

# C++ 语 言 程 序 设 计

## 实 验 报 告

### 实 验 一

姓名： 方尧

学号： 190410102

班级： 19 自动化 1 班

## 一 实验项目

- 1、熟悉 C++ 程序设计
- 2、掌握 C++ 基本输入输出方法
- 3、掌握 C++ 中 string 类型的使用方法
- 4、实现字符栈功能
- 5、实现表达式中数值与操作符的识别
- 6、实现表达式中括号匹配的判断

## 二 实验原理

- 数组的预处理及作用说明：

command[length] 用于存储接收进来的字符表达式，初始化全为 '='，为 char 型

num[length] 用于存储提取出的数值，初始化全为 0，为 double 型

brackets[length] 用于存放提取出的操作符，初始化为 0，为 int 型，值与字符对应关系为：

1-'{'，2-'['，3-'(', 4-')'，5-']'，6-')'，7-'+'，8-'-'，9-'\*'，10-'/'，11-'.'

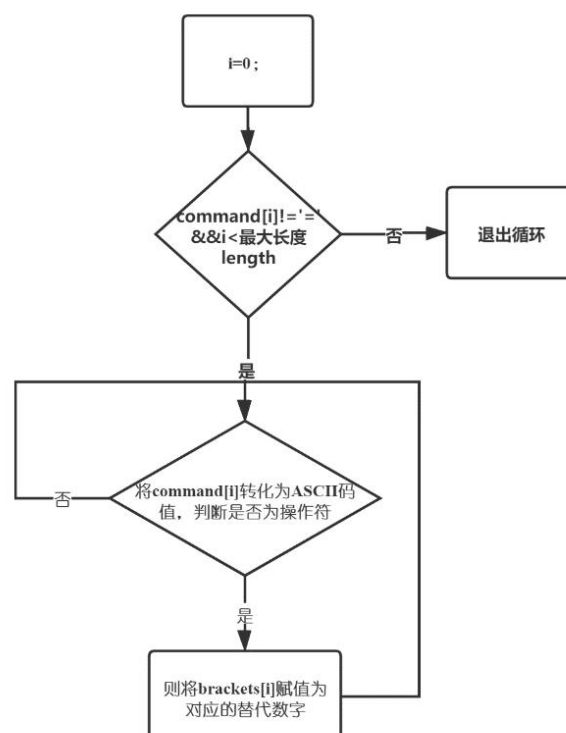
- 用到的 ASCII 码表：

figure 1 ASCII 码表

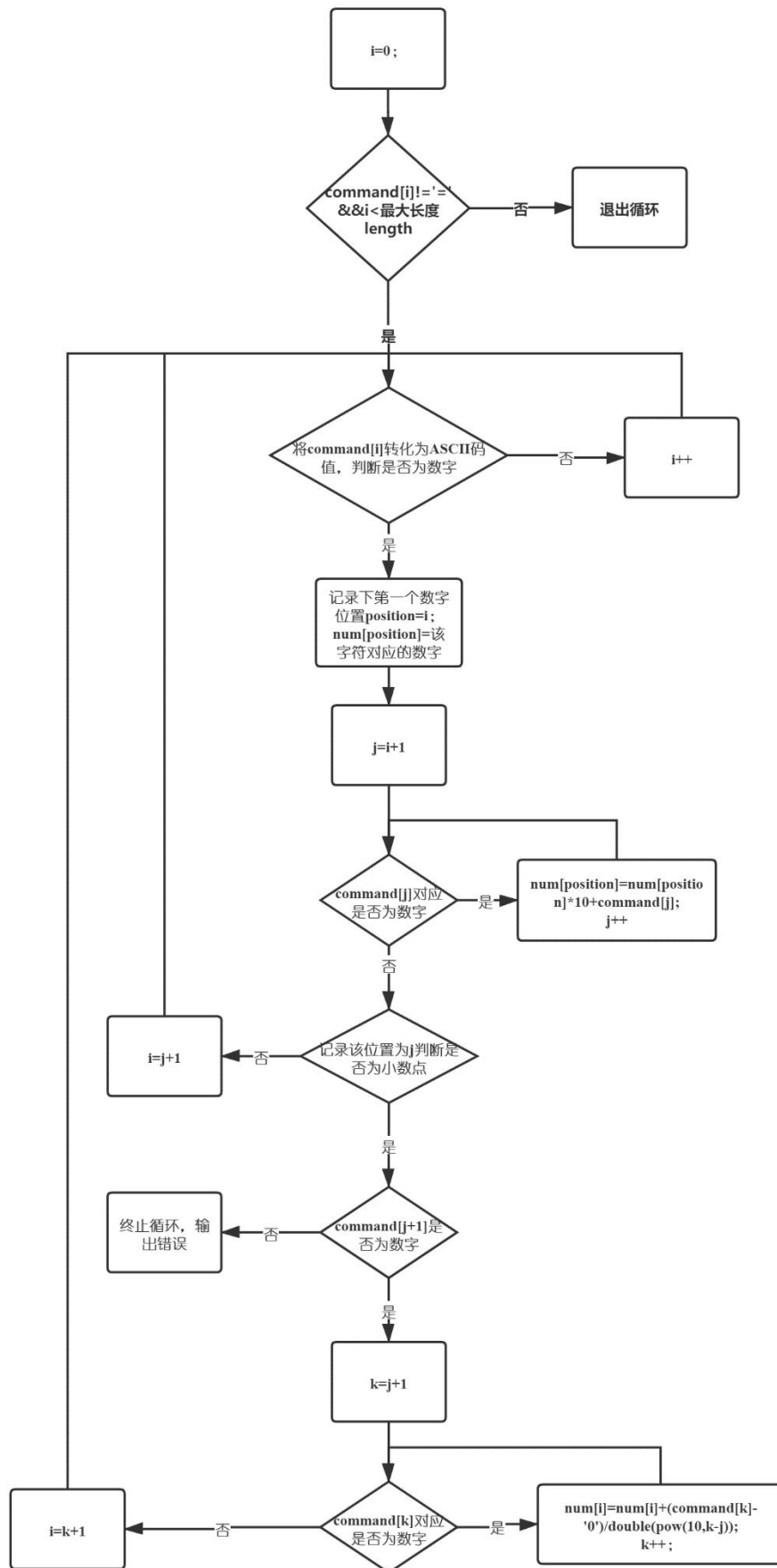
字符	ASCII 码	字符	ASCII 码	字符	ASCII 码	字符	ASCII 码
{	123	}	125	0	48	5	53
[	91	]	93	1	49	6	54
(	40	)	41	2	50	7	55
+	43	-	45	3	51	8	56
*	42	/	47	4	52	9	57
.	46						

