# 栈操作（栈-基本题）

<testDataSet count="5">  
    <testData1>  
        <input>1 567 0 -1  
</input>  
        <output>567 </output>  
    </testData1>  
    <testData2>  
        <input>1 56 1 35 1 89 1 -23 1 -8 1 -2 1 901 1 655 0 0 0 0 0 0 0 0 -1  
</input>  
        <output>655 901 -2 -8 -23 89 35 56 </output>  
    </testData2>  
    <testData3>  
        <input>1 3 1 2 1 8 0 0 0 1 5 1 7 1 8 0 1 78 1 89 0 1 35 1 788 1 23 0 0 0 0 0 0 -1  
</input>  
        <output>8 2 3 8 89 23 788 35 78 7 5 </output>  
    </testData3>  
    <testData4>  
        <input>1 5 0 0 1 8 0 1 90 0 1 356 0 1 234 0 1 890 0 1 2 0 1 -56 0 1 -45 0 -1  
</input>  
        <output>5 error 8 90 356 234 890 2 -56 -45 </output>  
    </testData4>  
    <testData5>  
        <input>1 5 1 8 1 90 0 0 0 0 0 0 0 1 356 1 234 0 1 890 0 0 0 0 1 2 1 5 1 -56 0 1 -45 0 -1  
</input>  
        <output>90 8 5 error error error error 234 890 356 error error -56 -45 </output>  
    </testData5>  
</testDataSet>

# C程序括号匹配检查

<testDataSet count="5">  
    <testData1>  
        <input filename="example.c">#include<stdio.h>

#include<string.h>

#include<stdbool.h>

typedef struct WordList{

char \*word;

int count;

struct WordList \*next;

}list;

list\* createNode(char \*word){

list \*tmp=(char\*)malloc(sizeof(list));

tmp->word=(char \*)malloc((strlen(word)+1)\*sizeof(char));

strcpy(tmp->word,word);

tmp->count=1;

return tmp;

}

int main(){

FILE \*fp=fopen("article.txt(}","r}{");

if(!fp){

printf("can't open file!\n");

return 0;

}

rewind(fp);

char line[1000];

list \*head=NULL;

while(fgets(line,1000,fp!=NULL){

int i=0;

char \*words;

int l=strlen(line);

while(i<l&&line[i]!='\n'){/\*zhushi\*\*/

while(i<l&&!isalpha(line[i])&&line[i]!='\n'){

i++;

}

if(i>=l||line[i]=='\n') break;

words=line+i;

while(i<l&&isalpha(line[i])){

if(isupper(line[i])){//如果是大写字母

line[i]=tolower(line[i]);

}

i++;

}

line[i]='\0';

//printf("%s\n",words);

if(head==NULL){

head=(list\*)malloc(sizeof(list)\*1);

head->word=(char\*)malloc((strlen(words)+1)\*sizeof(char));

strcpy(head->word,words);

head->count=1;

head->next=NULL;

}else{

list \*pre=NULL;

list \*current=head;

if(strcmp(words,current->word)<0){//应该插入到头结点

list \*tmp=createNode(words);

tmp->next=current;

head=tmp;

}else{

while(current!=NULL&&(strcmp(words,current->word)>0)){

pre=current;

current=current->next;

}

if(current==NULL||strcmp(words,current->word)<0){//插入到末尾或者是current之前

list \*tmp=createNode(words);

pre->next=tmp;

tmp->next=current;

}else{//找到了对应的单词 对应的count值增加

(current->count)++;

}

}

}

i++;

}

}

list \*tmp=head;

while(tmp!=NULL){

printf("%s %d\n",tmp->word,tmp->count);

tmp=tmp->next;

}

}

</input>  
        <output>without maching '(' at line 25</output>  
    </testData1>  
    <testData2>  
        <input filename="example.c">#include<stdio.h>

#include<string.h>

#include<stdbool.h>

typedef struct WordList{

char \*word;

int count;

struct WordList \*next;

