```c
#define _OUTPUT_C
#ifdef _OUTPUT_C
/*
*
   ================================================================================
   =========
*       Filename:   output.c
*    Description:   some output situations
*        Version:   1.1.2
*        Created:   2017.10.26 16:05:16
*      Time Used:   10h
*  Last Modified:   2017.10.28 01:49
*    Last Change:   error treatment added in dots input
*         Author:   伍瀚缘(Tree Wu), why2000@hust.edu.cn
*        Company:   Huazhong University of Science and Technology
*
   ================================================================================
   =========
*/
#include<why_calculator.h>
//#define NOW 1
#ifdef NOW//调试用

int main(void) {


    return 0;
}

#endif
extern int execstatus;//"why_calculator.c"
extern int preci;//"why_calculator.c"

//请求输入精度
int preciquest(void) {

    char foo;
    printf("输入的精度要求为一个大于等于0，小于等于15的整数\n");
    printf("请输入精度\n");
    /*那么为什么下面那种方法可以，而这样清空并关闭的话在连续输入两次2+.23之类
      的表达式的时候仍有问题呢？*/
    //fflush(stdin);

    int right = scanf("%d", &preci);//这里会出现输入残留问题么？
    scanf("%c", &foo);//当然会的啊
    if (right&&preci >= 0 && preci <= 15&&foo=='\n') {
        printf
          ("*************************************************************************
           ***************\n");
        printf("精度设置成功，当前精度为%d\n", preci);
        return 1;
    }
    else {
        printf("精度不符合要求，请重新输入\n");
        printf
          ("*************************************************************************
```

```
            ***************\n");
49          //setbuf(stdin, NULL);//这一句是与上面的fflush函数对应使用的
50
51          //************************可以这样清除缓冲区（而非关闭缓冲区）
              ****************************
52
53          char*buf = (char *)malloc(BUFSIZ);
54          free(buf);
55          buf = (char *)malloc(BUFSIZ);
56          setbuf(stdin,buf);
57          //
              ********************************************************************
              ***************
58          return preciquest();
59      }
60
61  }
62
63  //打印程序简介以及操作方式
64  void baseoutput(void) {
65      printf
          ("********************************************************************
          ***********\n");
66      printf("*                                  简易计算器V1.1.2
                        *\n");
67      printf("*此计算器可实现基本的四则运算以及部分的乘方/开方运算
                        *\n");
68      printf("*第一次使用请输入 help 以查看具体操作方式及输入要求
                        *\n");
69      printf("*输入一个合法的算数表达式后会在下一行输出算式及结果，并可继续输入
          新的表达式      *\n");
70      printf("*退出程序请输入quit或exit
                        *\n");
71      printf("*更改精度请输入change
                        *\n");
72      printf("*高精度下，输出结果的最后1-3位可能丢失精度，请以四舍五入的方式截取
          有效部分        *\n");
73      printf("*
                  -by why*\n");
74      printf("*
          2017/10/26 16:01*\n");
75      printf
          ("********************************************************************
          ***********\n");
76      preciquest();
77      endingoutput();
78
79  }
80
81  //一次计算完成后请求输入
82  void endingoutput(void) {
83      printf
          ("********************************************************************
          ***********\n");
84      printf("请继续输入表达式进行计算，输入quit或exit以结束程序，输入help以打开
          帮助界面\n");
```

```c
    printf("输入change以更改精度\n");
    printf
      ("*************************************************************************
      ***********\n");
}

//蠢蠢的分精度输出
void outputresult(double result) {//result为计算的结果
    int intlen = 0;
    switch (preci) {
    case 0:
        printf("%.0f\n", result);
        break;
    case 1:
        printf("%.1f\n", result);
        break;
    case 2:
        printf("%.2f\n", result);
        break;
    case 3:
        printf("%.3f\n", result);
        break;
    case 4:
        printf("%.4f\n", result);
        break;
    case 5:
        printf("%.5f\n", result);
        break;
    case 6:
        printf("%.6f\n", result);
        break;
    case 7:
        printf("%.7f\n", result);
        break;
    case 8:
        printf("%.8f\n", result);
        break;
    case 9:
        printf("%.9f\n", result);
        break;
    case 10:
        printf("%.10f\n", result);
        break;
    case 11:
        printf("%.11f\n", result);
        break;
    case 12:
        printf("%.12f\n", result);
        break;
    case 13:
        printf("%.13f\n", result);
        break;
    case 14:
        printf("%.14f\n", result);
        break;
    case 15:
```

```
139             printf("%.15f\n", result);
140             break;
141         }
142
143
144     }
145
146     //打印帮助界面
147     void outputhelp(void) {
148         printf
              ("*********************************************************************
              ***********\n");
149         printf("*****************************请务必使用全英文输入
              *********************************\n");
150         printf("输入帮助：\n");
151         printf("四则运算请直接输入表达式，加：+，减：-，乘：*，除：/，输入范例：1
              +2\n");
152         printf("乘方运算符：^，其左侧为底数，右侧为指数，输入范例：2^5\n");
153         printf("开方运算符：@，其左侧为被开方数，右侧为根次数，输入范例：5@2\n");
154         printf("负数：请勿在运算符的后面紧跟负号，应按照数学运算式的书写规范进行输
              入\n");
155         printf("括号：()请务必使用英文括号，用法与数学上相同\n");
156         printf
              ("*********************************************************************
              ***********\n");
157
158     }
159
160     //报错系统
161     int errorfound(const int index) {//index为错误索引
162     /*************************改动此函数时不要 删除 枚举类型中的任何项
          ****************************/
163         enum errornum { TMRIGHT = 1, TMLEFT, NEGSQRT, COMPRESULT, ZERODENO,
              ABNOMALSYN, TMSYN, NOINPUT, TMDOTS};
164         switch (index) {
165         case TMRIGHT:
166             printf("右括号过多)))\n");
167             break;
168         case TMLEFT:
169             printf("左括号过多(((\n");
170             break;
171         case NEGSQRT:
172             printf("开非奇整数次方时底数小于零@@@\n");
173             break;
174         case COMPRESULT:
175             printf("结果为虚数的幂运算^^^\n");
176             break;
177         case ZERODENO:
178             printf("分母为零000\n");
179             break;
180         case ABNOMALSYN:
181             printf("无法识别的输入$$$\n");
182             printf("请检查运算式中是否混入字母或使用了中文括号\n");
183             break;
184         case TMSYN:
185             printf("错误的运算式%%%%%%\n");
```

```
186             break;
187         case NOINPUT:
188             printf("无输入???\n");
189             break;
190         case TMDOTS:
191             printf("检测到多重小数点...\n");
192             break;
193         }
194 
195         execstatus = 2;
196         return 2;
197     }
198     #endif
199 
```