- 1、字符串表达式中数值提取与操作符识别的方法

- 操作符的提取

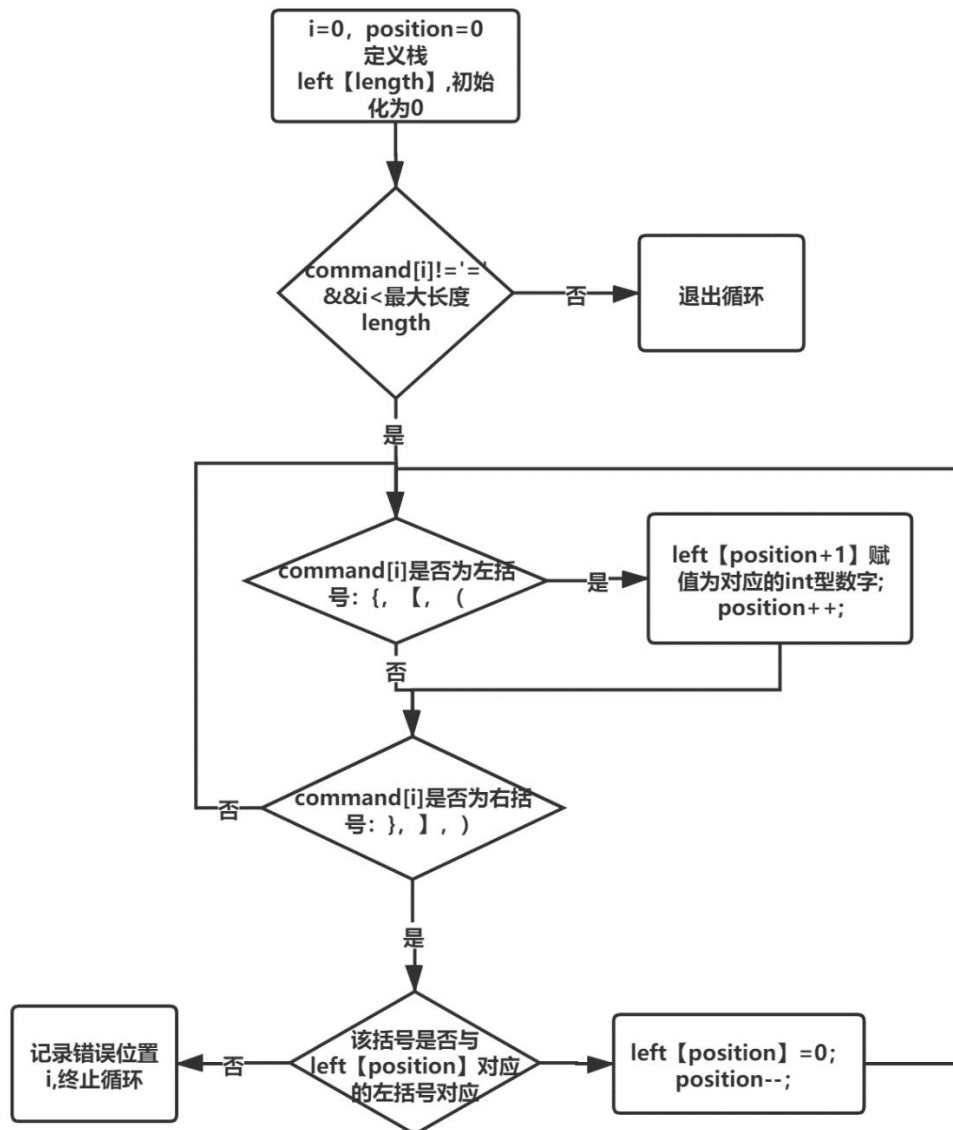


● 数值提取



说明：主要利用将字符型的输入指令通过 ASCII 码来对应可以实现的识别存储操作。`brackets` 同样下标位置存入 `command` 内该位置操作符对应的预先设定的整型数字；数值提取主要分为三阶段：第一阶段，提取整数部分，第二阶段，识别小数点，以及小数点的判断，第三部分，小数点后小数部分的提取。

## 2、表达式中括号匹配的判别方法



说明：在命令字符有效范围内循环，若为左括号，存入数组栈，并标记左括号的类型；若为右括号，与最新的左括号对比，判断是否为配对左右括号，若是，则删除最新的左括号标记；若否，终止循环，输出错误信息。直至命令字符组有效范围循环完毕，判断数组栈是否为空，若非空，输出错误信息；若空，则表明运算表达式正确。