}list;

list\* createNode(char \*word){

list \*tmp=(char\*)malloc(sizeof(list));

tmp->word=(char \*)malloc((strlen(word)+1)\*sizeof(char));

strcpy(tmp->word,word);

tmp->count=1;

return tmp;

}

int main(){

FILE \*fp=fopen("article.txt())","r");

if(!fp){

printf("can't open file!\n");

return 0;

}

rewind(fp);

char line[1000];

list \*head=NULL;

while(fgets(line,1000,fp)!=NULL){

int i=0;

char \*words;

int l=strlenline);

while(i<l&&line[i]!='\n'){/\*zhushi\*\*/

while(i<l&&!isalpha(line[i])&&line[i]!='\n'){

i++;

}

if(i>=l||line[i]=='\n') break;//}} /\*/

words=line+i;

while(i<l&&isalpha(line[i])){

if(isupper(line[i])){//如果是大写字母

line[i]=tolower(line[i]);

}

i++;

}

line[i]='\0';

//printf("%s\n",words);

if(head==NULL){

head=(list\*)malloc(sizeof(list)\*1);

head->word=(char\*)malloc((strlen(words)+1)\*sizeof(char));

strcpy(head->word,words);

head->count=1;

head->next=NULL;

}else{

list \*pre=NULL;

list \*current=head;

if(strcmp(words,current->word)<0){//应该插入到头结点

list \*tmp=createNode(words);

tmp->next=current;

head=tmp;

}else{

while(current!=NULL&&(strcmp(words,current->word)>0)){

pre=current;

current=current->next;

}

if(current==NULL||strcmp(words,current->word)<0){//插入到末尾或者是current之前

list \*tmp=createNode(words);

pre->next=tmp;

tmp->next=current;

}else{//找到了对应的单词 对应的count值增加

(current->count)++;

}

}

}

i++;

}

}

list \*tmp=head;

while(tmp!=NULL){

printf("%s %d\n",tmp->word,tmp->count);

tmp=tmp->next;

}

}

</input>  
        <output>without maching ')' at line 28</output>  
    </testData2>  
    <testData3>  
        <input filename="example.c">#include<stdio.h>

#include<string.h>

#include<stdbool.h>

typedef struct WordList{

char \*word;

int count;

struct WordList \*next;

}list;

list\* createNode(char \*word){

list \*tmp=(char\*)malloc(sizeof(list));

tmp->word=(char \*)malloc((strlen(word)+1)\*sizeof(char));

strcpy(tmp->word,word);

tmp->count=1;

return tmp;

}

int main(){

FILE \*fp=fopen("article.txt())","r");

if(!fp){

printf("can't open file!\n");

return 0;

}

rewind(fp);

char line[1000];

list \*head=NULL;

while(fgets(line,1000,fp)!=NULL){

int i=0;

char \*words;

int l=strlen(line);

while(i<l&&line[i]!='\n'){/\*zhushi\*\*/

while(i<l&&!isalpha(line[i])&&line[i]!='\n'){

i++;

}

if(i>=l||line[i]=='\n') break;// /\*/

words=line+i;

while(i<l&&isalpha(line[i])){

if(isupper(line[i])){//如果是大写字母

line[i]=tolower(line[i]);

}

i++;

}

line[i]='\0';

//printf("%s\n",words);

if(head==NULL){

head=(list\*)malloc(sizeof(list)\*1);

head->word=(char\*)malloc((strlen(words)+1)\*sizeof(char));

strcpy(head->word,words);

head->count=1;

head->next=NULL;

}else{

list \*pre=NULL;

list \*current=head;

if(strcmp(words,current->word)<0){//应该插入到头结点

list \*tmp=createNode(words);

tmp->next=current;

head=tmp;

}else{

while(current!=NULL&&(strcmp(words,current->word)>0)){

pre=current;

current=current->next;

}

if(current==NULL||strcmp(words,current->word)<0){//插入到末尾或者是current之前

list \*tmp=createNode(words);

pre->next=tmp;

tmp->next=current;

