

constant.c

```
1
2 #include "constant.h"
3 #include<stdbool.h>
4 #include<string.h>
5 #include<stdio.h>
6 #include <stddef.h>
7 #include<stdlib.h>
8 #include <ctype.h>
9
10 int COND_OPERATORS_SIZE = 6;
11 char COND_OPERATORS[6][5] = {
12     "lt", "gt", "eq", "neq", "le", "ge"
13 };
14
15 int ARITHMATIC_OPERATORS_SIZE = 6;
16 char ARITHMATIC_OPERATORS[6][5] = {
17     "add", "sub", "mul", "div", "dif", "rem"
18 };
19
20 bool isArithOp(char *ch){
21     for(int i=0; i<ARITHMATIC_OPERATORS_SIZE; i++){
22         if( strcmp(ARITHMATIC_OPERATORS[i], ch) == 0 ) return true;
23     }
24     return false;
25 }
26
27 bool isCondOpValid(char* op){
28     //char arr[6][5] = { "lt","gt","eq","neq","le","ge" };
29     for(int i=0; i<COND_OPERATORS_SIZE; i++){
30         if(strcmp(COND_OPERATORS[i],op) == 0) return true;
31     }
32     return false;
33 }
34
35 bool isConditionValid(double left, char* op, double right){
36     if( strncmp(op,"lt",2) == 0) return (left < right);
37     if( strncmp(op,"gt",2) == 0) return (left > right);
38     if( strncmp(op,"eq",2) == 0) return (left == right);
39
40     if( strncmp(op,"le",2) == 0) return (left <= right);
41     if( strncmp(op,"ge",2) == 0) return (left >= right);
42     if( strncmp(op,"neq",2) == 0) return (left != right); // 2 fine also
43 }
44
45
46 double getValue(double left, double right, char *op){
47     // "add", "sub"
48     if( strncmp(op,ARITHMATIC_OPERATORS[0],3) == 0 ) {
49         //printf("\nadd %lf %lf\n",left,right);
50         return left+right;
51     }
52     if( strncmp(op,ARITHMATIC_OPERATORS[1],3) == 0 ) return left-right;
53 }
```

```
54 // "mul", "div"
55 if( strcmp(op,ARITHMATIC_OPERATORS[2],3) == 0 ) return left*right;
56 if( strcmp(op,ARITHMATIC_OPERATORS[3],3) == 0 ) return left/( right == 0 ? 1 : right) ;
57
58 // "dif", "rem"
59 if( strcmp(op,ARITHMATIC_OPERATORS[4],3) == 0 ) return left>=right ? left-right : right-
left;
60 if( strcmp(op,ARITHMATIC_OPERATORS[5],3) == 0 ) return ((int)left) % ((int)right);
61
62 }
63
```