### 3、程序的源代码、关键代码的操作含义以及运行结果

源代码见附录。

操作说明：

**input()**函数，输入函数，只允许给定运算符和数字输入，忽略错误输入（字母，特殊符号等）

**reset()**函数，每次输入命令前重置 **command**，**num**，**brackets**

**former()**函数，数值整数部分操作函数

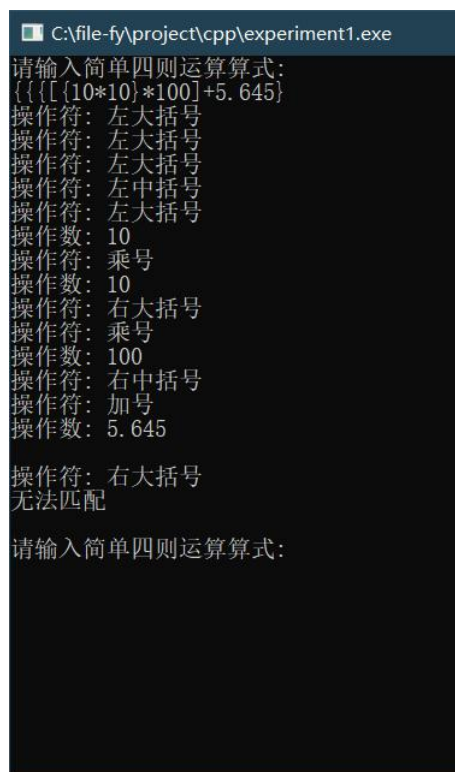
**later()**，数值小数部分操作函数

**spl()**函数，识别操作符，并将操作符信息存入 **brackets** 中；调用 **former()**和 **later()**函数，识别提取数值，并存入 **num** 中

**cal()**函数，判断括号匹配，若错误，输出错误信息。

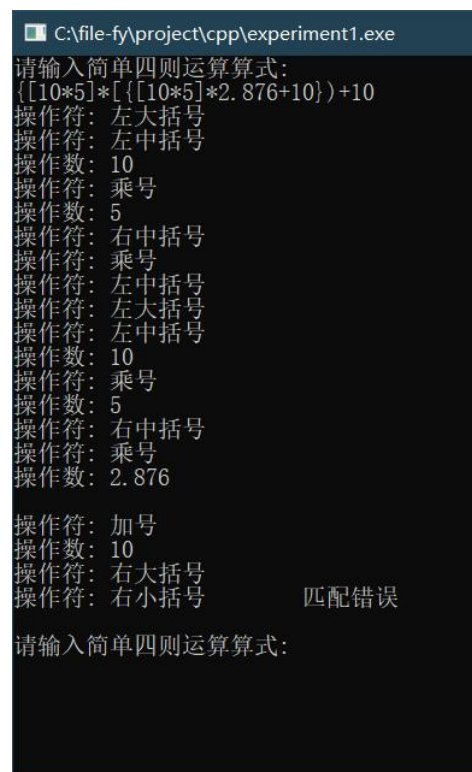
主要利用字符与 **ASCII** 码间的关系，识别字符串中的运算符以及数值。

运行结果：



```
C:\file-fy\project\cpp\experiment1.exe
请输入简单四则运算算式:
{{{[10*10]*100]+5.645}
操作符: 左大括号
操作符: 左大括号
操作符: 左大括号
操作符: 左中括号
操作符: 左大括号
操作数: 10
操作符: 乘号
操作数: 10
操作符: 右大括号
操作符: 乘号
操作数: 100
操作符: 右中括号
操作符: 加号
操作数: 5.645
操作符: 右大括号
无法匹配
请输入简单四则运算算式:
```

figure 2 错误示例-左括号多余



```
C:\file-fy\project\cpp\experiment1.exe
请输入简单四则运算算式:
{[10*5]*{[10*5]*2.876+10})+10
操作符: 左大括号
操作符: 左中括号
操作数: 10
操作符: 乘号
操作数: 5
操作符: 右中括号
操作符: 乘号
操作符: 左中括号
操作符: 左大括号
操作符: 左中括号
操作数: 10
操作符: 乘号
操作数: 5
操作符: 右中括号
操作符: 乘号
操作数: 2.876
操作符: 加号
操作数: 10
操作符: 右大括号
操作符: 右小括号
匹配错误
请输入简单四则运算算式:
```