}else{//找到了对应的单词 对应的count值增加

(current->count)++;

}

}

}

i++;

}

}

list \*tmp=head;

while(tmp!=NULL){

printf("%s %d\n",tmp->word,tmp->count);

tmp=tmp->next;

}

}

}

</input>  
        <output>without maching '}' at line 80</output>  
    </testData3>  
    <testData4>  
        <input filename="example.c">#include<stdio.h>

#include<string.h>

#include<stdbool.h>

typedef struct WordList{

char \*word;

int count;

struct WordList \*next;

}list;

list\* createNode(char \*word){

list \*tmp=(char\*)malloc(sizeof(list));

tmp->word=(char \*)malloc((strlen(word)+1)\*sizeof(char));

strcpy(tmp->word,word);

tmp->count=1;

return tmp;

}

{

int main(){

FILE \*fp=fopen("article.txt())","r");

if(!fp){

printf("can't open file!\n");

return 0;

}

rewind(fp);

char line[1000];

list \*head=NULL;

while(fgets(line,1000,fp)!=NULL){

int i=0;

char \*words;

int l=strlen(line);

while(i<l&&line[i]!='\n'){/\*zhushi\*\*/

while(i<l&&!isalpha(line[i])&&line[i]!='\n'){

i++;

}

if(i>=l||line[i]=='\n') break;// /\*/

words=line+i;

while(i<l&&isalpha(line[i])){

if(isupper(line[i])){//如果是大写字母

line[i]=tolower(line[i]);

}

i++;

}

line[i]='\0';

//printf("%s\n",words);

if(head==NULL){

head=(list\*)malloc(sizeof(list)\*1);

head->word=(char\*)malloc((strlen(words)+1)\*sizeof(char));

strcpy(head->word,words);

head->count=1;

head->next=NULL;

}else{

list \*pre=NULL;

list \*current=head;

if(strcmp(words,current->word)<0){//应该插入到头结点

list \*tmp=createNode(words);

tmp->next=current;

head=tmp;

}else{

while(current!=NULL&&(strcmp(words,current->word)>0)){

pre=current;

current=current->next;

}

if(current==NULL||strcmp(words,current->word)<0){//插入到末尾或者是current之前

list \*tmp=createNode(words);

pre->next=tmp;

tmp->next=current;

}else{//找到了对应的单词 对应的count值增加

(current->count)++;

}

}

}

i++;

}

}

list \*tmp=head;

while(tmp!=NULL){

printf("%s %d\n",tmp->word,tmp->count);

tmp=tmp->next;

}

}

</input>  
        <output>without maching '{' at line 16</output>  
    </testData4>  
    <testData5>  
        <input filename="example.c">#include<stdio.h>

#include<string.h>

#include<stdbool.h>

typedef struct WordList{

char \*word;

int count;

struct WordList \*next;

}list;

list\* createNode(char \*word){

list \*tmp=(char\*)malloc(sizeof(list));

tmp->word=(char \*)malloc((strlen(word)+1)\*sizeof(char));

strcpy(tmp->word,word);

tmp->count=1;

return tmp;

}

int main(){

FILE \*fp=fopen("article.txt())","r");

if(!fp){

printf("can't open file!\n");

return 0;

}

rewind(fp);

char line[1000];

list \*head=NULL;

while(fgets(line,1000,fp)!=NULL){

int i=0;

char \*words;

int l=strlen(line);

while(i<l&&line[i]!='\n'){/\*zhushi\*\*/

while(i<l&&!isalpha(line[i])&&line[i]!='\n'){

i++;

}

if(i>=l||line[i]=='\n') break;// /\*/

words=line+i;

while(i<l&&isalpha(line[i])){

if(isupper(line[i])){//如果是大写字母

line[i]=tolower(line[i]);

}

i++;

}

line[i]='\0';

//printf("%s\n",words);

if(head==NULL){

head=(list\*)malloc(sizeof(list)\*1);

head->word=(char\*)malloc((strlen(words)+1)\*sizeof(char));

strcpy(head->word,words);

head->count=1;

head->next=NULL;

