# C语言实现计算器

## 简易版本

实现简单的加减乘除。

```
#include <stdio.h>

int get_option();
void print_result(int num1,int num2,int result,int option);

int main(void)
{
    int done = 0;
    int option,num1,num2,result;

    while(!done)
    {
        option = get_option();
        if(option == 5)
        {
            done = 1;
        }
        else {
            do {
                printf("\n请输入两个数：");
                scanf("%d %d",&num1,&num2);
                if(option == 4 && num2 == 0)
                {
                    printf("\n对不起，除数不能为零");
                }
                else {
                    switch(option){
                        case 1:
                            result = num1 + num2;
                            break;
                        case 2:
                            result = num1 - num2;
                            break;
                        case 3:
                            result = num1 * num2;
                            break;
                        case 4:
                            result = num1 / num2;
                    }
                    print_result(num1,num2,result,option);
                }
            }while(option == 4 && num2 == 0);
        }
    }

    return 0;
}

int get_option()
```

```
{
    int option;
    do
    {
        printf("\n ****************");
        printf("\n *    1.加法    *");
        printf("\n *    2.减法    *");
        printf("\n *    3.乘法    *");
        printf("\n *    4.除法    *");
        printf("\n *    0.退出    *");
        printf("\n ****************");

        printf("\n请输入您需要的功能：");
        scanf("%d",&option);

        if(option <1 || option > 5)
        {
            printf("对不起您输入的数字有误，请重新输入。\n");
        }
    }while(option <1 || option > 5);

        return option;
}

void print_result(int num1,int num2,int result,int option){
    char operator;
    switch(option){
        case 1:
            operator = '+';
            break;
        case 21:
            operator = '-';
            break;
        case 3:
            operator = '*';
            break;
        case 4:
            operator = '/';
            break;
    }
    printf("\n** %d %c %d = %d **\n",num1,operator,num2,result);
}
```

# 用栈实现稍复杂的四则运算

为了达到目的，首先自学了栈：按照先进后出的原则存储数据，先进入的数据被压入栈底，最后的数据在栈顶，需要读数据的时候从栈顶开始弹出数据。允许进行插入和删除操作的一端称为栈顶(top)，另一端为栈底(bottom)；栈底固定，而栈顶浮动；栈中元素个数为零时称为空栈。插入一般称为进栈（PUSH），删除则称为退栈（POP）。允许进行插入和删除操作的一端称为栈顶(top)，另一端为栈底(bottom)；栈底固定，而栈顶浮动；栈中元素个数为零时称为空栈。插入一般称为进栈（PUSH），删除则称为退栈（POP）。其次查阅多方面资料结合所学知识进行编写而成。

（1）InitStack(S)初始化：初始化一个新的栈。

（2）Empty(S)栈的非空判断：若栈S不空，则返回TRUE；否则，返回 FALSE。

（3）Push(S,x)入栈：在栈S的顶部插入元素x，若栈满，则返回 FALSE；否则，返回TRUE。

（4）Pop(S)出栈：若栈S不空，则返回栈顶元素，并从栈顶中删除该元 素；否则，返回空元素NULL。

（5）GetTop(S)取栈顶元素：若栈S不空，则返回栈顶元素；否则返回 空元素NULL。

（6）SetEmpty(S)置栈空操作：置栈S为空栈。

（7） 常以top = −1表示空栈。

进栈（PUSH）算法

①若TOP≥n时，则给出溢出信息，作出错处理（进栈前首先检查栈是否已满，满则溢出；不满则作②）；

②置TOP=TOP+1（栈指针加1，指向进栈地址）；

③S(TOP)=X，结束（X为新进栈的元素）

退栈（POP）算法

①若TOP≤0，则给出下溢信息，作出错处理(退栈前先检查是否已为空栈，空则下溢；不空则作②)；

②X=S(TOP)，（退栈后的元素赋给X）：

③TOP=TOP−1，结束（栈指针减1，指向栈顶）

（1）函数功能介绍及介绍：能够实现连续的运算，混合运算，基本上可以等同于手机上计算器。仅限于加减乘除的四则运算。（强调运算时的括号必须是英文版本的，否则运行会出错。写表达式也可以加上“=”和不加不影响运行结果，最终还是以回车进行结束得到结果）。如果能在visualc++运行，稍微调整一下，可以利用自带的功能设置一个界面，这样就可以完成跟手机自带那种计算器相同了。