figure 3 错误示例-右括号不匹配

### 三 实验总结与建议

实验实施过程：先构思；画框架图；编写伪代码；用实际代码实现；运行调试；**debug** 直至无错误并得到预期结果。

实验解决方案：实验中主要运用数组栈和字符与 **ASCII** 码的对应关系，识别操作符，分割数值的小数位和整数位，得到运算算式中的字符和操作符信息，得到解决。

## 附录：程序源代码

```
1. #include<iostream>
2. #include<math.h>
3. #define length 100
4.
5. using namespace std;
6. void spl(char command[],double num[],int brackets[]);
7. void cal(char command[],double num[],int brackets[]);
8. void reset(char command[],double num[],int brackets[]);
9. int former(int i,char command[],double num[]);
10. int later(int i,int j,char command[],double num[]);
11. void input(char command[]);
12.
13. int main( )
14. {
15.     char command[length];
16.     double num[length]= {0};
17.     int brackets[length]= {0}; //1-[,2-[,3-(,4-[,5-[,6-[,7-[,8-[,9-[,10-[,11-.
18.     reset(command,num,brackets);
19.
20.     for(int i=1;; i++)
21.     {
22.         cout<<"请输入简单四则运算算式:"<<endl;
23.         //输入
24.         input(command);
25.         spl(command,num,brackets);
26.         cal(command,num,brackets);
27.         reset(command,num,brackets);
28.     }
29.     return 0;
30. }
31. void input(char command[])
32. {
33.     char a;
34.     int b=61;
35.     int i=0;
36.     for(;;)
37.     {
38.         a=getchar();
39.         b=a;
40.         if(b==123||b==125||b==91||b==93\
41.            ||b==40||b==41||b==46||b==43||b==45\
42.            ||b==42||b==47||(b>=48&&b<=57)||a=='='||a=='\n')
43.         {
44.             if(a=='='||a=='\n')
45.                 break;
46.             command[i]=a;
47.             i++;
48.         }
49.     }
50. }
51. void reset(char command[],double num[],int brackets[])
52. {
53.     int i,j;
54.     for(int i=0; i<length; i++)
55.     {
56.         command[i]='=';
57.         num[i]=0;
58.         brackets[i]=0;
59.     }
60. }
61. void spl(char command[],double num[],int brackets[])
62. {
63.
64.     for(int i=0; command[i]!='='&&i<length; i++)
65.     {
```

```

66.         switch (command[i])
67.         {
68.             case '{':
69.                 brackets[i]=1;
70.                 break;
71.             case '}':
72.                 brackets[i]=2;
73.                 break;
74.             case '[':
75.                 brackets[i]=3;
76.                 break;
77.             case ']':
78.                 brackets[i]=4;
79.                 break;
80.             case '(':
81.                 brackets[i]=5;
82.                 break;
83.             case ')':
84.                 brackets[i]=6;
85.                 break;
86.             case '+':
87.                 brackets[i]=7;
88.                 break;
89.             case '-':
90.                 brackets[i]=8;
91.                 break;
92.             case '*':
93.                 brackets[i]=9;
94.                 break;
95.             case '/':
96.                 brackets[i]=10;
97.                 break;
98.             case '.':
99.                 brackets[i]=11;
100.                break;
101.        }
102.    }
103.    for(int i=0; command[i]!='&&i<length; i++)
104.    {
105.        int j,k;
106.        if(int(command[i])>=48&&int(command[i])<=57)
107.        {
108.            num[i]=command[i]-'0';
109.            j=former(i,command,num);
110.            k=later(i,j,command,num);
111.            if(k!=0)
112.                i=k;
113.            else
114.            {
115.                i=j;
116.                cout<<"wrong\n";
117.                /*错误输出*/
118.            }
119.        }
120.    }
121. }
122.
123. //小数点前数字处理
124. int former(int i,char command[],double num[])
125. {
126.     int j;
127.     for(j=i+1; command[j]!='&&j<length; j++)
128.     {
129.         if(int(command[j])>=48&&int(command[j])<=57)
130.         {
131.             num[i]=num[i]*10+command[j]-'0';
132.         }
133.         else
134.             break;

```

```

135.     }
136.     return j;
137. }
138.
139. //小数点后的数字处理
140. int later(int i,int j,char command[],double num[])
141. {
142.     int k=0;
143.     if(int(command[j])==46)
144.     {
145.         if(!(command[j+1]>=48&&command[j+1]<=57))
146.             return 0;
147.         else
148.         {
149.             for(int k=j+1; command[k]!='&&k<length; k++)
150.             {
151.                 if(int(command[k])>=48&&int(command[k])<=57)
152.                 {
153.                     num[i]=num[i]+(command[k]-'0')/double(pow(10,k-j));
154.                 }
155.                 else
156.                     return k;
157.             }
158.         }
159.     }
160.     return j;
161. }
162. void cal(char command[],double num[],int brackets[])
163. {
164.     double operater1,operater2;
165.     int wrong=0;
166.     //left 从第二个存储位置开始存
167.     int left[50]= {0};
168.     int position=0;
169.     //判断括号是否匹配
170.     for(int i=0; command[i]!='&&i<length; i++)
171.     {
172.         if(brackets[i]==1||brackets[i]==3||brackets[i]==5)
173.         {
174.             left[position+1]=brackets[i];
175.             position++;
176.         }
177.         if(brackets[i]==2||brackets[i]==4||brackets[i]==6)
178.         {
179.             if(brackets[i]==2)
180.             {
181.                 if(left[position]!=1)
182.                 {
183.                     wrong=i;
184.                     break;
185.                 }
186.                 else
187.                 {
188.                     left[position]=0;
189.                     position--;
190.                 }
191.             }
192.             if(brackets[i]==4)
193.             {
194.                 if(left[position]!=3)
195.                 {
196.                     wrong=i;
197.                     break;
198.                 }
199.                 else
200.                 {
201.                     left[position]=0;
202.                 }
203.             }