}else{

list \*pre=NULL;

list \*current=head;

if(strcmp(words,current->word)<0){//应该插入到头结点

list \*tmp=createNode(words);

tmp->next=current;

head=tmp;

}else{

while(current!=NULL&&(strcmp(words,current->word)>0)){

pre=current;

current=current->next;

}

if(current==NULL||strcmp(words,current->word)<0){//插入到末尾或者是current之前

list \*tmp=createNode(words);

pre->next=tmp;

tmp->next=current;

}else{//找到了对应的单词 对应的count值增加

(current->count)++;

}

}

}

i++;

}

}

list \*tmp=head;

while(tmp!=NULL){

printf("%s %d\n",tmp->word,tmp->count);

tmp=tmp->next;

}

}

</input>  
        <output>{}(){()(())()((())())()}(){()(){()}()(()){()(){(()){}()(()){(()){()}}(){()(())()((())())()}{(()){()}{((())){}(()){()}{()}}}}}(){()}}</output>  
    </testData5>  
</testDataSet>

# 计算器（表达式计算-后缀表达式实现）

<testDataSet count="5">  
    <testData1>  
        <input>12 \* 15 + 12 /4 =  
</input>  
        <output>183</output>  
    </testData1>  
    <testData2>  
        <input>12 \* (15 + 12) /4       =   
</input>  
        <output>81</output>  
    </testData2>  
    <testData3>  
        <input>((12+13)/5+(    (100+15%13)/4-16)\*4)/   6   =      
</input>  
        <output>6</output>  
    </testData3>  
    <testData4>  
        <input>(((1+2)+3)+4)+5 =  
</input>  
        <output>15</output>  
    </testData4>  
    <testData5>  
        <input>(((1+2)+13%10)+4)\*  (5-5) = </input>  
        <output>0</output>  
    </testData5>  
</testDataSet>

# 文本编辑操作模拟（简）a

<testDataSet count="5">  
    <testData1>  
        <input>The standards committee was aware of the multi-national use of the C language, and thus provided, both in the  
1  
1 0 The  
  
2 4 5  
1 4 STAND  
-1  
</input>  
        <output>The STANDards committee was aware of the multi-national use of the C language, and thus provided, both in the</output>  
    </testData1>  
    <testData2>  
        <input>The standards committee was aware of the multi-national use of the C language, and thus provided, both in the  
3  
1 0 The  
1 109 the  
2 98 the  
  
3  
3  
3  
-1  
</input>  
        <output> standards committee was aware of the multi-national use of the C language, and thus provided, theboth in </output>  
    </testData2>  
    <testData3>  
        <input>The standards committee was aware of the multi-national use of the C language, and thus provided, both in the  
3  
1 0 The  
1 109 the  
2 98 the  
  
2 0 106  
1 3 standards  
-1  
</input>  
        <output>thestandards</output>  
    </testData3>  
    <testData4>  
        <input>The standards committee was aware of the multi-national use of the C language, and thus provided, both in the  
3  
1 0 The  
1 109 the  
2 98 the  
  
2 0 103  
1 6 standards  
3  
1 0 both  
3  
3  
3  
3  
-1  
</input>  
        <output>The standards committee was aware of the multi-national use of the C language, and thus provided, theboth in </output>  
    </testData4>  
    <testData5>  
        <input>The standards committee was aware of the multi-national use of the C language, and thus provided, both in the  
3  
1 0 The  
1 109 the  
2 98 the  
  
2 97 100  
1 97 both  
1 101 -  
3  
3  
1 97 -both  
1 35 -  
3  
3  
3  
3  
3  
3  
3  
3  
-1  
</input>  
        <output> standards committee was aware of the multi-national use of the C language, and thus provided, theboth in </output>  
    </testData5>  
</testDataSet>

