Jisuan.c:

#include "math.h"

#include "stdlib.h"

#define MAX 200

typedef struct{

char data[MAX];

int top;

}Stack;

typedef struct{

double data[MAX];

int top;

}dStack;

Stack\* CreatStack(void);

dStack\* CreatStack2(void);

int IsEmpty(Stack\* sp);

int IsEmpty2(dStack\* sp);

int IsFull(Stack\* sp);

int IsFull2(dStack\* sp);

int IsOpt(char c);

int IsNum(char c);

int IsTrue(char\* s);

int push(Stack\* sp,char data);

int pop(Stack\* sp,char\* dp);

int push2(dStack\* sp,double data);

int pop2(dStack\* sp,double\* dp);

char youxianji(char a,char b);

double getval(char c,double left,double right);

double getres(char\* dp,dStack\* datasp,Stack\* signsp);

double add(double a,double b);

double sub(double a,double b);

double mul(double a,double b);

double chu(double a,double b);

double cf(double a,double b);

char\* opt = "+-\*/^";

double (\*func[5])(double,double)={add,sub,mul,chu,cf};

Stack\* CreatStack(void)

{

Stack\* sp;

sp = (Stack\*)malloc(sizeof(Stack));

if(!sp)

{

return 0;

}

sp->top = -1;

return sp;

}

dStack\* CreatStack2(void)

{

dStack\* sp;

sp = (dStack\*)malloc(sizeof(dStack));

if(!sp)

{

return 0;

}

sp->top = -1;

return sp;

}

int IsEmpty(Stack\* sp)

{

return sp->top == -1;

}

int IsEmpty2(dStack\* sp)

{

return sp->top == -1;

}

int IsFull(Stack\* sp)

{

return sp->top == MAX-1 ;

}

int IsFull2(dStack\* sp)

{

return sp->top == MAX-1 ;

}

int IsOpt(char c)

{

if (c == '+' || c == '-' || c == '\*' || c == '/' || c == '^')

{

return 1;

}

return 0;

}

int IsNum(char c)

{

return ((c>='0' && c<='9' ) ||(c=='.'));

}

int IsTrue(char\* s)

{

int i=0,count;

while(i <= strlen(s)-1)

{

if(!IsNum(s[i]) && !IsOpt(s[i]) && s[i] != '(' && s[i] != ')' )

{

return 0;

}

if(( ( i==0 || s[i-1]=='(' ) && ( s[i]=='\*' || s[i]=='/' ) ) )

{

return 0;

}

if(i<strlen(s)-1 && IsOpt(s[i]) && IsOpt(s[i+1]) && s[i+1] != '-' )

{

return 0;

}

if(i<strlen(s)-1 && IsNum(s[i]) && s[i+1]=='(' )

{

return 0;

}

i++;

}

return 1;

}

int push(Stack\* sp,char data)

{

if( IsFull(sp))

{

return 0;

}

sp->top++;

sp->data[sp->top] = data;

return 1;

}

int pop(Stack\* sp,char\* dp)

{

if(IsEmpty(sp) )

{

return 0;

}

\*dp = sp->data[sp->top];

sp->top--;

return 1;

}

int push2(dStack\* sp,double data)

{

if( IsFull2(sp))

{

return 0;

}

sp->top++;

sp->data[sp->top] = data;

return 1;

}

int pop2(dStack\* sp,double\* dp)

{

if(IsEmpty2(sp) )

{

return 0;

}

\*dp = sp->data[sp->top];

sp->top--;

return 1;

}

char youxianji(char a,char b)

{

int i,j;

char compare[][5] ={

{'<','<','<','<','<'},

{'<','<','<','<','<'},

{'>','>','<','<','<'},

{'>','>','<','<','<'},

{'>','>','>','>','<'} };

for(i=0;opt[i]&&opt[i] != a;i++ );

for(j=0;opt[j]&&opt[j] != b;j++ );

return compare[i][j];

}

double getval(char c,double left,double right)

{

int i;

for(i=0;opt[i]&&opt[i] != c;i++ );

if(!opt[i])

{

printf("opt error!\n");

return -250;

}

return func[i](left,right);

}

double getres(char\* dp,dStack\* datasp,Stack\* signsp)

{

int i =0,j=0;

double left,right,pre=0,res=0;

char c;

char value[20]={'\0'};

while(i <= strlen(dp)-1)

{

if( IsNum(dp[i]) || (dp[i]=='-' && i==0 ) || (dp[i]=='-' && !IsNum(dp[i-1]) && dp[i-1]!=')'))

{

value[j]=dp[i];

j++;

pre = atof(value);

if(pre>3000)

{

printf("number too large!\n");

return -1;

}

i++;

if( !IsNum(dp[i]) )

{

if(value[0]=='.' )

{

printf("input error2\n");

return -250;

}

res = pre;

push2(datasp,pre);

pre=0;

for(j=0;j<=19;j++)

{

value[j]='\0';

}

j=0;

}

}else if( IsOpt(dp[i]) )

{

if( signsp->top == -1 || signsp->data[signsp->top] == '(' )

{

push(signsp,dp[i]);

}

else if( youxianji(dp[i],signsp->data[signsp->top] ) == '>' )

{

push(signsp,dp[i]);

}

else if( youxianji(dp[i],signsp->data[signsp->top] ) == '<' )

{

do{

pop(signsp,&c);

pop2(datasp,&right);

pop2(datasp,&left);

res = getval(c,left,right);

push2(datasp,res);

}while( youxianji(dp[i],signsp->data[signsp->top] ) == '<'

&& signsp->data[signsp->top] != '('

&& signsp->top != -1 );

push(signsp,dp[i]);

}

i++;

pre=0;

}else if( dp[i] == '(' )

{

push(signsp,dp[i]);

i++;

pre=0;

}else if( dp[i] == ')' )

{

while(signsp->data[signsp->top] != '(' )

{

if(IsEmpty(signsp))

{

printf("input error3!\n");

return -250;

}

pop(signsp,&c);

pop2(datasp,&right);

pop2(datasp,&left);

res = getval(c,left,right);

push2(datasp,res);

}

pop(signsp,&c);

i++;

pre=0;

}else

{

i++;

pre=0;

}

}

while(!IsEmpty(signsp))

{

if(!IsEmpty(signsp) && IsEmpty2(datasp) )

{

printf("input error4!\n");

return -250;

}

pop(signsp,&c);

pop2(datasp,&right);

pop2(datasp,&left);

res = getval(c,left,right);

push2(datasp,res);

}

pop2(datasp,&res);

return res;

}

double add(double a,double b){ return a+b; }

double sub(double a,double b){ return a-b; }

double mul(double a,double b){ return a\*b; }

double chu(double a,double b)

{

if(b==0)

{

printf("div error!\n");

return -250;

}

return a/b;

}

double cf(double a,double b){ return pow(a,b); }

main()

{

dStack\* datasp;

Stack\* signsp;

char s[200];

int i;

double result;

datasp = CreatStack2();

if(!datasp)

{

return;

}

signsp = CreatStack();

if(!signsp)

{

return;

}

gets(s);

if(!IsTrue(s))

{

printf("input error1!...\n");

return;

}

result = getres(s,datasp,signsp);

printf("%lf\n",result);

}