加法运算：1+2回车可得3，连续运算1+2+3+4+5回车可得15

减法运算：1−2回车为−1，连续运算5−1−2回车得2

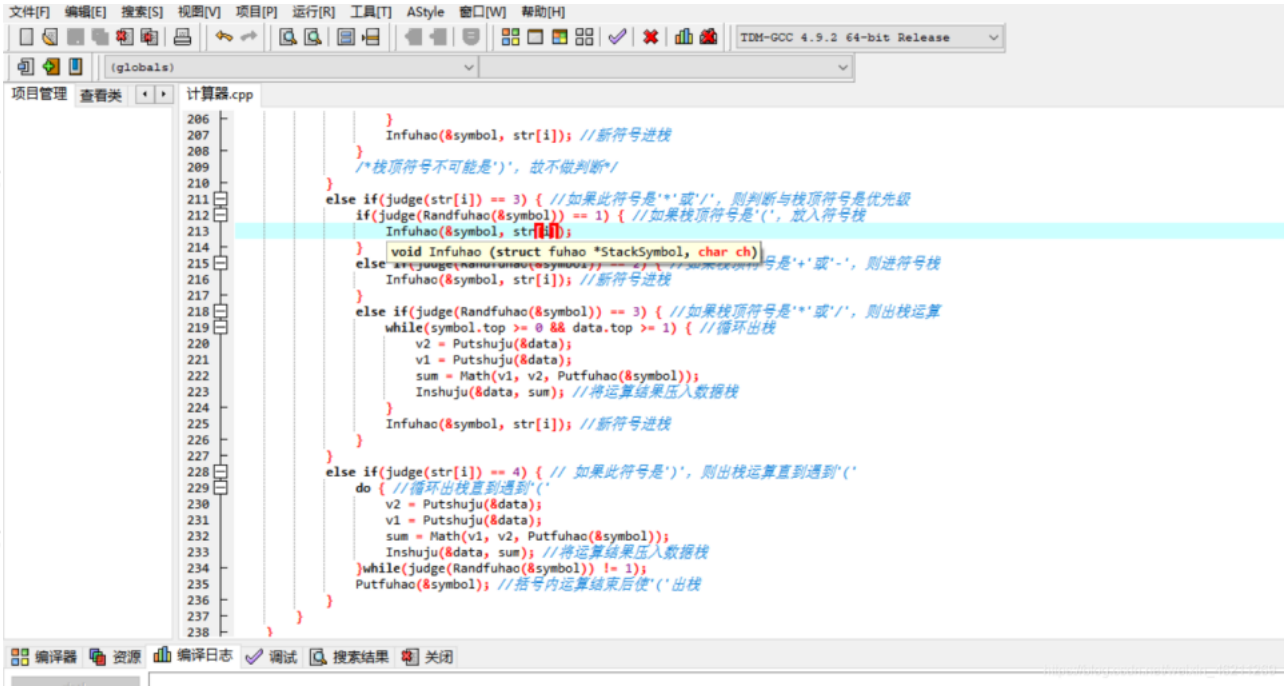
乘法：23回车6,连续运算23*4回车24

除法运算24/4回车6，24/2/2回车6、

混合运算：（5+2）*2回车14

# （一）软件环境：Devc++

我用的这个软件哈，个人感觉这里功能简单，特别容易上手。看图说话，是不是很简单嘛，又不复杂。




## （二）设计方案

根据自学所得栈进行数据和符号的存储再输出，先设立单独的数据栈符号栈，我们以top=-1为标准，判断其是否为空栈，当然也用到了学过的struct来构建栈，先把字符存进去再说，在里面我们要进行运算，然后再拿出来给我们看。于是我们就要四则运算，用switch四个case把加减乘除表达出来,这能进行简单的运算吧，那混合运算还带括号咋办，于是要用到判断优先级，加减，乘除分别同级，先左括号读到右括号我们要停止，这样就可以进行混合的运算了。后面我们经过调用前面设的函数想办法怎么把它输出来，我们就是要用到入栈顶什么的最后出栈，用个free(str)释放下内存打印出来得到结果。