# 银行排队模拟（生产者-消费者模拟）

<testDataSet count="5">  
    <testData1>  
        <input>6  
2  5  13  11  15   9  
</input>  
        <output>1 : 0  
2 : 0  
3 : 0  
4 : 0  
5 : 0  
6 : 1  
7 : 1  
8 : 0  
9 : 1  
10 : 1  
11 : 1  
12 : 1  
13 : 2  
14 : 2  
15 : 2  
16 : 2  
17 : 2  
18 : 3  
19 : 3  
20 : 3  
21 : 2  
22 : 2  
23 : 3  
24 : 3  
25 : 3  
26 : 3  
27 : 4  
28 : 4  
29 : 4  
30 : 4  
31 : 5  
32 : 4  
33 : 4  
34 : 5  
35 : 5  
36 : 5  
37 : 6  
38 : 6  
39 : 6  
40 : 7  
41 : 7  
42 : 7  
43 : 8  
44 : 8  
45 : 8  
46 : 9  
47 : 8  
48 : 8  
49 : 9  
50 : 9  
51 : 9  
52 : 10  
53 : 10  
54 : 10  
55 : 11  
</output>  
    </testData1>  
    <testData2>  
        <input>10  
1 5 10 15 25 15 10 5 1 1  
</input>  
        <output>1 : 0  
2 : 0  
3 : 0  
4 : 0  
5 : 1  
6 : 1  
7 : 0  
8 : 1  
9 : 1  
10 : 1  
11 : 1  
12 : 2  
13 : 2  
14 : 2  
15 : 2  
16 : 2  
17 : 2  
18 : 2  
19 : 2  
20 : 2  
21 : 2  
22 : 3  
23 : 3  
24 : 3  
25 : 3  
26 : 3  
27 : 4  
28 : 4  
29 : 4  
30 : 4  
31 : 4  
32 : 4  
33 : 4  
34 : 4  
35 : 4  
36 : 4  
37 : 5  
38 : 5  
39 : 5  
40 : 5  
41 : 5  
42 : 6  
43 : 6  
44 : 6  
45 : 6  
46 : 6  
47 : 7  
48 : 7  
49 : 7  
50 : 7  
51 : 7  
52 : 8  
53 : 8  
54 : 8  
55 : 8  
56 : 8  
57 : 8  
58 : 8  
59 : 8  
60 : 8  
61 : 9  
62 : 9  
63 : 9  
64 : 9  
65 : 10  
66 : 10  
67 : 10  
68 : 11  
69 : 11  
70 : 11  
71 : 12  
72 : 11  
73 : 11  
74 : 12  
75 : 12  
76 : 12  
77 : 13  
78 : 13  
79 : 13  
80 : 14  
81 : 14  
82 : 13  
83 : 14  
84 : 14  
85 : 14  
86 : 15  
87 : 14  
88 : 13  
</output>  
    </testData2>  
    <testData3>  
        <input>3  
1 2 3  
</input>  
        <output>1 : 0  
2 : 0  
3 : 0  
4 : 0  
5 : 0  
6 : 0  
</output>  
    </testData3>  
    <testData4>  
        <input>1  
50  
</input>  
        <output>1 : 0  
2 : 0  
3 : 0  
4 : 0  
5 : 0  
6 : 1  
7 : 1  
8 : 1  
9 : 1  
10 : 1  
11 : 2  
12 : 2  
13 : 2  
14 : 2  
15 : 2  
16 : 3  
17 : 3  
18 : 3  
19 : 3  
20 : 3  
21 : 4  
22 : 4  
23 : 4  
24 : 4  
25 : 5  
26 : 5  
27 : 5  
28 : 6  
29 : 6  
30 : 6  
31 : 7  
32 : 7  
33 : 7  
34 : 8  
35 : 8  
36 : 8  
37 : 9  
38 : 9  
39 : 9  
40 : 10  
41 : 10  
42 : 10  
43 : 11  
44 : 11  
45 : 11  
46 : 12  
47 : 12  
48 : 12  
49 : 13  
50 : 13  
</output>  
    </testData4>  
    <testData5>  
        <input>5  
0 0 0 0 10  
</input>  
        <output>1 : 0  
2 : 0  
3 : 0  
4 : 1  
5 : 1  
6 : 1  
7 : 2  
8 : 2  
9 : 2  
10 : 3  
</output>  
    </testData5>  
</testDataSet>

