1. 舍入和截取

1) 整理“我的数学库”

创建MyMath.h头文件，加入以下声明。

```
////////////////////////////////////////////////////////////////////////////
// 数据类型
////////////////////////////////////////////////////////////////////////////

// 逻辑类型
typedef short BOOL;
#define TRUE   1
#define FALSE 0
#define BSF "%hd"
#define BPF "%hd"

// 整数类型
typedef long INTEGER;
#define ISF "%ld"
#define IPF "%ld"

// 实数类型
typedef double REAL;
#define epsilon 1e-8
#define RSF "%lg"
#define RPF "%.15g"

////////////////////////////////////////////////////////////////////////////
// 整数运算
////////////////////////////////////////////////////////////////////////////

// 最大值
INTEGER Max(INTEGER x, INTEGER y);
// 最小值
INTEGER Min(INTEGER x, INTEGER y);

// 判断奇数
BOOL IsOdd(INTEGER x);
// 判断偶数
BOOL IsEven(INTEGER x);

// 判断倍数
BOOL IsMultiple(INTEGER m, INTEGER n);

// 最大公约数
INTEGER Gcd(INTEGER m, INTEGER n);
// 最小公倍数
INTEGER Lcm(INTEGER m, INTEGER n);

// 整数的最高位 (0的最高位按0处理)
INTEGER TopDigit(INTEGER x);
// 整数的最低位 (0的最低位按0处理)
INTEGER LowDigit(INTEGER x);
// 整数x的第n位数字
INTEGER Digit(INTEGER x, INTEGER n);
// 整数的位数 (0作1位数计)
INTEGER NumDigit(INTEGER x);
// 整数各位数字之和
INTEGER SumDigit(INTEGER x);

// 阶乘函数
REAL Fac(INTEGER n);
// 排列数函数
REAL Arg(INTEGER m, INTEGER n);
// 组合数函数
REAL Cmb(INTEGER m, INTEGER n);
```

```
// 以任意数制输出整数
void PrintInteger(INTEGER x, INTEGER r);

……

////////////////////////////////////////////////////////////////////////
// 实数运算
////////////////////////////////////////////////////////////////////////

// 最大值
REAL FMax(REAL x, REAL y);
// 最小值
REAL FMin(REAL x, REAL y);

// 判断大于
BOOL GT(REAL x, REAL y);
// 判断大于等于
BOOL GE(REAL x, REAL y);
// 判断小于
BOOL LT(REAL x, REAL y);
// 判断小于等于
BOOL LE(REAL x, REAL y);
// 判断等于
BOOL EQ(REAL x, REAL y);
// 判断不等于
BOOL NE(REAL x, REAL y);

// 一元一次方程
// ax + b = 0
//   0 无解
// -1 唯一解
// -2 无穷多解
INTEGER LinearEquation(REAL *x, REAL a, REAL b);

// 一元二次方程
// ax2 + bx + c = 0
//   0 无解
// -1 唯一解
// -2 无穷多解
//   1 两个相等的实根
//   2 两个不等的实根
//   3 两个共轭的虚根
INTEGER QuadraticEquation(REAL *x1, REAL *x2, REAL a, REAL b, REAL c);

// 二元一次方程组
// a1x + b1y = c1
// a2x + b2y = c2
//   0 无解
//   1 唯一解
//   2 无穷多解
INTEGER EquationGroup(REAL *x, REAL *y, REAL a1, REAL b1, REAL c1, REAL a2, REAL b2, REAL c2);

// 符号函数
REAL Sign(REAL x);
// 任意实数的任意整数次幂
REAL Power(REAL x, INTEGER n);
// 取整
INTEGER ToInteger(REAL x);
……
```

创建MyMath.c源程序文件，实现以上函数。

```
////////////////////////////////////////////////////////////////////////
// 整数运算
////////////////////////////////////////////////////////////////////////
```

```
// 最大值
INTEGER Max(INTEGER x, INTEGER y)
{
    return x >= y ? x : y;
}

// 最小值
INTEGER Min(INTEGER x, INTEGER y)
{
    return x <= y ? x : y;
}

……

////////////////////////////////////////////////////////////////////////////
// 实数运算
////////////////////////////////////////////////////////////////////////////

// 最大值
REAL FMax(REAL x, REAL y)
{
    return x >= y ? x : y;
}

// 最小值
REAL FMin(REAL x, REAL y)
{
    return x <= y ? x : y;
}

……
```

编写主函数，逐一检验以上函数是否正确。

2) 舍入

在MyMath.h和MyMath.c中添加舍入函数。

REAL Round(REAL x, INTEGER n);

