```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
/* INDICE : vous devez obtenir une complexité cyclomatique globale v(G),
    comprise entre 11 et 17, ce qui est mauvais : il faut entre 11 et17
     tests unitaires pour blinder ce code */
void extract(char* char1, char* char2, int debut, int nb);

int eval1(char* ch)
{
    int i;
    int valeur1, valeur2;
    int lgval2;
    char *val1, *val2;
    char operation;
    int resultat;
    /* Recherche d´un opérateur et de sa position */
    for( i=0 ; *(ch+i) !='+' && *(ch+i) !='-' && *(ch+i) !='*' && *(ch+i)
     !='/' && *(ch+i) !='\0'; i++)
    {
    }
    /* Traitement des erreurs */
    if(i==0) /* Le premier opérande manque */
    {
        printf("erreur : pas de <valeur1>");
        exit(0);
    }
    else if(i==strlen(ch)-1) /* Le deuxième opérande manque */
    {
        printf("erreur : pas de <valeur2>");
        exit(0);
    }
    else if(i==strlen(ch)) /* Il n´y a pas d´opérateur */
    {
        printf("erreur : pas de <operator>");
        exit(0);
    }
    /* char Extraction de la chaîne de caractère correspondant au
    premier opérande */
    val1=(char*) malloc((i+1)*sizeof(char));
    extract(ch,val1,0,i);
    /* Transformation de la chaîne de carctère en entier */
    sscanf(val1,"%d",& valeur1);
    /* Récuperation de l´opérateur */
    operation=*(ch+i);

    /* Extraction de la chaîne de caractère correspondant au deuxième
     opérande */
    lgval2=strlen(ch)-(i+1);
    val2=(char*) malloc((lgval2+1)*sizeof(char));
    extract(ch,val2,i+1,lgval2);
    /* Transformation de la chaîne de caractère en entier */
    sscanf(val2,"%d",&valeur2);
    /* Traitement de l´opération */
    switch(operation)
```

```
    {
        case '+':
            resultat=valeur1+valeur2;
            break;
        case '-':
            resultat=valeur1-valeur2;
            break;
        case '*':
            resultat=valeur1*valeur2;
            break;
        case '/':
            if(valeur2 != 0)
                resultat=valeur1/valeur2;
            else
                {
                resultat=0;
                printf("Erreur : impossible de diviser par 0");
                exit(0);
                }
    }
    return resultat;
}
/* Fonction qui extrait une sous-chaîne de chaîne1 dans chaîne2, de nb
 caractères à partir
du caractère début */
void extract(char* char1, char* char2, int debut, int nb)
{
    int i;
    char1= char1+debut;
    i=0;

    while(i<nb)
    {
        *char2= *char1;
        char1++;
        char2++;
        i++;
    }
    *char2='\n';
}

int main(int argc, char** argv)
{
    int res;
    if(argc!=2)
    {
        printf("Erreur, utilisation du programme : eval1<expression>");
    }
    else
    {
        res=eval1(argv[1]);
        printf("Le résultat de l´opération est : %d",res);
    }
}
```