**第3\_2题**

*设计一个复数类型，输入实部和虚部生成一个复数，可进行两个复数求和、两个复数求差、两个复数求积运算。*

**算法描述：**

复数类型定义为结构体complexNum，包含两个double类型变量分别为复数的实部和虚部。

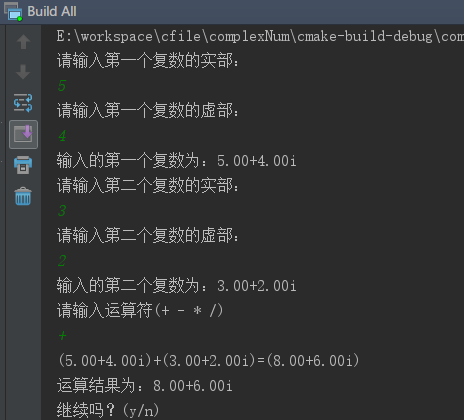
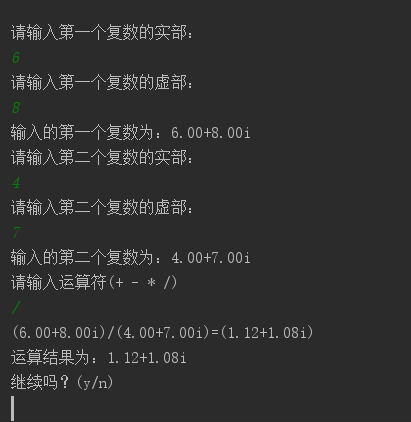
定义4个函数，传入两个complexNum类型的参数，返回值为complexNum，分别实现复数的加减乘除运算。

程序执行的过程中，调用这几个函数即可实现对复数的运算。

**源程序：**

no3\_2.c

**typedef struct** {  
 **double** n;  
 **double** i;  
} complexNum;  
  
complexNum add(complexNum complexNum1, complexNum complexNum2) {  
 /\*\*  
 \* @brief 两个复数相加  
 \* @param complexNum1 第一个复数  
 \* @param complexNum2 第二个复数  
 \* @return 两个复数相加的结果  
 \* @author 张俊华 16030199025  
 \*/  
 complexNum complexNum3;  
 complexNum3.n = complexNum1.n + complexNum2.n;  
 complexNum3.i = complexNum1.i + complexNum2.i;  
 **return** complexNum3;  
}  
  
complexNum del(complexNum complexNum1, complexNum complexNum2) {  
 /\*\*  
 \* @brief 两个复数相减  
 \* @param complexNum1 第一个复数  
 \* @param complexNum2 第二个复数  
 \* @return 两个复数相减的结果  
 \* @author 张俊华 16030199025  
 \*/  
 complexNum complexNum3;  
 complexNum3.n = complexNum1.n - complexNum2.n;  
 complexNum3.i = complexNum1.i - complexNum2.i;  
 **return** complexNum3;  
}  
  
complexNum multi(complexNum complexNum1, complexNum complexNum2) {  
 /\*\*  
 \* @brief 两个复数相乘  
 \* @param complexNum1 第一个复数  
 \* @param complexNum2 第二个复数  
 \* @return 两个复数相乘的结果  
 \* @author 张俊华 16030199025  
 \*/  
 complexNum complexNum3;  
 complexNum3.n = (complexNum1.n \* complexNum2.n) + (complexNum1.i \* complexNum2.i);  
 complexNum3.i = (complexNum1.n \* complexNum2.i) + (complexNum2.n \* complexNum1.i);  
 **return** complexNum3;  
}  
  
complexNum div(complexNum complexNum1, complexNum complexNum2) {  
 /\*\*  
 \* @brief 两个复数相除  
 \* @param complexNum1 第一个复数  
 \* @param complexNum2 第二个复数  
 \* @return 两个复数相除的结果  
 \* @author 张俊华 16030199025  
 \*/  
 complexNum complexNum3;  
 complexNum3.n = complexNum1.n / (complexNum2.n \* complexNum2.n + complexNum2.i \* complexNum2.i)  
 \* complexNum2.n  
 + complexNum2.i / (complexNum2.n \* complexNum2.n + complexNum2.i \* complexNum2.i)  
 \* complexNum2.i;  
 complexNum3.i = complexNum1.n / (complexNum2.n \* complexNum2.n + complexNum2.i \* complexNum2.i)  
 \* complexNum2.i  
 + complexNum2.i / (complexNum2.n \* complexNum2.n + complexNum2.i \* complexNum2.i)  
 \* complexNum2.n;  
 **return** complexNum3;  
}  
  
**char** \*show(complexNum aComplexNum) {  
 /\*\*  
 \* @brief 复数转字符串  
 \* @param aComplexNum 复数  
 \* @return 复数的字符串表示  
 \* @author 张俊华 16030199025  
 \*/  
 **static char** complexNumString[50];  
 sprintf(complexNumString, "%.2lf+%.2lfi", aComplexNum.n, aComplexNum.i);  
 **return** complexNumString;  
}  
  
**int** main() {  
 complexNum complexNum1;  
 complexNum complexNum2;  
 **while** (1) {  
 puts("请输入第一个复数的实部：");  
 scanf("%lf", &complexNum1.n);  
 puts("请输入第一个复数的虚部：");  
 scanf("%lf", &complexNum1.i);  
  
 printf("输入的第一个复数为：%.2lf+%.2lfi\n", complexNum1.n, complexNum1.i);  
  
 puts("请输入第二个复数的实部：");  
 scanf("%lf", &complexNum2.n);  
 puts("请输入第二个复数的虚部：");  
 scanf("%lf", &complexNum2.i);  
  
 printf("输入的第二个复数为：%.2lf+%.2lfi\n", complexNum2.n, complexNum2.i);  
  
 puts("请输入运算符(+ - \* /)");  
 **char** operator;  
 **while** (((operator = (**char**) getchar()) != '\n') && operator != EOF);  
  
 operator = (**char**) getchar();  
  
 **switch** (operator) {  
 **case** '+':  
 printf("(%s)+", show(complexNum1));  
 printf("(%s)", show(complexNum2));  
 printf("=(%s)\n", show(add(complexNum1, complexNum2)));  
 printf("运算结果为：%s", show(add(complexNum1, complexNum2)));  
 **break**;  
  
 **case** '-':  
 printf("(%s)-", show(complexNum1));  
 printf("(%s)", show(complexNum2));  
 printf("=(%s)\n", show(del(complexNum1, complexNum2)));  
 printf("运算结果为：%s", show(del(complexNum1, complexNum2)));  
 **break**;  
  
 **case** '\*':  
 printf("(%s)\*", show(complexNum1));  
 printf("(%s)", show(complexNum2));  
 printf("=(%s)\n", show(multi(complexNum1, complexNum2)));  
 printf("运算结果为：%s", show(multi(complexNum1, complexNum2)));  
 **break**;  
  
 **case** '/':  
 printf("(%s)/", show(complexNum1));  
 printf("(%s)", show(complexNum2));  
 printf("=(%s)\n", show(div(complexNum1, complexNum2)));  
 printf("运算结果为：%s", show(div(complexNum1, complexNum2)));  
 **break**;  
  
 **default**:  
 puts("请检查输入！");  
 **break**;  
 }  
 puts("\n继续吗？(y/n)");  
 **while** (((operator = (**char**) getchar()) != '\n') && operator != EOF);  
 operator = (**char**) getchar();  
 **if** (operator == 'n')  
 **break**;  
 }  
  
 **return** 0;  
}

**测试数据（输入、输出）：**