```



```

204.             position--;
205.         }
206.     }
207.     if(brackets[i]==6)
208.     {
209.         if(left[position]!=5)
210.         {
211.             wrong=i;
212.             break;
213.         }
214.         else
215.         {
216.             left[position]=0;
217.             position--;
218.         }
219.     }
220. }
221. }
222. if(wrong!=0)
223. {
224.     for(int i=0; command[i]!='='&&i<length&&i<=wrong; i++)
225.     {
226.         if(num[i]!=0)
227.         {
228.             cout<<"操作数:\t"<<num[i]<<endl;
229.         }
230.         if(brackets[i]!=0)
231.         {
232.             switch(brackets[i])
233.             {
234.                 case 1:
235.                     cout<<"操作符:\t 左大括号\t";
236.                     break;
237.                 case 2:
238.                     cout<<"操作符:\t 右大括号\t";
239.                     break;
240.                 case 3:
241.                     cout<<"操作符:\t 左中括号\t";
242.                     break;
243.                 case 4:
244.                     cout<<"操作符:\t 右中括号\t";
245.                     break;
246.                 case 5:
247.                     cout<<"操作符:\t 左小括号\t";
248.                     break;
249.                 case 6:
250.                     cout<<"操作符:\t 右小括号\t";
251.                     break;
252.                 case 7:
253.                     cout<<"操作符:\t 加号\t";
254.                     break;
255.                 case 8:
256.                     cout<<"操作符:\t 减号\t";
257.                     break;
258.                 case 9:
259.                     cout<<"操作符:\t 乘号\t";
260.                     break;
261.                 case 10:
262.                     cout<<"操作符:\t 除号\t";
263.                     break;
264.             }
265.             if(i==wrong)
266.                 cout<<"匹配错误"<<endl;
267.             cout<<endl;
268.         }
269.     }
270. }
271. else
272. {

```

```

273.         if(position==0)
274.         {
275.             //正确, 输出计算结果
276.             double result=0;
277.
278.             for(int j=0; j<length&&command[j]!='='; j++)
279.             {
280.                 cout<<command[j];
281.             }
282.             cout<<"="<<result<<endl;
283.         }
284.     else
285.     {
286.         for(int i=0; command[i]!='='&&i<length; i++)
287.         {
288.             if(num[i]!=0)
289.             {
290.                 cout<<"操作数:\t"<<num[i]<<endl;
291.             }
292.             if(brackets[i]!=0)
293.             {
294.                 switch(brackets[i])
295.                 {
296.                     case 1:
297.                         cout<<"操作符:\t 左大括号\t";
298.                         break;
299.                     case 2:
300.                         cout<<"操作符:\t 右大括号\t";
301.                         break;
302.                     case 3:
303.                         cout<<"操作符:\t 左中括号\t";
304.                         break;
305.                     case 4:
306.                         cout<<"操作符:\t 右中括号\t";
307.                         break;
308.                     case 5:
309.                         cout<<"操作符:\t 左小括号\t";
310.                         break;
311.                     case 6:
312.                         cout<<"操作符:\t 右小括号\t";
313.                         break;
314.                     case 7:
315.                         cout<<"操作符:\t 加号\t";
316.                         break;
317.                     case 8:
318.                         cout<<"操作符:\t 减号\t";
319.                         break;
320.                     case 9:
321.                         cout<<"操作符:\t 乘号\t";
322.                         break;
323.                     case 10:
324.                         cout<<"操作符:\t 除号\t";
325.                         break;
326.                 }
327.                 if(command[i+1]=='=')
328.                     cout<<endl<<"无法匹配"<<endl;
329.                 cout<<endl;
330.             }
331.         }
332.     }
333. }
334. }

```