说明：对实数x进行舍入处理，保留n位小数。修改主函数对舍入函数进行测试。

```
int main()
{
    REAL a, c;
    INTEGER b;
    printf("实数: ");
    scanf(RSF, &a);
    printf("小数: ");
    scanf(ISF, &b);
    c = Round(a, b);
    printf("结果: " RPF "\n", c);
    return 0;
}
```

测试用例:

实数: 45.012↵ 小数: 2↵ 结果: 45.01	实数: 45.014999998↵ 小数: 2↵ 结果: 45.02	实数: 45.0153↵ 小数: 2↵ 结果: 45.02	实数: -32.59998↵ 小数: 4↵ 结果: -32.6
实数: -18.499↵ 小数: 0↵ 结果: -18	实数: -18.499999991↵ 小数: 0↵ 结果: -19	实数: -18.84↵ 小数: 0↵ 结果: -19	实数: 21845039.76↵ 小数: -4↵ 结果: 21850000

3) 截取

在MyMath.h和MyMath.c中添加截取函数。

REAL Trunc(REAL x, INTEGER n);

说明：对实数x进行截断处理，保留n位小数。编写下面的主函数对截取函数进行测试。

```
int main()
{
    REAL a, c;
    INTEGER b;
    printf("实数: ");
    scanf(RSF, &a);
    printf("小数: ");
    scanf(ISF, &b);
    c = Trunc(a, b);
    printf("结果: " RPF "\n", c);
    return 0;
}
```

测试用例:

实数: 45.01002↵ 小数: 2↵ 结果: 45.01	实数: 45.01999↵ 小数: 2↵ 结果: 45.01	实数: 45.019999991↵ 小数: 2↵ 结果: 45.02	实数: -32.4000582↵ 小数: 4↵ 结果: -32.4
实数: -18.005↵ 小数: 0↵ 结果: -18	实数: -18.991↵ 小数: 0↵ 结果: -18	实数: -18.999999991↵ 小数: 0↵ 结果: -19	实数: 54036.18↵ 小数: -3↵ 结果: 54000

2. 复数运算

在“我的数学库”文件夹中创建MyComplex.h、MyComplex.c, 专用于处理复数运算。

复数的输入、输出格式规定为:“±*a*␣±␣*bi*”。编写下面的主函数, 完成复数的输入和输出。

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "MyMath.h"
#include "MyComplex.h"

int main()
{
    double rp, ip;
    char s, t;

    printf("复数: ");
    ////////////////////////////////////////////////////////////////////////////
    // 复数的输入
    ////////////////////////////////////////////////////////////////////////////
    scanf(RSF "␣%c" RSF "␣%c", &rp, &s, &ip, &t);
    if (s != '+' && s != '-' || t != 'i')
    {
        puts("不正确的复数!");
        exit(1);
    }
    if (s == '-')
    {
        ip = -ip; // 修改虚部系数
    }

    printf("复数: ");
    ////////////////////////////////////////////////////////////////////////////
    // 复数的输出
    ////////////////////////////////////////////////////////////////////////////
    printf(RPF, rp);
    putchar('␣');
    putchar(ip >= 0 ? '+' : '-');
    putchar('␣');
    printf(RPF, fabs(ip));
    putchar('i');

    putchar('\n');
    return 0;
}
```

测试用例:

复数: 3.6␣-␣4.5i↵ 复数: 3.6␣-␣4.5i	复数: 0␣+␣2.5i↵ 复数: 0␣+␣2.5i	复数: -1.8␣+␣0i↵ 复数: -1.8␣+␣0i	复数: 5␣#␣4i↵ 不正确的复数!	复数: 3␣+␣4a↵ 不正确的复数

1) 复数的输入和输出

void InputComplex(REAL *rp, REAL *ip);

说明: rp为指向复数实部的指针, ip为指向复数虚部系数的指针。函数以标准格式输入复数。

void OutputComplex(REAL rp, REAL ip);

说明: rp为复数的实部, ip为复数的虚部系数。函数以标准格式输出复数。

在MyComplex.h中添加函数声明, 在MyComplex.c中写入下面的命令, 并编写函数。

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "MyMath.h"
#include "MyComplex.h"
```

用下面的主函数进行测试。

```
int main()
{
    double rp, ip;
    printf("复数: ");
    InputComplex(&rp, &ip);
    printf("复数: ");
    OutputComplex(rp, ip);
    putchar('\n');
    return 0;
}
```