## （三）函数功能：

用到了第八章内容结构结合自学内容构造栈，switch表达式来判断优先级，主要用到的为自学的栈push进栈，pop出栈，top=-1划分是否为空字符，在前言写很清楚了。

## （四）全代码：

```
#include<stdio.h>
#include<stdlib.h>


/*数据栈*/
struct shuju   //struct结构体构建栈
{
	int data[100];
	int top;
};


/*符号栈*/
struct fuhao
```

```
{
    char symbol[100];
    int top;
};


void InitOperateNum(struct shuju *StackNum)    //数据栈非空
{
   StackNum->top = -1;
}


void InitOperateSymbol(struct fuhao *StackSymbol)    //符号栈非空
{
   StackSymbol->top = -1;
}


/*存入数据栈*/
void Inshuju(struct shuju *StackNum, int num)
{
    StackNum->top ++;
    StackNum->data[StackNum->top] = num;
}


/*存入符号栈*/
void Infuhao(struct fuhao *StackSymbol, char ch)
{
    StackSymbol->top ++;
    StackSymbol->symbol[StackSymbol->top] = ch;
}


/*读取数据栈*/
int Randshuju(struct shuju *StackNum)
{
    return StackNum->data[StackNum->top];
}


/*读取符号栈*/
char Randfuhao(struct fuhao *StackSymbol)
{
    return StackSymbol->symbol[StackSymbol->top];
}


/*从数据栈取出数据*/
int Putshuju(struct shuju *StackNum)
{
    int x;
    x = StackNum->data[StackNum->top];
    StackNum->top --;
    return x;
}
```

```
/*从符号栈取出符号*/
char Putfuhao(struct fuhao *StackSymbol)
{
	char c;
	c = StackSymbol->symbol[StackSymbol->top];
	StackSymbol->top --;
	return c;
}


/*符号优先级判断*/
int judge(char ch) {
	if(ch == '(')
	 {
		return 1;
	  }
	if(ch == '+' || ch == '-') {
		return 2;
	}
	else if(ch == '*' || ch == '/') {
		return 3;
	}
	else if(ch == ')') {
		return 4;
	}
}


/*四则运算*/
int Math(int v1, int v2, char c)
{
	int sum;
	switch(c) {
		case '+' : {
			sum = v1 + v2;
			break;
		}
		case '-' : {
			sum = v1 - v2;
			break;
		}
		case '*' : {
			sum = v1 * v2;
			break;
		}
		case '/' : {
			sum = v1 / v2;
			break;
		}
	}
	return sum;
}


int main()
{
	struct shuju data;
	struct fuhao symbol;
```

```
InitOperateNum(&data);  //调用数据
InitOperateSymbol(&symbol);  //调用符号
int i, t, sum, v1, v2;
char c;
i = t = sum = 0;
char v[100] = {0};
char *str = (char *)malloc(sizeof(char)*200);
while((c = getchar()) != '\n')  //非空字符
 {
        str[i] = c;
        i ++;
}
str[i] = '\0';
for(i = 0; str[i] != '\0'; i ++) {
        if(i == 0 && str[i] == '-') {
                v[t++] = str[i];
        }
        else if(str[i] == '(' && str[i+1] == '-') {
                i ++;
                v[t++] = str[i++];
                while(str[i] >= '0' && str[i] <= '9') {
                        v[t] = str[i];
                        t ++;
                        i ++;
                }
                Inshuju(&data, atoi(v));
                while(t > 0) {
                        v[t] = 0;
                        t --;
                }
                if(str[i] != ')') {
                        i --;
                        Infuhao(&symbol, '(');
                }
        }
        else if(str[i] >= '0' && str[i] <= '9') {
                while(str[i] >= '0' && str[i] <= '9') {
                        v[t] = str[i];
                        t ++;
                        i ++;
                }
                Inshuju(&data, atoi(v));
                while(t > 0) {
                        v[t] = 0;
                        t --;
                }
                i --;
        }
        else {
                if(symbol.top == -1)
                {        //如果符号栈没有元素，直接把符号放入符号栈
                        Infuhao(&symbol, str[i]);
                }