# 火车货运调度模拟（选做，不计分）

<testDataSet count="5">  
    <testData1>  
        <input>1  
shijiazhuang 280  
5  
0039 shijiazhuang  
5217 shijiazhuang  
0262 shijiazhuang  
2122 shijiazhuang  
3280 shijiazhuang  
</input>  
        <output>0039 5217 0262 2122 3280   
5</output>  
    </testData1>  
    <testData2>  
        <input>10  
shijiazhuang 280  
xingtai 390  
xinxiang 610  
zhengzhou 689  
wuchang 1221  
chibi 1339  
yueyang 1434  
changsha 1559  
shaoguan 2057  
guangzhou 2273  
10  
8911 guangzhou  
5217 zhengzhou  
3231 yueyang  
7222 wuchang  
3333 shijiazhuang  
2115 shaoguan  
6732 xingtai  
1234 chibi  
9090 changsha  
8765 xinxiang  
</input>  
        <output>8911 2115 9090 3231 1234 7222 5217 8765 6732 3333   
36  
</output>  
    </testData2>  
    <testData3>  
        <input>10  
shijiazhuang 280  
xingtai 390  
xinxiang 610  
zhengzhou 689  
wuchang 1221  
chibi 1339  
yueyang 1434  
changsha 1559  
shaoguan 2057  
guangzhou 2273  
12  
8911 guangzhou  
5217 zhengzhou  
3231 yueyang  
7222 wuchang  
1516 guangzhou  
3333 shijiazhuang  
2115 shaoguan  
6732 guangzhou  
1234 chibi  
9090 changsha  
8765 xinxiang  
3555 guangzhou  
</input>  
        <output>8911 1516 6732 3555 2115 9090 3231 1234 7222 5217 8765 3333   
52</output>  
    </testData3>  
    <testData4>  
        <input>10  
shijiazhuang 280  
xingtai 390  
xinxiang 610  
zhengzhou 689  
wuchang 1221  
chibi 1339  
yueyang 1434  
changsha 1559  
shaoguan 2057  
guangzhou 2273  
20  
8911 guangzhou  
5217 zhengzhou  
3231 yueyang  
7222 wuchang  
1516 guangzhou  
3333 shijiazhuang  
2115 shaoguan  
6732 guangzhou  
1234 chibi  
9090 changsha  
8765 xinxiang  
3555 guangzhou  
1111 shijiazhuang  
2222 shijiazhuang  
3334 shijiazhuang  
4444 shijiazhuang  
5555 shijiazhuang  
6666 shijiazhuang  
7777 shijiazhuang  
8888 shijiazhuang  
</input>  
        <output>8911 1516 6732 3555 2115 9090 3231 1234 7222 5217 8765 3333 1111 2222 3334 4444 5555 6666 7777 8888   
60</output>  
    </testData4>  
    <testData5>  
        <input>10  
shijiazhuang 280  
xingtai 390  
xinxiang 610  
zhengzhou 689  
wuchang 1221  
chibi 1339  
yueyang 1434  
changsha 1559  
shaoguan 2057  
guangzhou 2273  
20  
0111 shaoguan  
0222 changsha  
0022 wuchang  
0102 chibi  
0333 zhengzhou  
0403 changsha  
0809 yueyang  
0909 wuchang  
8911 guangzhou  
5217 zhengzhou  
3231 yueyang  
7222 wuchang  
1516 guangzhou  
3334 shijiazhuang  
2115 shaoguan  
6732 guangzhou  
1234 chibi  
9090 changsha  
8765 xinxiang  
3555 guangzhou  
</input>  
        <output>8911 1516 6732 3555 0111 2115 0222 0403 9090 0809 3231 0102 1234 0022 0909 7222 0333 5217 8765 3334   
125</output>  
    </testData5>  
</testDataSet>