2) 复数的运算

void AddComplex(REAL *zrp, REAL *zip, REAL xrp, REAL xip, REAL yrp, REAL yip);

void SubComplex(REAL *zrp, REAL *zip, REAL xrp, REAL xip, REAL yrp, REAL yip);

void MulComplex(REAL *zrp, REAL *zip, REAL xrp, REAL xip, REAL yrp, REAL yip);

void DivComplex(REAL *zrp, REAL *zip, REAL xrp, REAL xip, REAL yrp, REAL yip);

说明: 以上四个函数分别完成复数的加法、减法、乘法和除法运算。其中, zrp、zip为指向结果复数z的实部和虚部系数的指针, xrp、xip为复数x的实部和虚部系数, yrp、yip为复数y的实部和虚部系数。

要求: 当除数为0时, 报告错误并强行结束。

用下面的主函数测试复数的加法。

```
int main()
{
    REAL arp, aip, brp, bip, crp, cip;
    printf("a = ? ");
    InputComplex(&arp, &aip);
    printf("b = ? ");
    InputComplex(&brp, &bip);
    AddComplex(&crp, &cip, arp, aip, brp, bip);
    printf("c = ");
    OutputComplex(crp, cip);
    putchar('\n');
    return 0;
}
```

测试用例:

a = ? 5.1 - 6.4i↵ b = ? -2.5 + 7.2i↵ c = 2.6 + 0.8i	a = ? 3.5 - 2.1i↵ b = ? -3.5 + 4.3i↵ c = 0 - 2.2i	a = ? 1 + 1i↵ b = ? 1 - 1i↵ c = 2 + 0i

修改主函数, 测试复数的减法。测试用例:

a = ? 7.2 - 3.6i↵ b = ? 3.8 + 1.5i↵ c = 3.4 - 5.1i	a = ? -1.6 + 3.5i↵ b = ? 2.5 + 3.5i↵ c = -4.1 + 0i	a = ? 1 + 1i↵ b = ? 1 - 1i↵ c = 0 + 2i

修改主函数, 测试复数的乘法。测试用例:

a = ? 2.4 + 1.8i↵ b = ? 1.2 + 0.6i↵ c = 1.8 + 3.6i	a = ? -2.5 + 0.5i↵ b = ? 1.5 -7.5i↵ c = 0 + 19.5i	a = ? 1 + 1i↵ b = ? 1 - 1i↵ c = 2 + 0i

修改主函数，测试复数的除法。测试用例:

a = ? 1.4 + 3.5i↵ b = ? 0.7 - 2.1i↵ c = -1.3 + 1.1i	a = ? -7.5 + 1.5i↵ b = ? 12.5 -2.5i↵ c = -0.6 + 0i	a = ? 1 + 1i↵ b = ? 1 - 1i↵ c = 0 + 1i	a = ? 4.5 + 2.7i↵ b = ? 0 + 0i↵ 除以零错误!

3) 复数功能菜单

void MenuComplex();

该无值函数显示如下图所示的复数运算功能菜单。

```
A-加法 S-减法 M-乘法 D-除法 Q-退出 > _
```

若用户选择“Q”或“q”，则直接退出。若用户选择“A”或“a”、“S”或“s”、“M”或“m”、“D”或“d”，则输入两个复数、分别计算复数的和、差、积、商并输出相应的结果，然后退出；若用户错选，则报告错误并退出。用下面的主函数来测试。

```
#include "MyMath.h"
#include "MyComplex.h"

int main()
{
    MenuComplex();
    return 0;
}
```

测试用例:

A-加法 S-减法 M-乘法 D-除法 Q-退出 > A↵ a = ? 5.1 - 6.4i↵ b = ? -2.5 + 7.2i↵ c = 2.6 + 0.8i	A-加法 S-减法 M-乘法 D-除法 Q-退出 > s↵ a = ? 7.2 - 3.6i↵ b = ? 3.8 + 1.5i↵ c = 3.4 - 5.1i
A-加法 S-减法 M-乘法 D-除法 Q-退出 > m↵ a = ? 2.4 + 1.8i↵ b = ? 1.2 + 0.6i↵ c = 1.8 + 3.6i	A-加法 S-减法 M-乘法 D-除法 Q-退出 > d↵ a = ? 1.4 + 3.5i↵ b = ? 0.7 - 2.1i↵ c = -1.3 + 1.1i
A-加法 S-减法 M-乘法 D-除法 Q-退出 > q↵	A-加法 S-减法 M-乘法 D-除法 Q-退出 > T↵ 不正确的选项!