                else if(judge(str[i]) == 1) { //如果此符号是'('，直接放入符号栈
                        Infuhao(&symbol, str[i]);
                }
                else if(judge(str[i]) == 2) { //如果此符号是'+'或'-'，判断与栈顶符号是优先级
                        if(judge(Randfuhao(&symbol)) == 1) { //如果栈顶符号是'('，放入符号栈
```

```
                        Infuhao(&symbol, str[i]);
                }
                else if(judge(Randfuhao(&symbol)) == 2) { //如果栈顶符号是'+'或'-'，则出栈运算
                        while(symbol.top >= 0 && data.top >= 1) { //循环出栈
                                v2 = Putshuju(&data);
                                v1 = Putshuju(&data);
                                sum = Math(v1, v2, Putfuhao(&symbol));
                                Inshuju(&data, sum); //将运算结果压入数据栈
                        }
                        Infuhao(&symbol, str[i]); //新符号进栈
                }
                else if(judge(Randfuhao(&symbol)) == 3) { //如果栈顶符号是'*'或'/'，则进符号栈
                        while(symbol.top >= 0 && data.top >= 1) { //循环出栈
                                v2 = Putshuju(&data);
                                v1 = Putshuju(&data);
                                sum = Math(v1, v2, Putfuhao(&symbol));
                                Inshuju(&data, sum); //将运算结果压入数据栈
                        }
                        Infuhao(&symbol, str[i]); //新符号进栈
                }
                /*栈顶符号不可能是')'，故不做判断*/
        }
        else if(judge(str[i]) == 3) { //如果此符号是'*'或'/'，则判断与栈顶符号是优先级
                if(judge(Randfuhao(&symbol)) == 1) { //如果栈顶符号是'('，放入符号栈
                        Infuhao(&symbol, str[i]);
                }
                else if(judge(Randfuhao(&symbol)) == 2) { //如果栈顶符号是'+'或'-'，则进符号栈
                        Infuhao(&symbol, str[i]); //新符号进栈
                }
                else if(judge(Randfuhao(&symbol)) == 3) { //如果栈顶符号是'*'或'/'，则出栈运算
                        while(symbol.top >= 0 && data.top >= 1) { //循环出栈
                                v2 = Putshuju(&data);
                                v1 = Putshuju(&data);
                                sum = Math(v1, v2, Putfuhao(&symbol));
                                Inshuju(&data, sum); //将运算结果压入数据栈
                        }
                        Infuhao(&symbol, str[i]); //新符号进栈
                }
        }
        else if(judge(str[i]) == 4) { // 如果此符号是')'，则出栈运算直到遇到'('
                do { //循环出栈直到遇到'('
                        v2 = Putshuju(&data);
                        v1 = Putshuju(&data);
                        sum = Math(v1, v2, Putfuhao(&symbol));
                        Inshuju(&data, sum); //将运算结果压入数据栈
                }while(judge(Randfuhao(&symbol)) != 1);
                Putfuhao(&symbol); //括号内运算结束后使'('出栈
        }
    }
}
free(str); //释放内存空间
while(symbol.top != -1) {
        v2 = Putshuju(&data);
        v1 = Putshuju(&data);
        sum = Math(v1, v2, Putfuhao(&symbol));
        Inshuju(&data, sum);
}
printf("%d", data.data[0]);
```

```
    return 0;
}
```

我们来看看演示结果：

```
1+2+3+4+5=
15
--------------------------------
Process exited after 11.54 seconds with return value 0
请按任意键继续. . .
```

```
(2+5)*2=
14
--------------------------------
Process exited after 17.31 seconds with return value 0
请按任意键继续. . .
```

```
24/2/2=
6
--------------------------------
Process exited after 20.83 seconds with return value 0
请按任意键继续. . .
```