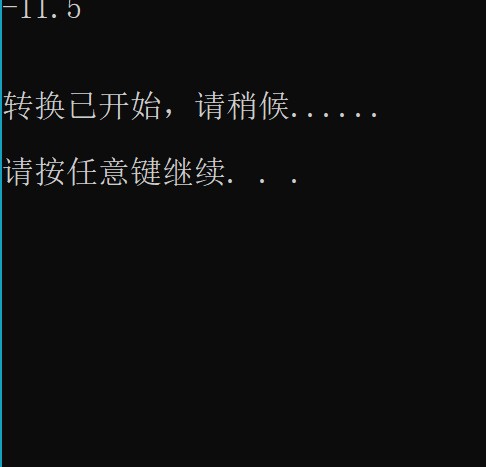


具体转换结果示例：

2->8(负数+小数)

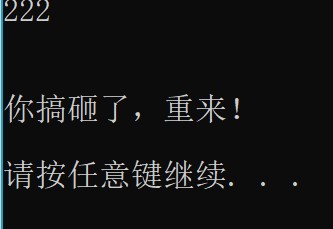


8->16(负数+小数)



4.2异常情况

出错会跳回上级菜单（以二进制输入出错为例）：



5 心得体会

5.1心得体会

1、对于提高界面美观性的个人方法：使用pause和屏幕清空组合技

2、简单的随用随测方法：构建一个简单的测试函数，进行各个子函数的测试

3、养成写日志的习惯，例如：

//菜单开发日志：

//主函数用于调用一级菜单，一级菜单调用转换功能函数；功能函数中包含输入输出和分离过程。

//将十进制作为中转，构建2、8、16到10和10到2、8、16的字符串转换功能函数

//输入-转化为10进制开发日志：

//首先进行合法性检查和负数检查。以2进制为起始的为例；

//合法性:开头没有零、小数点、负号没有两次且不在开头和末尾

// 对各自进制输入的数字要求有不同for循环做。

//负数检查；给一个负数标记，最后加上符号。全局变量负号。

//10进制转其他进制开发日志：做数组然后先做辗转相除。

4、为了减少代码的冗余度，应当把相同的功能集中到特别的一个函数身上，并通过传参数方式进行连接

参考文献

[1] 谭浩强.C++程序设计(第3版).中国.清华大学出版社.2011.